



Michelangelo Ceci · Donato Malerba · Letizia Tanca

SEBD 2007

**15th Italian Symposium
on Advanced Database Systems**

**Hotel del Levante
Torre Canne di Fasano (BR), Italy
June 17-20, 2007**

Michelangelo Ceci • Donato Malerba • Letizia Tanca

SEBD 2007

PROCEEDINGS OF THE
FIFTEENTH ITALIAN SYMPOSIUM
ON ADVANCED DATABASE SYSTEMS

Torre Canne di Fasano (BR), Italy
17-20 June, 2007

Organized and Supported

By

Università degli Studi di Bari
Dipartimento di Informatica

VOLUME EDITORS:

Michelangelo Ceci

Dipartimento di Informatica
Università degli Studi di Bari
ceci@di.uniba.it

Donato Malerba

Dipartimento di Informatica
Università degli Studi di Bari
malerba@di.uniba.it

Letizia Tanca

Dipartimento di Elettronica e Informazione
Politecnico di Milano
tanca@elet.polimi.it

Finito di stampare nel mese di Maggio 2007
da **Sagraf srl** – Z.I. Capurso (BA)
info@sagraf.net

Preface

This volume collects the papers and demos selected for presentation at the Fifteenth Italian Symposium on Advanced Database Systems (SEBD 2007), held at Torre Canne di Fasano (Brindisi), from the 17th to the 20th of June 2007. The first edition of SEBD was held in 1993 and it was just one of the ‘deliverables’ of a nationwide project funded by the Italian National Research Council. The success of the event encouraged organizers to repeat the experience year after year, until the symposium has become THE annual event of the Italian database research community.

The database research community plays a fundamental role in creating the technological infrastructure for information management. When the first edition of SEBD was organized, the demands placed on the classical relational database systems already required new solutions to answer challenges posed by emerging applications. A next-generation of database systems was envisaged to support new types of data (images, multimedia, spatial, temporal, etc.) and new data models and languages (object oriented, logic-programming based), new methodologies were proposed for the development of innovative database applications, alternative transaction models were studied for those applications (such as CAD and workflow systems) where data is ‘checked out’ and not restored for long time, novel methods for information integration were proposed to keep a pace with the newborn data warehouses, and results of the recently established area of human-computer interaction were applied to database management systems in order to sustain the spreading of this technology in a community of non-specialists.

Fifteen years later, we can say that some solutions to these challenges have been provided, to the point that databases now do a good job as storage managers for many applications. However, the requirements placed on database systems have drastically changed in the meantime, thus demanding new solutions to fit the altered landscape. The advent of the Web has introduced new protocols (HTML and XML, to cite some), different communication infrastructures (e.g., intranet and extranet), novel software architectures (e.g., Web Services) and new paradigms for data distribution, access and interoperation. Mobile and ubiquitous computing is creating a new class of applications and posing new challenges. In the mobile wireless computing environment of the future, a massive number of computing devices with limited computing, storage and communication capabilities, as well as limited battery power will query databases over wireless communication channels. Mobile clients will often be disconnected for prolonged periods of time and will connect to different data servers at different times. Considering the nature of data collected in ubiquitous environments, ensuring privacy and security becomes a big challenge. Other emerging applications, from scientific (weather forecasting, genome databases, astronomy) to leisure and entertainment, require ‘meaningful information’ rather than ‘raw data’. Database

queries are no longer expected to return a collection of stored data, but a set of patterns extracted from them which convey new useful pieces of knowledge.

As a consequence of all these rapid changes of context, the database research community is revisiting the database concepts and is looking for new technological infrastructures for data storage and information management. Databases as we know them are becoming a home for old data that legacy systems still retain for inertial reasons.

This shift of focus on the role of data in computing is also reflected by the papers published in this volume. Sixty-four contributions were originally submitted, of which nineteen were accepted as long papers and thirty were accepted as extended abstracts. Each submission was evaluated by three independent referees. Seven demos have also been selected for a Demo Session that provides a forum for researchers and practitioners who want to show their implemented software systems in an interactive setting. Most of the contributions came from universities and the CNR (Italian National Research Council), but we are glad to note that some papers were also submitted by people working in industry, where publication activity may be discouraged. We hope that SEBD will continue to be a gathering forum where people from academic and industrial environments meet, discuss and exchange experiences on the theory and practice of information storage and management.

Besides paper and demo presentations, the programme also featured two invited talks by Karl Aberer (EPFL, Lausanne, Switzerland) and Michael Kifer (State University of New York at Stony Brook, USA), and a tutorial given by Michael May (Fraunhofer Institute of Autonomous Intelligent Systems, Sankt Augustin, Germany).

We would like to thank all the authors who submitted papers and all the conference participants. We are also grateful to the members of the Programme Committee and the external referees for their thorough work in reviewing submitted contributions with expertise and patience, and the members of the SEBD Steering Committee for their support in awarding the two student prizes. A special thank is due to all the members of the Organizing Committee which made this event possible. We gratefully acknowledge the many sponsoring institutions and the patronage of the Dipartimento di Informatica of the University of Bari.

June 2007

Michelangelo Ceci
Donato Malerba
Letizia Tanca

External Referees

Fabrizio Angiulli	Vincenzo D'Elia	Andrea Mazzoni
Daniele Apiletti	Ivan Di Pietro	Michele Melchiori
Annalisa Appice	Yael Dubinsky	Diego Milano
Lorenzo Baldacci	Riccardo Dutto	Stefano Montanelli
Enrico Bertini	Nicola Fanizzi	Mirco Nanni
Devis Bianchini	Bettina Fazzinga	Riccardo Ortale
Francesco Bonchi	Mandreoli Federica	Francesco Parisi
Marco Botta	Alfo Ferrara	Wilma Penzo
Giulia Bruno	Alessandro Fiori	Antonio Perdichizzi
Francesco Buccafurri	Francesco Folino	Antonella Poggi
Gianluca Caminiti	Salvatore Garruzzo	Giovani Ponti
Gianluca Capuzzi	Paolo Garza	Luigi Pontieri
Antonella Carbonaro	Alberto Gemelli	Domenico Potena
Egidio Cardinale	Matteo Golfarelli	Andrea Proli
Michelangelo Ceci	Gianluigi Greco	Giovanni Quattrone
Jessica Cellini	Francesco Gullo	Enrico Ronchetti
Tania Cerquitelli	Stephen Kimani	Domenico Rosaci
Eugenio Cesario	Gianluca Lax	Marco Ruzzi
Stefano Chessa	Stefano Lodi	Giuseppe Santucci
Silvia Chiusano	Vanessa Magionami	Claudio Sartori
Gianni Costa	Giuseppe Manco	Simona Sassatelli
Alessandro Cucchiarelli	Paolo Manghi	Monica Scannapieco
Alfredo Cuzzocrea	Biba Marenglen	Howard Scordio
Claudia d'Amato	Davide Martinenghi	Luca Spalazzi
Massimiliano De Leoni	Riccardo Martoglia	Giorgio Villani
Pasquale De Meo	Elio Masciari	
Franca Debole	Giuseppe M. Mazzeo	

Sponsoring Institutions

Dipartimento di Informatica (Università degli Studi di Bari)
 Università degli Studi di Bari
 Dipartimento di Elettronica, Informatica e Sistemistica (Università della Calabria)
 Istituto di Calcolo e Reti ad Alte Prestazioni (Consiglio Nazionale delle Ricerche)
 Dipartimento ICT (Consiglio Nazionale delle Ricerche)
 Regione Puglia
 Azienda Speciale A.I.C.A.I. (Assistenza Imprese Commerciali Artigiane ed Industriali)
 IBM Italia S.p.A.
 AURIGA INFORMATICA S.r.l.
 ALISEA S.r.l.
 AMT Services S.r.l.
 ARDA Software S.r.l.
 FIMESAN S.p.A.
 Pearson Education

Table of Contents

Tutorial

Geographic and Spatial Data Mining	1
<i>Michael May</i>	

Invited Talks

Smart Earth: From Pervasive Observation over Emergent Understanding to Trusted Information	2
<i>Karl Aberer</i>	

Semantic Web: Schism of the Languages	3
<i>Michael Kifer</i>	

Long Papers

Implementing BRICKS, a Digital Library Management System	4
<i>Nicola Aloia, Cesare Concordia, and Carlo Meghini</i>	

Inducing Multi-Target Model Trees in a Stepwise Fashion	16
<i>Annalisa Appice and Saso Džeroski</i>	

Service Matching and Discovery in P2P Semantic Community	28
<i>Devis Bianchini, Valeria De Antonellis, Michele Melchiori, and Denise Salvi</i>	

Mining Time-series Sequences of Reactions for Biologically Active Patterns in Metabolic Pathways	40
<i>Marenglen Biba, Floriana Esposito, Stefano Ferilli, Nicola Di Mauro, and Teresa M.A. Basile</i>	

A Fast Partial Memory Approach to Incremental Learning through an Advanced Data Storage Framework	52
<i>Marenglen Biba, Stefano Ferilli, Floriana Esposito, Nicola Di Mauro, and Teresa M.A. Basile</i>	

Un Protocollo di Sicurezza contro le Vulnerabilità derivanti da Software di Firma non Trusted	64
<i>Francesco Buccafurri and Gianluca Lax</i>	

KDDBroker: Description and Discovery of KDD Services	76
<i>Jessica Cellini, Claudia Diamantini, and Domenico Potena</i>	

A Hierarchical Probabilistic Model for Co-Clustering High-Dimensional Data	88
<i>Gianni Costa, Francesco Folino, Giuseppe Manco, and Riccardo Ortale</i>	
Translating Context Data in Adaptive Web Applications	100
<i>Roberto De Virgilio and Riccardo Torlone</i>	
Autonomic Deadlock Prevention in Self-Managing Databases	112
<i>Domenico Di Giulio</i>	
Approximate Query Answering and Ranking for Semantic Knowledge Bases	124
<i>Nicola Fanizzi, Claudia d'Amato, and Floriana Esposito</i>	
Uno Strumento Web di Esplorazione Interattiva Multidimensionale dei Dati Geo-Riferiti	136
<i>Sonia Fernandes Silva and Paolo Menicori</i>	
Tecnologie Cognitive nei Sistemi per la Modellistica Geo-spaziale	148
<i>Alberto Gemelli, Claudia Diamantini, and Domenico Potena</i>	
ChronoGeoGraph: an Expressive Spatio-Temporal Conceptual Model	160
<i>Donatella Gubiani and Angelo Montanari</i>	
Accurate and Fast Similarity Detection in Time Series	172
<i>Francesco Gullo, Giovanni Ponti, Andrea Tagarelli, and Sergio Greco</i>	
Finding Generalized Closed Frequent Itemsets for Mining Non Redundant Association Rules	184
<i>Corrado Loglisci, Margherita Berardi, Saverio D'Alessandro, and Pietro Leo</i>	
Musifind: A System for the Automatic Identification of Music Works	196
<i>Nicola Orio</i>	
Trajectory Data Warehouses: Design Issues and Use Cases	208
<i>Salvatore Orlando, Renzo Orsini, Alessandra Raffaetà, Alessandro Roncato, and Claudio Silvestri</i>	
Web Service Discovery at Process-level Based on Semantic Annotation ...	220
<i>Francesco Pagliarecci, Marco Pistore, Luca Spalazzi, and Paolo Traverso</i>	
Extended Abstracts	
Hiding Sequences	233
<i>Osman Abul, Maurizio Atzori, Francesco Bonchi, and Fosca Giannotti</i>	

Query Optimization for Wireless Sensor Network Databases in the MadWise system	242
<i>Giuseppe Amato, Paolo Baronti, and Stefano Chessa</i>	
Query Answering in Expressive Variants of DL-Lite	250
<i>Alessandro Artale, Diego Calvanese, and Roman Kontchakov and Michael Zakharyashev</i>	
Towards an Effective Semi-Automatic Technique for Image Annotation . .	258
<i>Ilaria Bartolini and Paolo Ciaccia</i>	
A New Type of Metadata for Querying Data Integration Systems	266
<i>Sonia Bergamaschi, Francesco Guerra, Mirko Orsini, and Claudio Sartori</i>	
Towards Constraint-Based Subgraph Mining	274
<i>Michele Berlingerio, Francesco Bonchi, and Fosca Giannotti</i>	
Generating Extended Conceptual Schemas from Business Process Models	282
<i>Marco Brambilla, Jordi Cabot, and Sara Comai</i>	
RADAR: Research of Anomalous Data through Association Rules	290
<i>Giulia Bruno, Paolo Garza, Elisa Quintarelli, and Rosalba Rossato</i>	
Modellazione della QoS in Ambienti Web-Service con Applicazioni di Video Streaming	298
<i>Francesco Buccafurri, Luca Console, Pasquale De Meo, Maria Grazia Fugini, Anna Goy, Gianluca Lax, Pasquale Lops, Stefano Modafferi, Barbara Pernici, Claudia Picardi, Domenico Redavid, Giovanni Semeraro, and Domenico Ursino</i>	
Containment of Conjunctive Object Meta-Queries	308
<i>Andrea Cali and Michael Kifer</i>	
Query Optimisation for Web Data Sources: Minimisation of the Number of Accesses	316
<i>Andrea Cali, Davide Martinenghi, and Domenico Carbotta</i>	
Ontology-based Database Access	324
<i>Diego Calvanese, Giuseppe De Giacomo, Domenico Lembo, Maurizio Lenzerini, Antonella Poggi, and Riccardo Rosati</i>	
A Data Mining Methodology for Anomaly Detection in Network Data: Choosing System-Defined Decision Boundaries	332
<i>Costantina Caruso</i>	
Exploiting Peer Ontologies for Semantic Query Propagation	341
<i>Silvana Castano, Alfio Ferrara, and Stefano Montanelli</i>	

Conditional Preferences: A New Semantics for Database Queries	349
<i>Paolo Ciaccia</i>	
XML Type Projection: A Maximum Flow Approach	357
<i>Dario Colazzo and Carlo Sartiani</i>	
Advanced OLAP Visualization of Multidimensional Data Cubes: A Semantics-Driven Compression Approach	365
<i>Alfredo Cuzzocrea, Vincenzo Russo, Domenico Saccà, and Paolo Serafino</i>	
Constraint Hardness for Modelling, Matching and Ranking Semantic Web Services	373
<i>Claudia d'Amato</i>	
Checking e-Service Consistency Using Description Logics	381
<i>Luigi Dragone</i>	
L-SME: A Tool for the Efficient Discovery of Loosely Structured Motifs in Biological Data	389
<i>Fabio Fasseti, Gianluigi Greco, and Giorgio Terracina</i>	
An Information-Theoretic Framework for High-Order Co-Clustering of Heterogeneous Objects	397
<i>Gianluigi Greco, Antonella Guzzo, and Luigi Pontieri</i>	
Exploiting Preference Rules for Querying Databases	405
<i>Sergio Greco, Cristian Molinaro, and Francesco Parisi</i>	
Semantic Web Service Composition in the NeP4B Project: Challenges and Architectural Issues	414
<i>Federica Mandreoli, Wilma Penzo, and Antonio Massimiliano Perdichizzi</i>	
Clustering Web and Desktop Searches	422
<i>Giansalvatore Mecca, Salvatore Raunich, and Alessandro Pappalardo</i>	
Online Distribution Estimation for Streaming Data: Framework and Applications	430
<i>Themis Palpanas, Vana Kalogeraki, and Dimitrios Gunopulos</i>	
On the Schema Exchange Problem	439
<i>Paolo Papotti and Riccardo Torlone</i>	
Beyond Anonymity in Location Based Services	447
<i>Linda Pareschi and Claudio Bettini</i>	
Towards a Definition of an Image Ontology	455
<i>Antonio Penta, Antonio Picariello, and Letizia Tanca</i>	

Extending Datalog for Matchmaking in P2P E-Marketplaces	463
<i>Azzurra Ragone, Umberto Straccia, Tommaso Di Noia, Eugenio Di Sciascio, and Francesco M. Donini</i>	
On Homogeneity Evaluation and Seed Selection in Clustering Relational Data	471
<i>Antonio Varlaro, Annalisa Appice, Antonietta Lanza, and Antonio Fittipaldi</i>	
Demos	
Distributed Information Retrieval and Automatic Identification of Music Works in SAPIR	479
<i>Maristella Agosti, Emanuele Di Buccio, Giorgio Maria Di Nunzio, Nicola Ferro, Massimo Melucci, Riccardo Miotto and N. Orio</i>	
Simultaneous Previews for Time Series Forecasting	483
<i>Paolo Buono</i>	
Analysing Multidimensional Data with a Visualization Tool	487
<i>Paolo Buono and Maria Francesca Costabile</i>	
DB2OWL : A Tool for Automatic Database-to-Ontology Mapping	491
<i>Nadine Cullot, Raji Ghawi, and Kokou Yétongnon</i>	
A Tool for the Visual Synthesis and the Logical Translation of Spatio-Temporal Conceptual Schemas	495
<i>Donatella Gubiani and Angelo Montanari</i>	
Disambiguation of Structure-Based Information in the STRIDER System	499
<i>Federica Mandreoli, Riccardo Martoglia, and Enrico Ronchetti</i>	
Ontology-based Data Access with MASTRO	503
<i>Antonella Poggi and Marco Ruzzi</i>	
Author Index	507

Geographic and Spatial Data Mining

Michael May

Department Knowledge Discovery
Fraunhofer Institute of Autonomous Intelligent Systems
Sankt Augustin, Germany
`michael.may@ais.fraunhofer.de`

Abstract. The widespread use of ubiquitous and mobile technologies such as sensor networks, GPS, mobile phones and RFID, as well as the recent success of Google Earth lead to a situation where more and more data mining applications will have to deal with non-trivial problems of spatio-temporal data analysis. Applications range from telecommunication, retail and market research to scientific applications from ecology or epidemiology.

Despite the importance, standard data mining tools and methods cannot adequately deal with spatial information. Consequently, important information is thrown away, leading to non-optimal results. The last years have seen several lines of research that try to change this situation. Various classes of data mining algorithms - e.g. clustering, association rules, decision trees, subgroup discovery - have been upgraded to handle geographic objects such as lines, points and polygons and their spatial relationships. Nicely complementing classical approaches that have been pioneered in geostatistics (e.g. Kriging, Point Pattern Analysis), those approaches are often rooted in some form of Multi-Relational Data Mining.

In this tutorial, we will first clarify the various data types relevant for geographic data mining and work out the specific characteristics and challenges of geographic data. Next, we discuss several examples of algorithms that take advantage of these data types. Finally, we present a wide range of applications to illustrate the potential, successes and shortcomings of current Spatial Data Mining approaches. We conclude by pointing out some future challenges and directions.

Smart Earth: From Pervasive Observation over Emergent Understanding to Trusted Information

Karl Aberer

Ecole Polytechnique Fédérale de Lausanne (EPFL)
Lausanne, Switzerland
`karl.aberer@epfl.ch`

Abstract. We introduce a model for information management in a world where information from the physical environment is gathered through a ubiquitous wireless networks. Thus the Internet information space and the physical environment become increasingly entangled in a tightly connected system, that we call a Smart Earth. We illustrate technical challenges and applications from the work of the Swiss National Competence Centre of Research in Mobile Information and Communication Systems (NCCR-MICS). We identify the three layers of data access networks, semantic overlay networks and social networks as essential building blocks for networked information management and observe that each of these network layers will be largely self-organized. Finally we argue that understanding the interplay among these self-organizing network layers will be an important research challenge for the future.

Semantic Web: Schism of the Languages

Michael Kifer

Department of Computer Science
State University of New York at Stony Brook, USA
`kifer@cs.stonybrook.edu`

Abstract. Having defined the Web Ontology Language (OWL) in 2004, the W3 Consortium was expecting strong uptake of this technology and hoping that the Semantic Web technology will become part of everyday life - just like the ordinary Web. The uptake, indeed, took place, but the early adopters of the technology quickly realized that OWL is grossly inadequate for their needs. OWL allows them to build ontologies - essentially glorified database schemata - but, as with most database applications, the logic of the applications (such as business rules) remains hidden in procedural programs.

Even before OWL was officially blessed by W3C, many users started to demand that OWL be extended by or combined with logical rules. Unfortunately, this turned out to be easier said than done. First, there are many different rule languages with different logical semantics. Production rules - the most popular type of rules - have no logical semantics at all! Second, all popular rule languages use negation as failure whose various formalizations are based on non-first-order logic. In contrast, OWL is based on classical first-order logic. In the last two years some partial solutions for rules+ontology combinations have emerged, but it is too early to judge their practical impact.

Realizing the importance of rule-based languages for the future of the Semantic Web, W3C has established in late 2005 a Working Group on Rule Interchange Format (RIF). The main goals of RIF are threefold. First, the working group is supposed to design standardized ways for exchanging rule sets with similar and dissimilar semantics through the Web. Second, RIF is expected to provide a practical answer to the question of interfacing rules with ontologies and ensure a degree of standardization for such interfaces. Third, although RIF is intended to be an interchange format, it will inevitably acquire (in fact, already has acquired) the characteristics of a language with formal semantics.

In this talk we will describe the current efforts towards defining a Semantic Web knowledge representation language. We will briefly deal with the foundations of the current standard, the Web Ontology Language (OWL), its limitations, and the difficulties in integrating OWL with a rule language. We will then describe the current status of RIF, its architecture, and the main ideas underlying RIF Core - the core component of the RIF architecture.

Implementing BRICKS, a Digital Library Management System

Nicola Aloia, Cesare Concordia, Carlo Meghini

Institute of Information Science and Technologies (ISTI)
Area della ricerca CNR, via G. Moruzzi 1, 56124 PISA, Italy
{Nicola.Aloia, Cesare.Concordia, Carlo.Meghini}@isti.cnr.it

Abstract. Digital Libraries (DLs) play a central role in the way information is produced, accessed and used in the Internet era. While the Web provides a universal access to uncontrolled information resources, DLs allow large, distributed repositories of multimedia content, annotated with possibly rich semantic structures and regulated by digital rights, to be created and managed to satisfy the needs of vast user communities. This paper describes the main features of BRICKS, an open-source DL management system based on an innovative peer-to-peer architecture and integrating advanced information management techniques, ranging from model-agnostic content and metadata management to distributed query processing.

1 Introduction

Digital Library Management Systems (DLMS) [2,3,7,8] are very complex systems, due to their inherently interdisciplinary nature. A DLMS integrates knowledge, techniques and technology from various disciplines, such as hypertext management, information retrieval, multimedia content management, distributed database management, human computer interaction, to name but a few [1]. Informally, a Digital Library (DL) is a *managed* collection of information, with associated services, where the information is stored in digital format, and accessible over a network [2]. A digital library is not really a “*digitized-library*” [3] nor an automatic library management system, although it must offer at least the services of a traditional library with respect to the way information is stored and accessed. An attempt to define a formal model for DLs is in [1], which defines a Digital Library as consisting of a repository, a set of metadata catalogs for a set of collections in the repository, and a set of services for a society of users. A collection is a set of digital objects. A digital object can be either a simple stream or a complex composition of multimedia objects. The services of a DL must include indexing, searching and browsing of the stored objects. The society of users is the highest level component of the DL, including information needs, use, privacy, ownership, intellectual property right, security, and other user-related concepts.

BRICKS is a DLMS developed in the context of the European Integrated Project: *Building Resources for Integrated Cultural Knowledge Services*, in the Sixth

Framework Program¹, under the IST priority². The aim of the BRICKS project has been to design and develop an open-user and service-oriented infrastructure to share knowledge and resources in the Cultural Heritage domain. Even if the primary goal of the BRICKS project has been to supply infrastructure and suitable services for the Cultural Heritage community, with a special emphasis on small museums, its resulting DLMS can be successfully used in different application domains. BRICKS integrates into a unique, coherent design, a number of relevant enabling technologies, making them available as Web Services through simple but powerful Application Programming Interfaces. This paper presents the main features of BRICKS, focusing on the Collection Manager. The BRICKS Collection Manager³ provides the operations for managing collections and their content, and for querying the information space, making distribution and heterogeneity transparent [11]. The paper is organized as follows. Section 2 gives a brief overview of the relevant literature as well as of comparable DL management systems. The basic BRICKS functionalities are described in Section 3. The BRICKS component-based software architecture, along with a brief description of the main components, is presented in Section 4. Section 5 contains a detailed description of the Collection Manager (CM) component. Conclusions and future works are outlined in Section 6.

2 Related Work

The notion of “*a digital library seems hard to completely understand and evades definitional consensus*” [1]. DLs seat at the cross of many roads. Christine Borgman [6] identifies two distinct senses in which the term “digital library” has been used:

- The technological definition stating that “digital libraries are a set of electronic resources and associated technical capabilities for creating, searching and using information”. In this sense DLs are an extension and enhancement of information storage and retrieval systems that manipulate digital data in any medium (text, images, sounds; static or dynamic images) and exist in distributed networks. This view is contrasted by:
- the social view stating that “digital libraries are constructed, collected and organized, by (and for) a community of users, and their functional capabilities support the information needs and uses of that community”. In this sense DLs are an extension, enhancement, and integration of a variety of information institutions as physical places where resources are selected, collected, organized, preserved, and accessed in support of a user community: libraries, museums, archives, schools, offices, laboratories, homes, public spaces, and so on.

Historically, DLs were launched around the mid 90s by the U. S. Digital Library Initiative in two successive phases, aimed at “dramatically advance the means to

¹ www.brickscmmunity.org

² Contract no. 507457

³ The BRICKS Collection Manager has been designed and implemented at the Institute of Information Science and Technologies of the Italian National Research Council (CNR).

collect, store, and organize information in digital forms, and make it available for searching, retrieval, and processing via communication networks -- all in user-friendly ways". As it is well-known, Google was founded by researchers recruited for the first DL Initiative. The European awareness on DLs started in the 5th Framework Programme and continued in the 6th. In the context of these Programmes, many projects were funded, sided by the scientific activities carried out within the DELOS Network of Excellence on DLs⁴. Significant activity has also been carried out in non-US, non-EU countries, notably New-Zealand⁵.

The notion of Digital Library Management System (DLMS) is still not standardized. Currently, there exist software platforms supporting operational Digital Libraries, each having its own functionality with some overlapping and practically no agreement on a common information model. In fact, the most substantial and systematic attempt at creating a reference model for Digital Libraries is an on-going activity of the DELOS Network of Excellence [10,13,18]. Relevant on-going projects:

- Massachusetts Institute of Technology (MIT) and Hewlett-Packard Labs develop the DSpace⁶ [14,15] digital repository management system. It is used by many organizations around the world to store, preserve and disseminate scientific and educational content.
- The Fedora⁷ [16] Management Service defines an open interface for administering the repository, including creating, modifying, and deleting digital objects, or components within digital objects.
- The DILIGENT⁸ [17] project is creating an advanced test-bed that will allow virtual e-Science communities to share knowledge and collaborate in a secure, coordinated, dynamic and cost-effective way. The DILIGENT test-bed will be built by integrating Grid and Digital Library technologies.

Relating BRICKS to each one of these initiatives cannot be done for reasons of space. In synthesis, BRICKS is currently the only project aiming at providing a full DLMS functionality on a distributed infrastructure, exploiting the peer-to-peer paradigm.

3 BRICKS Functionalities

BRICKS is an extensible Distributed Digital Library Management⁹ built as a set of web-services (BRICKS *foundation services*). The foundation services provide generic support for basic DL services, such as storing, querying and accessing both digital content and metadata; integrating heterogeneous metadata; for security services for

⁴ www.delos.info

⁵ www.nzdl.org

⁶ www.dspace.org

⁷ www.fedora.info

⁸ www.diligentproject.org

⁹ published as open source GNU LGPL (<http://foundation.bricksfactory.org/>)

authentication, authorisation and digital rights management. BRICKS code is platform independent and scalable.

The main goal of BRICKS DLMS is to supply the functionality for helping institutions in (a) setting up their digital libraries and (b) implementing sophisticated applications on top of the global content network, based on a distributed architecture. The complexity of distributed information management (based on a P2P infrastructure in our case) is completely hidden to the application developers. The fundamental BRICKS components provide services for information storage and access, as well as for ontology and metadata schema management. One BRICKS component implements distributed information retrieval capabilities by simple search (Google like), and by advanced search based on knowledge represented by ontology and metadata schemas. Others BRICKS components implement cross-language discovery, possibly personalized based on user profiles. Security management and Intellectual Property Protection, as well as a sophisticated Annotation management component are supplied to users applications without any burden on developers. BRICKS is an extensible system, new components and new functionalities to existing components can be easily added without any change in the already implemented applications.

BRICKS is entirely written in Java and integrates some open source systems for basic functions (e.g. Lucene¹⁰ as a full text information retrieval system). Institutions installing BRICKS software can decide to use proprietary software (e.g. commercial DBMS as a back-end system for BRICKS metadata manager or content manager) when configuring his own environment. Typical digital library tools, are the core applications that can have benefit in using BRICKS, especially those dealing with multimedia documents. Nevertheless cooperative working applications can have great advantages in using BRICKS functionalities.

3.1 The BRICKS Conceptual model

The BRICKS Conceptual Model (see Figure 1) is, at heart, a simple one. A BRICKS DL consists of a set of nodes, called BNodes. A BNode is an abstract notion representing an installation of the system, and it is not specifically related on a single machine: for scalability reasons, a BNode may be deployed on several machines, and for organizational reasons a single machine may host several BNodes, each corresponding to the resources of, e.g. one institution. A Bnode consists of several collections, each of which may be of one of three kinds:

1. Physical Collection, structured in sub-collections
2. Virtual Collection, also structured in sub-collections
3. Stored Query.

These notions will be illustrated in detail in Section 5. DL Items are stored into Physical Collections, Each DL Item has a Content Model which describes it from a structural point of view, that is by describing the part of which it may be composed,

¹⁰ <http://lucene.apache.org/>

the type of these parts and their structure, recursively. It is important to notice that BRICKS does not make any commitment towards a specific content model. Rather, it allows different content models to coexist, each expressed as a model in the Java Content Repository (JCR) API¹¹. In the BRICKS conceptual model, JCR plays the role of a content metamodel. In the course of the BRICKS Project, the Docbook¹² and TEI Lite¹³ models have been expressed in JCR, as well as other application specific models.

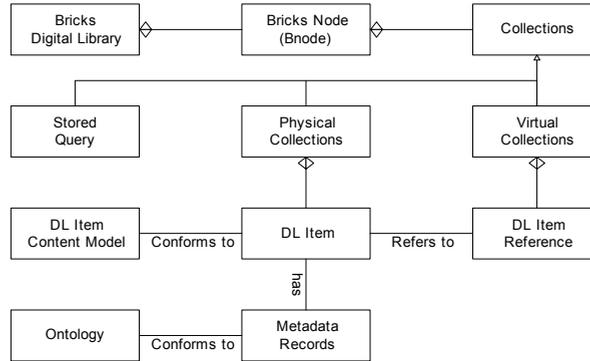


Fig. 1. The BRICKS Conceptual Model

Each DL Item has associated a set of metadata records, which give information about the item required to support several tasks, notably discovery and management. Each metadata record conforms to an ontology, which is expressed in OWL¹⁴. Analogously to content modelling, then, BRICKS does not make any commitment towards a specific metadata model, but rather supports the coexistence of any model, as long as it can be expressed in OWL. For basic interoperability reasons, all DL Items must have an associated Dublin Core metadata record.

The BRICKS Conceptual Model has other entities, to support fundamental DL management operations, such as Digital Right Management, User Management and others. The model illustrated thus far covers however the main aspects and is sufficient to describe the main aspects of the BRICKS architecture, which is done in the sequel.

4 BRICKS architecture

Figure 2 shows the main components of the BNode architecture. The basic layer is provided by the software implementing the network protocols and the P2P layer. The latter implements the P-Grid distributed hash table [4] [5], that is a hash table whose entries are distributed among different peers (we will use terms peer, node and BNode

¹¹ jcp.org/aboutJava/communityprocess/final/jsr170/index.html

¹² www.docbook.org/specs/cs-docbook-docbook-4.2.html

¹³ www.tei-c.org/Lite/

¹⁴ www.w3.org/TR/owl-features/

as synonyms). Each peer is responsible for a partition of the overall key space and maintains information (routing table) to route requests to other peers to be able to forward requests that cannot be answered locally. On top of the basic layer, there are the BRICKS components proper, categorized in 3 main classes:

1. Fundamental bricks are those present in any installation, because they provide necessary services for a BNode to function.
2. Core bricks are those providing a first, elementary level of service, through which a user can be defined and log in (User Management), be recognized (Authentication and Authorization) and perform content discovery or browsing (Collection Manager).
3. Basic bricks are those completing the functionality of the foundations for full-fledged BNodes wishing to upload content protected by digital right management, to associate metadata to content, also as annotations, and creating more advanced services by composing more basic ones.

The P2P paradigm must be understood in this light: by supporting self-maintainability, autonomy and total freedom of joining and leaving the DL. This paradigm is ideal for those institutions, such as small museums, which want to avoid major investments and heavy commitment.

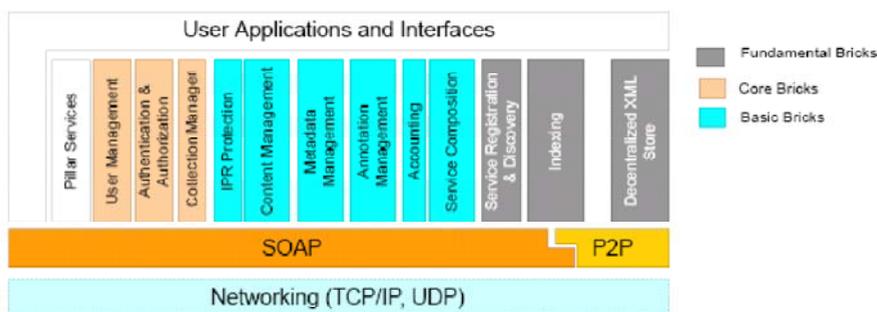


Fig. 2. The BRICKS BNode Architecture

5 Collection Manager

The BRICKS Collection Manager (CM) is a mediator between the DL applications and the DL contents. Applications may range from GUIs to arbitrarily complex programs for performing domain specific tasks. DL contents are the digital objects stored in the DL and their associated information (metadata). The task of the CM is to represent the DL contents via a conceptual model that is understandable and intuitive from an application viewpoint and to offer primitives for the manipulation of this model in support of applications. The CM support two basic kinds of collections: Physical and Logical. Physical collections are a central notion in the BRICKS content model: not only content is organized by physical collection, but so is the discovery of resources and the definition of logical collections. A physical collection is a set of content items (DL objects and metadata) which belong to the same content provider and are homogeneous from the provider point of view (e.g. items are of the same

kind, are described by the same metadata format(s), have same digital rights, and so on). A Logical Collection in BRICKS can be seen as a DBMS view. The items of a Logical Collection can be dynamically computed at run time (if we define the collection as a *Stored Query*) or can be materialized as a set of references to DL items. Any authorized BRICKS user can create and operate upon Logical Collections. Once a logical collection is created, it becomes part of the digital library information space and can be searched by other users as any other digital library resource. A BRICKS content item may be referenced in many Logical Collections (in local and/or remote BNode(s)). Conversely, a Logical Collection may contain references to many items, coming from many different Physical Collections stored in local and/or remote BNode(s). (fig. 3).

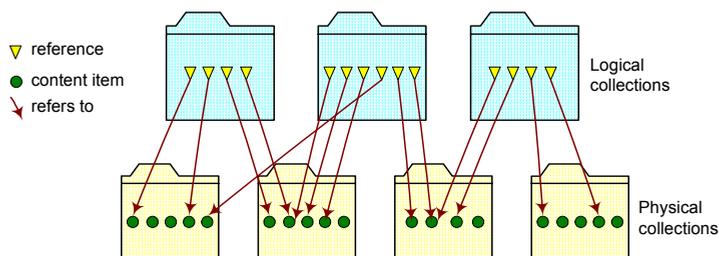


Fig. 3. Physical and logical collections

BRICKS Collections are described by metadata schemas and can be searched like any other DL item. The Application Program Interface (API) of BRICKS Collection Manager is the single access point for all the operations (local and remote) involving the DL contents (objects and metadata). Figure 4 shows the main architecture of the Collection Manager and his relations with the other BRICKS components.

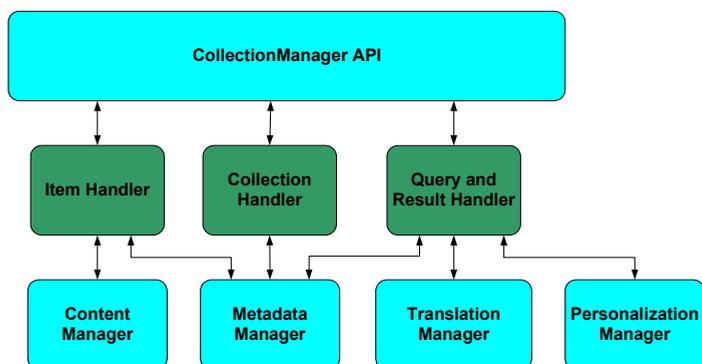


Fig. 4. Collection Manager architecture

The rectangle on the top in figure 4 represents the CM API that can be invoked by users applications via web services. The CM is composed by a set of Java classes (the boxes labelled *Handler* in figure 4) that act as bridges toward the other BRICKS components, hiding the components communications details (e.g. web service

invocation, service composition, etc). The *Collection Handler* class manages collections metadata, it interact with the BRICKS *Metadata Manager*¹⁵ component to create/update/delete a metadata repository¹⁶ for DL items metadata records¹⁷. The *Item Handler* class manages DL items (digital object and metadata), it interact with the BRICKS *Metadata Manager* and *Content Manager* components to insert/remove/update digital objects and metadata, using a transaction like mechanism. The *Query and Result Handler* (described in details in the next section) manages simple and advanced queries on collections and/or items metadata, it interact with the BRICKS *Metadata Manager* to retrieve metadata records, with the BRICKS *Translation Manager* for cross-language query translation and with the BRICKS *Personalization Manager* to return records according to the user preferences.

5.1 Query Handler

The BRICKS Query Handler acts as a sort of metasearch engine [9], retrieving and combining information from multiple sources [12] and identifying useful information objects from returned results. It supports two main kinds of queries: queries searching for a list of terms, in the style of Web search engines, and structured queries where metadata schemas properties can be used to build boolean expressions. As stated in paragraph 3.1, in BRICKS each DL Item has one or more associated metadata records, conform to different schemas. Queries are executed over metadata records describing DL Items¹⁸. The result set of a query consists of all the metadata records of the DL items matching the query filters.

A BRICKS query can have *restrictions*, *conditions* or both, where:

- **Restrictions:** is a list of BNodes or collections identifiers defining the search space, i.e. the set of resources where the query will be executed (e.g. search into *Painter Collection* of ISTI BNode).
- **Conditions:** is a boolean expression defining the filter. Boolean condition may be expressed in terms of some schemas (Advanced Query, Ontology Query) or may be a sequence of terms with wildcards (Simple Query).

BRICKS supports the following kind of queries:

- **Simple Queries:** The language for simple query conditions is that offered by the IR system *Lucene*: it allows to express sequence of unconstrained terms. The BRICKS simple query supports cross-language retrieval. In a cross-language retrieval the condition terms are transformed in order to search for DL Items metadata written in the languages preferred by the user.

¹⁵ based on JENA framework (<http://jena.sourceforge.net/>)

¹⁶ repository is a generic name for a container of metadata records in the Metadata Manager terminology, in this case correspond to the CM notion of Collection.

¹⁷ A Bricks Collection is in turn a DL item so has metadata records describing it.

¹⁸ content based queries on DL object are not currently implemented.

- Advanced Queries.** In these queries, conditions are expressions whose terms have the syntax: “*field_name operator value*” (simple conditions), where operator is one of the usual comparison operator. Each simple condition can refer to a different metadata schema (e.g. `DC.creator="Bob" AND XY.date>01.01.2000`, will return all metadata records of DL items described at least by DC and XY metadata schemas whose fields values match the filter). Note that conditions in advanced queries can also refer to the Collections metadata schema, this means either that collections can be searched and that collections metadata fields can be used as filters for finding DL Items. Advanced queries are translated into SPARQL¹⁹ expressions and executed by the underlying Jena RDF query processor.

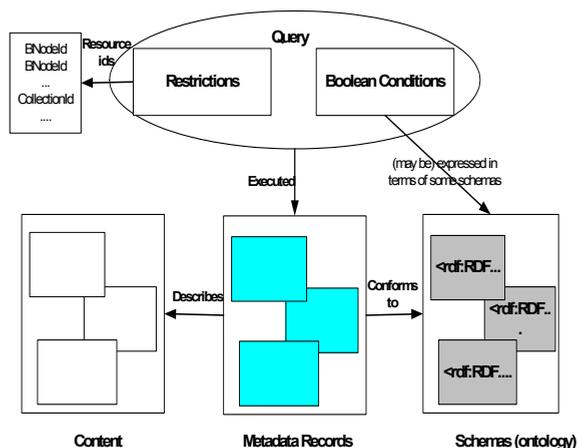


Fig. 5. Evaluating a BRICKS query

In order to make the current BRICKS implementation independent from any specific platform for query evaluation, the query service is made available through an object-based interface.

5.2 Query Processing

The goal of the BRICKS query processing is to define the search space for the query and to build a set of access plan. One of the most important processing step is the query optimisation. In the following we will describe the *source selection*, used by the query optimizer to limit the data transfer. The source selection algorithm is based on the routing indices, described in the next section.

5.2.1 Routing Indices

There are two indices that contain information about BNodes at different granularities: collections and property values.

¹⁹ <http://www.w3.org/TR/rdf-sparql-query/>

- *property value pairs index* contains attribute value pairs of DL items metadata records stored in the BRICKS data space. This is used to evaluate the query conditions.
- *collection index* containing metadata of collections stored in BRICKS. This is similar to the catalog in Relational DBMS and is used to evaluate the query restrictions.

Due to the different type, amount and use of information stored in them, we have chosen two different ways to distribute indices. Property value index is stored in a Distributed Hash Table (DHT). For each property value pair a special key is computed and each BRICKS peer uses a routing table to find whom to send a message to store or retrieve a given key. The Distributed XMLStorage component is used to distribute the collection index. Metadata are stored as XML files and each BNode uses XPATH queries to retrieve a set of collection based on metadata fields conditions.

5.2.2 Source Selection

A source selection algorithm is used by the query optimizer, in order to limit the space to be searched. The source selection does the following:

1. the Boolean Condition part of the query is transformed in an equivalent DNF expression. The transformation process may require an exponential number of steps, but the number of simple conditions is not expected to be high. The *property value index* is then used to find, for every conjunctive term, the set of collections having metadata records matching it. The join of all obtained collection is the *preliminary candidate* collection space for the query.
2. if the query restriction part is not empty, the intersection of the user supplied collection space (could be a superset of the obtained one) and the one obtained in the previous phase is the *candidate* collection space for the query.
3. if the query restriction part is empty the *preliminary candidate* collection space is equal to the *candidate* collection space for the query.
4. the *collection index* is then used to restrict the *candidate* collection space to the effective collection space, by removing empty collections and collections belonging to BNodes not currently connected to the BRICKS network.
5. the access plan is then built by grouping collections by BNode.

5.3 Query execution

Basically, the access plan defines the set of Metadata Manager services that must perform the query and the order in which they will be called. To improve efficiency we have adopted an asynchronous algorithm for query execution: when a query is executed by a client application, a token identifying the query is returned. The identifier will be used by the client application to fetch metadata records in a RBMS cursor like way. Figure 6 shows a simplified sequence diagram describing query execution in BRICKS. After returning the query identifier, the BRICKS search component queries in parallel all the Metadata Managers in the access plan using a pool of threads. Every metadata manager executes the query, using local indexes, and

returns the result. Results are collected by the coordinator and stored in a cache area memory for the client application.

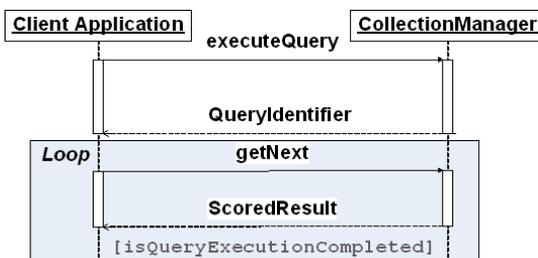


Fig. 6. Application-Collection Manager Interaction

6 Conclusions

The main features of the BRICKS DLMS have been illustrated, focusing on the Collection Manager, which is the central component of the system. BRICKS DLMS has been used to develop a certain number of applications in the area of Cultural Heritage by our project partners. A set of BNodes has been installed and are currently used by BRICKS partners (including small museums and libraries). People interested in BRICKS DLMS software can download source and/or binary code at <http://foundation.bricksfactory.org/>. Future works includes query optimization improvement and content based search (similarity search).

7 Acknowledgements

The BRICKS Project is partially funded by the European Commission as an Integrated Project of the 6th Framework Programme, within the framework of the specific research and technological development programme "Integrating and Strengthening the European Research Area (2002-2006)", under contract number 507457. Thanks are due to our colleagues Francesco Furfari, Maria Teresa Paratore, Federico Tonioni and Loredana Versienti who significantly contributed to the design and implementation of the Collection Manager component.

References

1. M.A. Gonçalves, E.A. Fox, L.T. Watson and N.A. Kipp. Stream, Structures, Spaces, Scenarios, Societies (5S): A Formal Model for Digital Library. *ACM TOIS*, 22(2): 270-312, 2004.
2. W.Y. Arms. *Digital Libraries*. The MIT Press, 2000.
3. I.H. Witten and D. Bainbridge. *How to Build a Digital Library*. The Morgan Kaufmann Series in Multimedia and Information Systems, 2003.

4. K. Aberer. P-Grid: A self-organizing access structure for P2P information systems. *Proceedings of the Sixth International Conference on Cooperative Information Systems (CoopIS)*, 2001.
5. K. Aberer, M. Hauswirth, M. Puceva and R. Schmidt. Improving Data Access in P2P Systems. *IEEE Internet Computing*, 6(1): 58-67, 2002.
6. C. L. Borgman. *From Gutenberg to the Global Information Infrastructure*. Cambridge, MA: The MIT Press, 2000.
7. M. Lesk. *Understanding Digital Libraries* (Second ed.). San Francisco, CA: Morgan Kaufman Publishers, 2004.
8. G.G. Chowdhury and S. Chowdhury. *Introduction to Digital Libraries*. London: Facet, 2003.
9. W. Meng, C. Yu and K.-L. Liu (2002): Building Efficient and Effective Metasearch Engines, *ACM Computing Surveys*, vol. 34, no. 1, pp. 48-89.
10. L. Candela, D. Castelli, P. Pagano, C. Thanos, Y. Ioannidis, G. Koutrika, S. Ross, H.-J. Schek and H. Schuldt. Setting the Foundations of Digital Libraries. *The DELOS Manifesto. D-Lib Magazine*, 13(3/4), March/April 2007.
11. F. Pentaris and Y. Ioannidis: Query optimization in distributed networks of autonomous database systems. *ACM Transactions on Database Systems*, 31(2): 537-583, June 2006.
12. A.P. Sheth and J.A. Larson: Federated database systems for managing distributed, heterogeneous, and autonomous databases. *ACM Comput. Surv.*, 22(3):183-236, 1990.
13. D. Castelli and P. Pagano: A System for Building Expandable Digital Libraries. *Proceedings of the 3rd ACM/IEEE-CS JCDL '03*, 2003.
14. R. Tansley, M. Bass, D. Stuve, M. Branschofsky, D. Chudnov, G. McClellan and M. Smith. The DSpace institutional digital repository system: current functionality. In *Proceedings of the 3rd ACM/IEEE-CS Joint Conference on Digital Libraries* (Houston, Texas, May 27 - 31, 2003). International Conference on Digital Libraries. IEEE Computer Society, Washington, DC, 87-97, 2003.
15. M. Smith, M. Barton, M. Bass, M. Branschofsky, G. MacClellan, D. Stuve, R. Tansley and J.H. Walker. An Open Source Dynamic Digital Repository. *D-Lib Magazine* 9(1), January 2003.
16. T. Staples and R. Wayland. Virginia Dons FEDORA: A Prototype for a Digital Object Repository. *Dlib Magazine*, 6(7/8), July/August 2000.
17. L. Candela, M. Simi, P. Pagano and D. Castelli. DILIGENT: A DL Infrastructure for Supporting Joint Research. In *Proc. 2nd IEEE-CS International Symposium Global Data Interoperability- Challenges and Technologies*, pp. 56-59, 2005.
18. H.-J. Schek and H. Schuldt: DelosDLMS: global prototype development. *DELOS Research Activities 2006* C. Thanos (editor) pp. 19-20, 2006.

Inducing Multi-Target Model Trees in a Stepwise Fashion

Annalisa Appice¹ and Saso Džeroski²

¹Dipartimento di Informatica, Università degli Studi di Bari
via Orabona, 4 - 70126 Bari - Italy

²Department of Knowledge Technologies, Jožef Stefan Institute
Jamova 39 - Ljubljana, Slovenia 1000
appice@di.uniba.it, Saso.Dzeroski@ijs.si

Abstract. Model trees are tree-based models that associate leaves with multiple linear models and are used to solve prediction problems in which the target variable is continuous. In this paper, we address the task of inducing multi-target model trees, which predict the values of several target variables simultaneously. Each leaf contains several linear models, each predicting the value of a different target variable. We propose an algorithm for inducing such trees in a stepwise fashion. At each step of tree construction, we choose either to partition the current training set (split node) or to introduce a regression variable in the linear models, one for each target variable, to be associated with the leaves (regression node). Experiments show that multi-target model trees are much smaller than the corresponding sets of single-target model trees, while achieving comparable accuracies, and in addition are induced much faster.

1 Introduction

In the classical regression setting, data is described by a fixed set of explanatory (independent) variables $\mathbf{X} = X_1, X_2, \dots, X_m$ corresponding to the (discrete and continuous) properties of the units of analysis. Each unit of analysis is labeled according to an unknown function g whose range is a set of real numbers \mathbb{R} . This representation is that adopted in statistics, which makes it possible to devise efficient algorithms by taking a training sample $S = \{(\mathbf{x}_i, y_i) \in \mathbf{X} \times Y \mid y_i = g(\mathbf{x}_i) \wedge i = 1 \dots N\}$ as input and returning a function f which is hopefully close to g on domain \mathbf{X} . Learning algorithms that induce a model tree approximate the function g by a piecewise (linear) function, since they associate leaves with multiple linear models. RETIS [7], M5' [15], RT [12], MAUVE [13] and SMOTI [8] are only some examples of model tree induction systems.

Although being successful regression techniques, traditional regression and model trees are not capable of predicting several target variables, Y_1, \dots, Y_n , simultaneously. The trivial solution to the *multi-target regression* task is to build a collection of separate models, one for each target variable Y_j , but a turning perspective comes from the *predictive clustering* framework [1], where now exist

methods to construct clusters of examples that are similar to each other and simultaneously associate a predictive model with each constructed cluster.

Predictive clustering framework has been applied to both classification and regression trees [1][11] and rules [14], but there is no attempt of building a model tree to predict the values of several continuous target variables simultaneously.

In this paper, we propose a *Multi Target Stepwise Model Tree Induction* (MTSMOTI) algorithm for inducing such trees in a stepwise fashion. The model tree is top-down induced by choosing at each step to either partition the training space (split nodes) or introduce a regression variable in the set of linear models to be associated with leaves (regression nodes). Internal regression nodes contribute to define multi-target multiple models and capture *global* effects, while straight-line regressions with leaves capture only *local* effects.

The stepwise construction of a model tree is described in [8] for the single-target case. We propose a novel implementation of SMOTI that preserves all nice characteristics of a tree structure with split nodes and regression nodes reducing the time complexity without affecting accuracy. Similarly, to SMOTI, the multiple linear function associated with each leaf involves all the continuous variables in the regression nodes along the path from the root the leaf. Only a subset of continuous variables may be involved in multiple linear models associated with the leaves, thus solving problems due to collinearity. The validity of either a regression step or a split test is based on the heuristic evaluation measure from the MAUVE system that is adapted for evaluating regression nodes in addition to the split ones. The new heuristic function reduces SMOTI time complexity of evaluating a node yielding trees with better accuracy power. The post-pruning procedure makes use of an estimate of the expected error that will be experienced at each node for the test data. Pruning allows the tree size to be reduced with additional improvement in accuracy power. Finally, in the multi-target context, MTSMOTI builds a single multi-target model tree that is much smaller than the total size of the individual trees, although it preserves accuracy in prediction. This single tree is induced much faster than the set of individual trees.

The paper is organized as follows. The stepwise induction of multi-target model trees is presented in Section 2. Experimental results are reported in Section 3 and some conclusions are drawn.

2 The algorithm

Our system for the induction of multi-target model trees employs the basic stepwise construction of a regression model as it is incorporated in SMOTI [8]. To explain the stepwise mechanism, let us consider a simple example: suppose we are interested in analyzing a target variable Y in a region R of a feature space described by two continuous explanatory variables X_1 and X_2 when R can be partitioned into two regions R_1 and R_2 and two linear regression models involving both X_1 and X_2 can be built independently for each region R_i . It may be found that the Y value is proportional to X_1 and this behavior is independent of any partitioning of R . The effect of X_1 on Y is *global*, since it can be reli-

ably predicted on the whole region R . The initial regression model can be then approximated by regressing on X_1 for the whole region $R : \hat{Y} = \hat{a}_0 + \hat{b}_0 X_1$. The effect of another variable in the partially constructed regression model is introduced by eliminating the effect of X_1 : we have to compute the regression model for the whole region R , that is, $\hat{X}_2 = \hat{a}_{20} + \hat{b}_{21} X_1$ and then to compute the residuals $X'_2 = X_2 - \hat{X}_2$ and $Y' = Y - \hat{Y} = Y - (\hat{a}_0 + \hat{b}_0 X_1)$. The subsequently partitioning of R into two regions R_1 and R_2 may lead to build two independent regression models and capture the effect of the variables X_2 *locally* to the subregions R_1 and R_2 , respectively. Obviously, a straight-line regression now involves residual variables of Y' and X'_2 and it is automatically translated into a multiple linear function involving Y and X_1, X_2 . Practically, this corresponds to a tree structure with split nodes to binary partition training data and regression nodes to perform straight-line regressions (see Figure 1). For a regression node, both target variables and continuous explanatory variables not included in the model are transformed to remove the linear effect of those variables already included.

$$\begin{aligned}
 &| \text{--REGRESSION NODE: } \hat{Y} = \hat{a}_0 + \hat{b}_0 X_1 (R) \\
 &\quad \hat{X}_2 = \hat{a}_{20} + \hat{b}_{21} X_1 \\
 &| \text{--SPLIT NODE: } X_1 \leq \alpha (R_1) \\
 &| \text{---LEAF NODE: } Y - (\hat{a}_0 + \hat{b}_0 X_1) = \hat{a}_1 + \hat{b}_1 (X_2 - (\hat{a}_{20} + \hat{b}_{21} X_1)) \\
 &| \text{--SPLIT NODE: } X_1 > \alpha (R_2) \\
 &| \text{---LEAF NODE: } Y - (\hat{a}_0 + \hat{b}_0 X_1) = \hat{a}_2 + \hat{b}_2 (X_2 - (\hat{a}_{20} + \hat{b}_{21} X_1))
 \end{aligned}$$

Fig. 1. A model tree with split and regression nodes. The root performs a regression step on X_1 . The straight-line regressions associated with the leaves can be translated into multiple linear functions on $X_1 \times X_2 \rightarrow Y$, that is, $Y = (\hat{b}_0 + \hat{b}_{21} \hat{b}_2) X_1 + \hat{b}_2 X_2 + \hat{a}_0 + \hat{a}_2 + \hat{b}_2 \hat{a}_{20}$ on R_1 and $Y = (\hat{b}_0 + \hat{b}_{21} \hat{b}_3) X_1 + \hat{b}_3 X_2 + \hat{a}_0 + \hat{a}_2 + \hat{b}_3 \hat{a}_{20}$ on R_2 .

The main point where MTSMOTI differs from SMOTI is in predicting several target variables simultaneously by assuming some dependence among such variables. Furthermore, MTSMOTI differs from SMOTI in that it uses other heuristic for evaluating both split and regression nodes such decreasing time complexity of node selection step without significantly affecting accuracy of tree, stopping criteria and a post-pruning method. We discuss these topics below.

2.1 Model Tree Construction

The development of the tree structure is determined by a recursive partitioning procedure (splitting) and some intermediate prediction functions (regression) which take into account multi-target nature of data. The validity of either a split test or a regression step t on a variable X_i is based on two distinct evaluation measures, $s(t, \mathbf{Y})$ and $r(t, \mathbf{Y})$, respectively. The variable X_i is of a continuous type in the regression case, and of any type in the split case.

A split node t on a variable X_i performs a binary test. If X_i is continuous and α is a threshold value for X_i the split test is in the form $X_i \leq \alpha$ vs $X_i > \alpha$.

Possible values of α are found by sorting the distinct values of X_i in the training sample associated with t , then identifying one threshold in each distinct value. If X_i is discrete, a discrete split partitions attribute values into two complementary sets, so that a binary tree is always built. To determine discrete threshold, we use the same criterion applied in CART. If k is the number of distinct values for X_i and $S_{X_i} = \{x_{i_1}, \dots, x_{i_k}\}$ is the set of distinct values of X_i in t , S_{X_i} is sorted according to the sample mean of a target variable Y (or residual of Y) over all training cases falling in t , that is, $\bar{Y}_1, \dots, \bar{Y}_k$. A theorem by Breiman et al. [2] (Theorem 4.5, Proposition 8.16) proves that the best binary split is one of $k - 1$ partitions $\{x_{i_1}, \dots, x_{i_h}\}$ and $S_{X_i} - \{x_{i_1}, \dots, x_{i_h}\}$ ($h < k$), thus greatly reducing the search for the best subset of categories from 2^{k-1} to $k-1$ partitions. The validity of this result is based on the assumption that the linear relation between a continuous variable Y (or its residual) and a discrete one X_i can be approximated by the sample mean on Y (or residual of Y). In the multi-target case, \mathbf{Y} , the set of distinct values of X_i is sorted according to the ‘‘average’’ of the sample means for each target variable Y_j . This brings up the problem of combining variables whose range may differ in several orders of magnitude. To overcome this problem, for each target variable Y_j , the sample means are scaled within the range $[0, 1]$: the lowest (highest) value is assigned a real value of 0(1). We denote by \bar{Y}_{j_s} the sample mean of Y_j on all training cases in t having $X_i = x_{i_s}$. The scaled value of \bar{Y}_{j_s} is obtained as follows:

$$\bar{Y}_{j_s \rightarrow [0,1]} = \frac{|\bar{Y}_{j_s} - \min_j|}{(\max_j - \min_j)} \quad (1)$$

where $\min_j = \min_{s=1, \dots, k} \{\bar{Y}_{j_s}\}$ and $\max_j = \max_{s=1, \dots, k} \{\bar{Y}_{j_s}\}$.

A regression node on X_i performs a set of straight-line regressions on X_i , one for each target variable Y_j ($Y_j = \alpha_j + \beta_j X_i$). The slope (α_j) and intercept (β_j) are computed by least square regression [5]. Straight-line regressions in the subtree rooted in a regression node will involve residuals of both the target variables and the continuous explanatory variables not yet included in the model.

2.2 Split and Regression Node Evaluation

Details on the evaluation of either a split or regression node in MTSMOTI are reported below.

If t is a split on X_i then $s_j(t, Y_j)$ is computed as follows:

$$s_j(t; Y_j) = \frac{N(t_L)}{N(t)} RE(t_L; Y_j) + \frac{N(t_R)}{N(t)} RE(t_R; Y_j), \quad (2)$$

where $N(t)$ is the number of cases reaching t , $N(t_L)$ ($N(t_R)$) is the number of cases passed down to the left (right) child, and $RE(t_L)$ ($RE(t_R)$) is the resubstitution error of the left (right) child. Resubstitution error is computed as:

$$RE(t; Y_j) = \sqrt{\frac{1}{N(t)} \sum_{i=1}^{N(t)} (y_{j_i} - \hat{y}_{j_i})^2} \quad (3)$$

For the left(right) child of a split t , the estimate $\hat{y}_j = \alpha_0 + \sum \beta_{j_s} x_s$ is combines the straight-line regressions associated with regression nodes along the path from the root to t_L (t_R) with the straight-line regression on X_i computed on t_L (t_R). In the case X_i is a discrete variable, straight-line regression on t_L (t_R) is replaced with the sample mean of Y_j (or residual of Y_j) values falling in t_L (t_R).

This evaluation function for split node is derived by MAUVE [13] as an alternative to consider no regression (M5'), simple regression on all continuous variables (SMOTI) or multiple regression on all continuous variables together (RETIS). The motivation in favor of this evaluation measure is in the time computational complexity. MAUVE, similarly to M5', is linear in the number of variables, while SMOTI is quadratic and RETIS is cubic. At same time, as proved in [13], MAUVE split evaluation avoids some pathological behavior of M5' that may reduce the explanatory power of the induced tree. An additional motivation is that, in our stepwise construction, a split on X_i is assumed to be an alternative to perform a regression step on X_i . This alternative is valuable when the contribution of X_i can be reliably estimated only on separate sub-sets of the sample data falling in t . In this case the intuition is that sub-sets can be heuristically searched within the set of candidate binary splits involving only X_i without significantly affecting accuracy of trees.

The evaluation of a regression step $Y_j = \alpha_j + \beta_j X_i$ at node t is based on the resubstitution error $RE(t; Y_j)$. In this way, the selection of the best regression step requires the computation of a straight-line regression, whose complexity is linear in N , for each of the m variables.

In the multi-target framework, measures obtained at t for separate target variables are combined as follows:

$$s(t, \mathbf{Y}) = \frac{1}{n} \sum_{j=1}^n s_{\rightarrow[0,1]}(t; Y_j) \quad \left(r(t, \mathbf{Y}) = \frac{1}{n} \sum_{j=1}^n RE_{\rightarrow[0,1]}(t; Y_j) \right) \quad (4)$$

For each target variable Y_j , values of $s_j(t, Y_j)$ and $RE(t, Y_j)$ are scaled within the range (0,1). In this case, \min_j (\max_j) is the minimum (maximum) resubstitution error on the set including both split and regression candidates ($\{s_j(t, Y_j)\} \cup \{RE(t, Y_j)\}$). The most promising split minimizes the evaluation measure s on the set of split candidates. Similarly, the most promising regression step minimizes r on the set of regression candidates. Since both s and r are defined as resubstitution errors, they can be actually compared to choose between growing the tree by adding either a split node or a regression one.

As pointed in [8], a regression step on X_i would result in values of $r(t; Y_j)$ less than or equal to values of $s(t; \mathbf{Y})$ for some split test involving X_i . In SMOTI, this is solved by growing the tree at a further level in order to base the computation of $r(t; Y_j)$ on the best multiple linear regressions after the regression step on X_i is performed: for each straight-line regression a look-ahead best split is computed. To preserve the linear time complexity of the regression step evaluation and to be able to identify explanatory variables having global effect, the split selection criterion in MTSMOTI is improved to consider the special case of identical regression models associated with both children (left and right). When

this occurs, the set of straight-line regressions associated with the best split t is the same as that associated with both t_L and t_R , up to some statistically insignificant difference. In other words, the split is useless and it can be replaced with corresponding regression candidate. To check this special case, MTSMOTI compares each pair of regression lines associated with the children according to a statistical test for coincident regression lines [16] with $O(n)$ time complexity.

2.3 Stopping Criteria

Three different stopping criteria are implemented. The first uses the partial F-test to evaluate the actual contribution provided by a new explanatory variable to the model [5]. The F-test is performed separately for each target variable. Hence, stopping can operate at different tree depth for separate target variables, while the stepwise construction of the regression model proceeds only for those target variables whose model can be significantly improved by adding new variables. The second requires the number of examples in each node to be greater than a minimum value. The third stops the induction process when all continuous explanatory variables along the path from the root to the current node are used in regression steps and there are no discrete variables in the training set.

2.4 Pruning

Similarly to other TDIMT approaches, MTSMOTI may generate model trees that overfit training data. Almost all TDIMT systems use some simplifying techniques to determine which nodes of the tree should be taken as leaves. These techniques are generally derived from those developed for decision trees. RETIS bases its pruning algorithm on Niblett and Bratko's method [10], extended later by Cestnik and Bratko [4]. M5' uses a pessimistic-error-pruning-like strategy since it compares the error estimates obtained by pruning a node or not. The error estimates are based on the training cases and corrected in order to take into account the complexity of the model in the node. SMOTI can optionally adopt simplification techniques that use a pruning set that is independent of the set of observations used to build the tree. Therefore, the training set is partitioned into a growing set used to build the tree and a pruning set used to simplify T. Although pruning, is generally beneficial, the comparison with M5', which uses the training data both for growing and for pruning the tree, has shown that putting aside some data for pruning can lead to worse results [3].

MTSMOTI, similarly to M5', adopts a pruning procedure that computes the set of linear models for each interior node of the un-pruned tree. Linear models built in a stepwise fashion at each node are expanded by sequentially adding variables, one at a time based on the strength of the resulting average resubstitution error. The models are built by using only the continuous variables tested or regressed in the subtree below this node. For each target variable, the contribution of an added term is immediately evaluated according to the F-test and eventually dropped whenever it is not statistically significant. Finally, once

a linear model is in place for an interior node, the tree is pruned back from the leaves so long as the expected estimated error decreases.

The estimate of expected error is the average of the resubstitution errors (scaled in the range $[0,1]$) on the training cases reaching that node for each target variable. To compensate the fact that this average may underestimate the expected error on unseen cases, it is multiplied by the factor $(N(t)-\nu(t))/(N(t)+\nu(t))$, where $N(t)$ is the number of training cases that reach t and $\nu(t)$ is the number of parameters (variables) in the linear model associated with the node.

Table 1. Single-target datasets used in the empirical evaluation of MTSMOTI.

D	Dataset	#Cases	%Miss.	#Con.	#Disc.	D	Dataset	#Cases	%Miss.	#Con.	#Disc.
1	AutoHorse	203	1.12	17	8	7	Housing	506	0	12	1
2	AutoMpg	398	0.22	4	3	8	Quake	2178	0	3	0
3	AutoPrice	159	0	15	0	9	Sensory	575	0	0	11
4	Tumor	286	0.35	1	8	10	Servo	167	0	0	4
5	Cloud	108	0	4	2	11	Strike	625	0	5	1
6	CPU	209	0	6	1	12	Veteran	137	0	3	4

3 Experimental results

MTSMOTI is empirically evaluated on both single-target and multi-target regression datasets. Each dataset is analyzed by means of a 10-fold cross-validation. For each target variable Y_j , performances are evaluated on the basis of the average relative root mean square error (RRMSE) on 10 hold-out folds, that is:

$$RRMSE(D, Y_j) = \frac{1}{k} \sum_{i=1}^k \left(\sqrt{\sum_{h=1}^{N(D_i)} (y_{jh} - \hat{y}_{jh}(D/D_i))^2} / \sqrt{\sum_{h=1}^{N(D_i)} (y_{jh} - \bar{y}_j(D_i))^2} \right) \quad (5)$$

where $D = \{D_1, \dots, D_k\}$ is a cross-validation partition, D_i is a set of indices of testing dataset, k is the number of folds (i.e., 10), $N(D_i)$ is the number of cases stored in D_i , $\hat{y}_{jh}(D/D_i)$ is the value predicted for the j -th target variable of the h -th testing case by the model tree induced on D/D_i and \bar{y}_j is the mean value of y_j on D_i . RRMSE is averaged on separate target variables. The complexity of trees is evaluated on the basis of the average number of leaves. MT-SMOTI is run by requiring that number of examples in each node to be greater than a 10% of dataset size. F-Value is set to 0.001.

3.1 Single-Target Datasets

MTSMOTI is initially tested on twelve single-target regression datasets (see Table 1) taken from the UCI Machine Learning (<http://www.ics.uci>). They have

a single continuous variable to be predicted and they have been used as benchmarks in related studies on regression trees and model trees. In this empirical study, data cases with missing values are ignored.

MTSMOTI is run in two settings to test significance of the stepwise model tree induction. In the former setting (SR), we evaluate model trees built in a stepwise fashion (with post-pruning or no), while in the latter setting (S), we evaluate model trees built by only partitioning training sample and then associating multiple linear models at leaves by post pruning the tree. MTSMOTI is compared with REGTREE, that is our implementation of a regression tree learner, SMOTI and M5' as well as predictive clustering trees (PCT) and rules (PCR Ordered and PCR UnOrdered)¹. SMOTI and M5' are run by using default stopping thresholds. Differently from the empirical study reported in [8], the pruning of M5' is now enabled, but no smoothing factor is used. Results on the accuracy and tree size are reported in Table 2.

Table 2. Single-target regression: comparison of the average RRMSE and average size of MTSMOTI (SR and S), REGTREE, SMOTI, M5', PC-Tree (PCT), PCR Ordered and PCR Unordered.

D	RRMSE									Size								
	MT (SR) No Pr.	MT (SR) Pr.	MT (S)	REG TR.	SM O TI	M5	PC T	PC R Ord.	PC R UnOrd.	MT (SR) No Pr.	MT (SR) Pr.	MT (S)	REG TR.	SM O TI	M5	PC T	PC R Ord.	PC R UnOrd.
1	0.44	0.38	0.43	0.47	0.49	0.35	0.41	0.68	0.72	7.7	3.8	4.2	6.8	6.6	2.4	23	4	1
2	0.41	0.40	0.44	0.44	0.46	0.37	0.45	0.7	0.69	14.6	12.7	10.5	16.5	10	4.8	16	4	1
3	0.46	0.48	0.46	0.60	0.53	0.40	0.49	0.52	0.63	7.5	7.2	4.5	10.2	6.6	7.4	9	3	1
4	1.02	1.00	1.16	1.16	1.34	0.99	0.96	1.12	1	5.5	4.7	17	16.6	8	1.1	3	17	1
5	0.54	0.53	0.68	0.82	0.70	0.52	0.58	0.72	0.72	3.6	3.2	4.2	6.9	5	2.4	9	3	1
6	0.17	0.17	0.34	0.65	0.80	0.20	0.33	0.52	0.74	10.6	8.3	3.9	13	7	3.1	12	3	1
7	0.44	0.45	0.47	0.49	0.45	0.44	0.43	0.67	0.87	16.8	4.5	18.5	21.1	11	13.5	32	7	1
8	1.01	0.99	1.12	1.03	2.38	1.01	0.99	1.16	1	2.4	2.3	29	28.8	23	3.3	3	129	1
9	0.96	0.93	0.93	0.93	1.00	0.96	0.94	1.07	1	16.6	16.5	16.5	16.5	12	4.7	7	34	1
10	0.65	0.60	0.60	0.60	0.45	0.43	0.42	0.61	0.47	9.5	7	7	7	7	5.6	11	5	3
11	0.94	0.94	0.91	0.96	1.88	1.24	0.98	1.26	0.98	6.6	3.9	18.7	18.7	12	6.5	12	14	1
12	1.22	1.20	1.15	1.34	2.40	1.22	0.99	1.22	1.06	3.6	3.2	6.1	7.8	6	1	2	8	4
Avg	0.68	0.67	0.72	0.79	1.07	0.67	0.66	0.85	0.82	8.75	6.44	11.6	14.1	9.51	4.65	11.5	19.2	1.41

Several conclusions are drawn from these experimental results. First, model trees outperform regression trees in terms of accuracy and tree size. Second, our implementation of the stepwise tree construction clearly improves performance of SMOTI in terms of accuracy. The exception is the “servo” data collection that includes only discrete variables. Difference in accuracy probably depends on the

¹ Results for PC-Tree, PCR Ordered and PCR Unordered on single-target datasets are provided by Bernard Ženko [14].

different criterion adopted to determine discrete threshold. MTSMOTI uses the same criterion adopted by CART, while SMOTI relies on the greedy strategy as suggested in [9]. Third, post-pruning is in general conservative. It increases accuracy of trees by avoiding overfitting training data and it reduces the tree size. Order of size reduction depends on dataset. Fourth, the comparison between trees built in a stepwise fashion (SR) and trees built by firstly partitioning training data and then associating leaves with (multiple) linear models (S) is in favor of the stepwise procedure. Trees with split and regression nodes achieve better (or at worst comparable) accuracy than trees with only split nodes. At the same time, they allow to capture presence of global effects of variables, which are completely ignored otherwise. No general conclusion can be drawn on the tree size. Fifth, the comparison with M5' accuracy shows that MTSMOTI is sometime better, at worst comparable, to M5', but M5' typically builds smaller trees. Hence, MTSMOTI is able to detect the presence of global effects without significantly affecting accuracy, thus preserving the main desirable characteristic of stepwise construction, that is, the easy interpretability of the induced trees. Finally, the comparison with predictive clustering methods shows that MTSMOTI generally outperforms rule learners, while it does not exhibit an irrefutable superiority with respect to PC-TREE, although results are still good.

3.2 Multi-Target Datasets

Multi-target regression is a relatively new data mining task, and there are few publicly available data sets. There is only solar-flare in the UCI Machine Learning Repository that can be regarded as a multi-target regression data set. All the other data sets used in this study are not publicly available. A brief description of all datasets is reported below.

EDM dataset [14] describes 154 actions taken by a human operator controlling two variables (target variables), namely the flow of the dielectric fluid and the gap between electrodes in the electrical discharge machining process. For each target variable, three actions are possible: human operator increased the variable (value 1), decreased the variable (value -1), or took no action (value 0). The reasons for each action are described in terms of 16 continuous variables. MICROARTHROPODS dataset [14] collects 1944 measurements of soil quality from 142 continuous variables describing agricultural measures and events. Soil quality is described in terms of mites and springtails species as well as Shannon biodiversity (target variables). SIGMEA-REAL dataset [14] collects 817 measurements of the rate of herbicide resistance of two lines of plants (target variables), that is, the transgenic male-fertile (MF) and the non-transgenic male-sterile (MS) line of oilseed rape. Data is described by means of six explanatory variables. SIGMEA-SIM dataset [14] describes the effects of the individual field characteristics and cropping systems on the rate of pollen and the seed dispersal rate, (target variables). Dataset includes 10368 cases. The explanatory variables of this study are two discrete variables and nine continuous ones selected in order to form contrasted situations of gene flow, ranging from maximum risk to risk free. SOLAR FLARES dataset collects measurements of the sudden and intense

variations in the brightness of the Sun for a total of 323 flare producing region of the Sun. The status is described in terms of 10 discrete variables. The task is to predict the flares production of each class (common, moderate, or severe flares) in the following 24 hours. WATER QUALITY dataset [14] comprises biological and chemical data collected through regular monitoring of rivers in Slovenia. In total, 1060 examples are available in the dataset that collects the measured values of 16 different continuous parameters together with a list of all taxa (plant and animal species) present at the sampling site and with their abundances (14 target variables). LANDSAT ETM+ dataset [6] concerns multi-temporal Landsat data that is calibrated with high resolution airborne laser scanning (ALS) data. The dataset consists of 160 continuous explanatory variables and 11 target variables. 60607 examples are used in this study.

Table 3. Multi-target regression: comparison of the average RRMSE and average size of MTSMOTI, STSMOTI, MTREGTREE, PC-Tree (PCT), PCR Ordered and PCR Unordered. For each data set, the average RRMSE over all target attributes is given.

Dataset	RRMSE						Size					
	MT SMOTI	ST SMOTI	MT REG TREE	PC T	PCR Ord.	PCR Un-Ord.	MT SMOTI	ST SMOTI	MT REG TREE	PC T	PCR Ord.	PCR Un-Ord.
EDM	0.86	0.86	0.83	0.72	0.88	0.92	4	5	7	11	9	2
MICROA	0.78	0.60	0.87	0.87	0.85	1.01	18	47	17	50	108	10
SIGMEAR	0.71	0.91	1.15	0.61	0.61	0.85	3	8	11	7	9	1
SIGMEAS	0.03	0.03	0.03	0.03	0.13	0.13	23	27	49	166	2	2
SOLARF	1.02	1.06	1.02	1	1.15	1.02	13	30	13	2	13	1
WATERQ	0.96	0.97	0.98	0.96	1.09	0.99	12	92	13	5	185	2
LANDSAT	0.67	0.64	0.69	0.62	-	-	21	238	31.9	518	-	-

For each dataset, multi-target model trees (MTSMOTI) are firstly compared with the set of single-target model trees (STSMOTI), induced one for each target variable. The RRMSE is averaged on separate target variables, while the tree size is the sum of size for all separate trees. Secondly, model trees are compared with multi-target regression trees (MTREGTREE) as well as predictive clustering trees and rules. Results on accuracy and tree size are reported in Table 3.

Results show that multi-target model trees are much smaller than the set of single-target model trees, while achieving comparable (sometime better) accuracy. MICROARTHROPODS is the only dataset where the multi-target model tree performs significantly worse than the set of single-target trees (0.78 vs. 0.60). A deeper analysis of this data reveals that the worst performance involves only the prediction of the Shannon biodiversity (0.7 vs. 0.28), while the accuracy estimates are comparable for both mites (0.85 vs 0.82) and springertails (0.78 vs 0.72). This negative result may reveal the absence of a “linear” dependence between the Shannon biodiversity and the variables mites and springertails. In any case, multi-target trees are always induced much faster than the set of single-

target ones (18 vs 25 (EDM), 202 vs 412 (MICROARTHROPODS), 5 vs 9 (SIGMEA-REAL), 69 vs 84 (SIGMEAS-SIM), <1 vs 2 (SOLAR-FLARE), 718 vs 1272 (WATER QUALITY) and 9214 vs 25372 (LANDSAT ETM+))². The comparison between multi-target model trees and multi-target regression trees reveals that although model trees are typically smaller than regression trees, they achieve comparable (or sometime better) accuracy than corresponding regression trees. In addition, MTSMOTI is capable of detecting the presence of a global effect of some explanatory variable on “all” the target variables that no previous study on these datasets have revealed. In this way, it is possible to reveal not only linear relationships between the target variables and the explanatory ones, but also the existence of some linear dependences among the target variables at different depth of the tree hierarchy.

An example of global effect is discovered on EDM data. MTSMOTI builds a multi-target model tree with one regression node in the root. The straight-line regressions (one for each target variable) selected at the root express the linear relationships between the target variables (flow of the dielectric fluid and gap between electrodes) and the explanatory variable CLM_C that describes the mean value of electric current between electrodes. The contribution of the CLM_C variable on the model to be associated with leaves is reliably estimated on the entire sample set.

Finally, the comparison with predictive clustering trees and rules³ confirms the main considerations suggested by experiments on single-target datasets. In general, model trees outperform accuracy of ordered and unordered rule learners, but unordered rules are always smaller than the corresponding model trees. Conversely, the number of ordered rules is typically higher than the size of model trees. On the other hand, predictive clustering trees are sometime more accurate than model trees (EDM and SIGMEA-REAL), but clustering trees can be significantly more complex. Finally, both clustering trees and rules predict the same constant values (for each example covered by the same leaf or rule), thus they do not be able of capturing any linear pattern in data.

4 Conclusions

In this work, we propose a new TDIMT system, MTSMOTI, that addresses the task of building multi-target model trees and predicting the values of several target variables. Each leaf of such a tree contains several linear models, each predicting the value of a different target variable. Multi-target model trees are built with two types of nodes: split nodes and regression nodes. Experiments on single-target datasets shows that MTSMOTI is competitive with respect to regression tree learner, state-of-art TIDMT systems like SMOTI and M5’ as well as predictive clustering trees and rules. Experiments on multi-target datasets,

² Running times are in secs. Experiments are performed on Intel Pentium 4 - 2.00 GHz CPU RAM 512Kb running Windows Professional 2000 on one fold of each dataset

³ Results for PC-Tree, PCR Ordered and PCR Unordered on multi-target datasets are provided by Bernard Ženko [14].

confirm that multi-target model trees are much smaller than the set of single-target model trees, while achieving comparable accuracies. In addition, they are induced much faster. As a future work, we plan to investigate differences in accuracy by using a different scaling technique and tuning mechanisms to determine thresholds for stopping criteria. Finally, we intend to combine decision trees and model trees to predict continuous and discrete target variables, simultaneously.

References

1. H. Blockeel, L. D. Raedt, and J. Ramon. Top-down induction of clustering trees. In J. Shavlik, editor, *Proceedings of the 15th International Conference on Machine Learning*, pages 55–63. Morgan Kaufmann, 1998.
2. L. Breiman, J. Friedman, R. Olshen, and J. Stone. *Classification and Regression Tree*. Wadsworth & Brooks, 1984.
3. M. Ceci, A. Appice, and D. Malerba. Comparing simplification methods for model trees with regression and splitting nodes. In N.Zong, Z. Ras, S. Tsumoto, and E. Suzuki, editors, *Foundations of Intelligent Systems, 14th International Symposium*, volume 2871 of *LNAI*, pages 49–56. Springer-Verlag, 2003.
4. B. Cestnik and I. Bratko. On estimating probabilities in tree pruning. In *Proceedings of the 5th European Working Session on Learning*, pages 151–163, 1991.
5. N. R. Draper and H. Smith. *Applied regression analysis*. John Wiley & Sons, 1982.
6. S. Džeroski, A. Kobler, V. Gjorgjioski, and P. Panov. Using decision trees to predict forest stand height and canopy cover from landsat and lidar data. In *Enviro Info 2006*, 2006.
7. A. Karalic. Linear regression in regression tree leaves. In *Proceedings of International School for Synthesis of Expert Knowledge*, pages 151–163, Slovenia, 1992.
8. D. Malerba, F. Esposito, M. Ceci, and A. Appice. Top down induction of model trees with regression and splitting nodes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(5):612–625, 2004.
9. M. Mehta, R. Agrawal, and J. Rissanen. SLIQ: A fast scalable classifier for data mining. In *Proceedings of the 5th International Conference on Extending Database Technology*, pages 18–32, 1996.
10. T. Niblett and I. Bratko. Learning decision rules in noisy domains. In M. Bramer, editor, *Research and Development in Expert Systems*, volume 3, pages 25–34. Cambridge University Press, 1986.
11. J. Struyf and S. Džeroski. Constraint based induction of multi-objective regression trees. In F. Bonchi and J.-F. Boulicaut, editors, *Workshop on Knowledge Discovery in Inductive Databases*, pages 222–233, 2005.
12. L. Torgo. Functional models for regression tree leaves. In D. Fisher, editor, *Proceedings of the 14th International Conference on Machine Learning*, pages 385–393. Morgan Kaufmann, 1997.
13. C. Vens and H. Blockeel. A simple regression based heuristic for learning model trees. *Intelligent Data Analysis*, 10(3):215–236, 2006.
14. B. Ženko. *Learning Predictive Clustering Rules*. PhD thesis, Faculty of Computer and Information Science, University of Ljubljana, Slovenia, 2007.
15. Y. Wang and I. Witten. Inducing model trees for continuous classes. In M. Van Someren and G. Widmer, editors, *Proceedings of the 9th European Conference on Machine Learning*, pages 128–137. Springer-Verlag, 1997.
16. S. Weisberg. *Applied regression analysis*. Wiley, second edition, 1985.

Service Matching and Discovery in P2P Semantic Community ^{*}

Devis Bianchini Valeria De Antonellis Michele Melchiori Denise Salvi

Università degli Studi di Brescia - Dip. di Elettronica per l'Automazione
Via Branze, 38 - 25123 Brescia - Italy
{bianchin|deantone|melchior|salvi}@ing.unibs.it

Abstract. Recently, more and more web services are being made available and their use can potentially highly increase cooperation in P2P systems, where different partners intend to share and exchange digital resources. A major concern, in this framework, is related to the development of semantic driven techniques for description and organization of available services in semantic communities for discovery purposes. We propose an ontology-based Semantic Driven Service Discovery approach in P2P systems, where peers are organized in a semantic community to support effective automation of service discovery. In the approach ontologies are exploited both to improve precision and recall of search results by means of an innovative service matchmaking strategy and to organize services in the community by means of intra-peer and inter-peer semantic links. Some experimental results are briefly discussed.

1 Introduction

Recently, an ever-growing number of web services are being made available to increase cooperation and in P2P systems, where different partners intend to share and exchange digital resources. Automation of service discovery is a crucial aspect in this context. P2P systems are featured by lack of a common understanding of the world and high heterogeneity in resource description. A major concern is related to the development of ontology-based techniques for description and organization of available services in semantic communities for discovery purposes. However, members of a P2P semantic community cannot be constrained to use the same reference ontology. In our application scenario, each peer of the community adopts its own ontology (called *peer ontology*) to express domain knowledge related to service descriptions.

During community constitution, we assume that a peer called promoter spreads out a manifesto containing a relevant portion of its peer ontology, expressing a core domain knowledge for the community. A peer that aims at joining

^{*} This work has been partially supported by the ESTEEM PRIN Project (<http://www.dis.uniroma1.it/~esteem/index.html>) and TEKNE FIRB Project (<http://www.tekne-project.it/>), funded by the Italian Ministry of Education, University and Research.

the community matches its own peer ontology against the manifesto and replies to the promoter. Once the community is established, services registered on the community members can be searched through ontology-based techniques. When the peer receives a service request expressed in terms of a different peer ontology, an innovative ontology-based service matchmaking procedure must be applied and strategies to forward the request over the P2P network must be exploited to improve discovery efficiency and avoid network overloading.

In particular, in this paper we propose the Semantic Driven Service Discovery approach for open P2P systems, that applies the service matchmaking strategies introduced in [5]: (i) to improve discovery results working with different peer ontologies in a distributed scenario; (ii) to improve performances through the identification of semantic connections between the peers of the community (called *inter-peer semantic links*) by comparing their service descriptions. This ontological framework extends functionalities of traditional UDDI Registry, improving keyword-based searching strategies.

The paper is organized as follows: Section 2 shows how different peer ontologies can be used to enable service discovery in P2P systems; Section 3 presents the adopted matchmaking models, while Section 4 explains how these models are used to define semantic links for optimization purposes; Section 5 gives some experimental results; Section 6 compares our approach with related proposals in literature; finally, in Section 7 final considerations and future work are discussed.

2 Exploitation of peer ontologies

We consider a P2P semantic community, where each peer can be a service provider or a broker where service descriptions are published in a UDDI Registry extended through its own peer ontology to provide semantic knowledge related to service descriptions. Brokers are in charge of receiving service requests, applying matchmaking models on services published locally and forwarding requests to the other brokers. Service descriptions represent functional aspects, based on the UDDI and WSDL standards, in terms of service category, service functionalities (operations) and input/output messages (parameters). Each peer ontology is constituted by: (i) a *Service Functionality Ontology (SFO)*, that provides knowledge on the concepts used to express service functionalities (operations); (ii) a *Service Message Ontology (SMO)*, that provides knowledge on the concepts used to express input and output messages (parameters) of services. In general, a service request description could contain terms not defined in the peer ontology of the broker to which the request is sent. To deal with this problem, the peer ontology is extended by a *thesaurus* that provides terms and terminological relationships (synonymy SYN, broader/narrower term BT/NT and related term RT) with reference to names of concepts in the ontology. In this way, the resulting extended registry (called *Semantic Peer Registry*) enables enlarged matchmaking capabilities when looking for correspondences between elements in service descriptions and concepts in the ontology. Furthermore, we

assume that a broker can serve more semantic communities, that is, its registry can contain descriptions of services offered by different communities.

The joint use of the peer ontology and the thesaurus allows to define the two basic concepts presented in the following.

Definition 1 (Name Affinity coefficient). *Given the thesaurus \mathcal{TH} , the Name Affinity coefficient between two terms $t, t' \in \mathcal{TH}$, denoted by $NA(t, t')$, is: (i) 1.0 if $t = t'$; (ii) $\max_l(\tau(t \rightarrow^l t'))$ if $t \neq t' \wedge t \rightarrow^l t', l \geq 1$, where $t \rightarrow^l t'$ denotes a path of terminological relationships from t to t' ; (iii) 0.0 otherwise. A weight $\sigma_{tr} \in [0, 1]$ is associated to each kind of terminological relationship tr , in order to evaluate its implication for name affinity; in our experimentation, $\sigma_{SYN} = 1$, $\sigma_{BT/NT} = 0.8$ and $\sigma_{RT} = 0.5$. The function $\tau(t \rightarrow^l t') = \prod_{k=1}^l(\sigma_{tr_k}) \in [0, 1]$ defines the strength of $t \rightarrow^l t'$ as the product of the weights of all terminological relationships in the path. Since between two terms in the thesaurus there can exist more than one path, the one with the highest strength is chosen. We say that t and t' have name affinity ($t \sim t'$) if and only if $NA(t, t') \geq \alpha$, where $\alpha > 0$ is a threshold given by experimental results to select only terms with high values of the name affinity coefficient.*

Definition 2 (Affinity-based subsumption test). *Given an atomic concept C in the peer ontology \mathcal{PO} , we define the set of terms in the thesaurus that have name affinity with a given concept as $C_{\mathcal{TH}} = \{T \in \mathcal{TH} \mid T \sim C\}$. Analogously, we define the set of concepts of \mathcal{PO} that have name affinity with a term T in \mathcal{TH} as $T_{\mathcal{PO}} = \{C \in \mathcal{PO} \mid T \in C_{\mathcal{TH}}\}$.*

Given the peer ontology \mathcal{PO} , the thesaurus \mathcal{TH} and a pair of terms $T1$ and $T2$ used in service descriptions to denote service elements, $T1$ is subsumed by $T2$ with respect to \mathcal{TH} , denoted by $T1 \sqsubseteq_{\mathcal{TH}} T2$, if and only if there exists $C \in T1_{\mathcal{PO}}$ and $D \in T2_{\mathcal{PO}}$ such that $C \sqsubseteq D$ is satisfied in \mathcal{PO} . Note that we pose $T1 \equiv_{\mathcal{TH}} T2$ if both $T1 \sqsubseteq_{\mathcal{TH}} T2$ and $T2 \sqsubseteq_{\mathcal{TH}} T1$ hold.

3 Service matchmaking model

The Name Affinity coefficient and the affinity-based subsumption test are exploited to perform service comparison according to different matchmaking models [5]: (a) a *deductive model*, based on deduction algorithms for reasoning on service functional descriptions [4]; (b) a *similarity-based model*, where the degree of match between services is measured [6]. These matchmaking models are combined together to produce better searching results according to precision and recall values, as shown in Section 5.

In the deductive model, the affinity-based subsumption test is exploited to determine the match type between a service request \mathcal{R} and a supplied service \mathcal{S} , considering separately service elements (categories, operations, parameters), as summarized in Figure 2. A formal description of these match types to infer the match between two service descriptions is given in [5]. During a pre-filtering phase, categories are used to select a set of services, called *candidate services*,

ENTITY-BASED SIMILARITY
$ESim(\mathcal{R}, \mathcal{S}) = \frac{2 \cdot A_{tot}(IN_{\mathcal{R}}, IN_{\mathcal{S}})}{ IN_{\mathcal{R}} + IN_{\mathcal{S}} } + \frac{2 \cdot A_{tot}(OUT_{\mathcal{R}}, OUT_{\mathcal{S}})}{ OUT_{\mathcal{R}} + OUT_{\mathcal{S}} } \in [0, 2]$ <p> $IN_{\mathcal{R}}, IN_{\mathcal{S}}$ - input parameter names of \mathcal{R} and \mathcal{S} $OUT_{\mathcal{R}}, OUT_{\mathcal{S}}$ - output parameter names of \mathcal{R} and \mathcal{S} $A_{tot}(IN_{\mathcal{R}}, IN_{\mathcal{S}}) = \sum_{in_{\mathcal{R}}^i \in IN_{\mathcal{R}}, in_{\mathcal{S}}^j \in IN_{\mathcal{S}}} NA(in_{\mathcal{R}}^i, in_{\mathcal{S}}^j)$ $A_{tot}(OUT_{\mathcal{R}}, OUT_{\mathcal{S}}) = \sum_{out_{\mathcal{R}}^i \in OUT_{\mathcal{R}}, out_{\mathcal{S}}^j \in OUT_{\mathcal{S}}} NA(out_{\mathcal{R}}^i, out_{\mathcal{S}}^j)$ </p>
OPERATION SIMILARITY
$OpSim(op_{\mathcal{R}}^i, op_{\mathcal{S}}^j) = NA(op_{\mathcal{R}}^i, op_{\mathcal{S}}^j) + \frac{2 \cdot A_{tot}(IN_{\mathcal{R}}^i, IN_{\mathcal{S}}^j)}{ IN_{\mathcal{R}}^i + IN_{\mathcal{S}}^j } + \frac{2 \cdot A_{tot}(OUT_{\mathcal{R}}^i, OUT_{\mathcal{S}}^j)}{ OUT_{\mathcal{R}}^i + OUT_{\mathcal{S}}^j } \in [0, 3]$ <p> $IN_{\mathcal{R}}^i, IN_{\mathcal{S}}^j$ - input parameter names of the i-th operation of \mathcal{R} and the j-th operation of \mathcal{S} $OUT_{\mathcal{R}}^i, OUT_{\mathcal{S}}^j$ - output parameter names of the i-th operation of \mathcal{R} and the j-th operation of \mathcal{S} </p>
FUNCTIONALITY-BASED SIMILARITY
$FSim(\mathcal{R}, \mathcal{S}) = \frac{2 \cdot \sum_{i,j} OpSim(op_{\mathcal{R}}^i, op_{\mathcal{S}}^j)}{ OP(\mathcal{R}) + OP(\mathcal{S}) } \in [0, 3]$ <p> $OP(\mathcal{R}), OP(\mathcal{S})$ - operations of \mathcal{R} and \mathcal{S} </p>
GLOBAL SIMILARITY
$GSim(\mathcal{R}, \mathcal{S}) = w_1 \cdot NormESim(\mathcal{R}, \mathcal{S}) + (1 - w_1) \cdot NormFSim(\mathcal{R}, \mathcal{S}) \in [0, 1]$ <p> $w_1 \in [0, 1]$ - weight used to assess the relevance of each kind of similarity $NormESim(), NormFSim()$ - $ESim()$ and $FSim()$ normalized to the range $[0, 1]$ </p>

Fig. 1. Similarity Coefficients between service descriptions \mathcal{R} (request) and \mathcal{S} (supply).

that have at least one associated category related in any generalization hierarchy with the category of \mathcal{R} in peer ontology.

Similarity-based model applies the Name Affinity coefficient between service elements to quantify the service similarity. In particular, **exact** and **plug-in** matches denote that the supplied service completely fulfills the request and service similarity is set to 1.0 (full similarity); if **mismatch** occurs, the similarity value is set to 0.0; finally, **subsume** and **intersection** matches denote partial fulfillment of the request and similarity coefficients exposed in Figure 1 are computed. Matching results are only those services such that the match type is not **mismatch** and $GSim$ value is equal or greater than a global similarity threshold δ .

MatchType	Description
Exact	\mathcal{S} and \mathcal{R} have the same capabilities, that is, they have: (i) equivalent operations; (ii) equivalent output parameters; (iii) equivalent input parameters
Plug-in	\mathcal{S} offers at least the same capabilities of \mathcal{R} , that is, names of the operations in \mathcal{R} can be mapped into operations of \mathcal{S} and, in particular, the names of corresponding operations, input parameters and output parameters are in any generalization hierarchy in the peer ontology
Subsume	\mathcal{R} offers at least the same capabilities of \mathcal{S} , that is, names of the operations in \mathcal{S} can be mapped into operations of \mathcal{R} ; it is the inverse match type with respect to plug-in
Intersection	\mathcal{S} and \mathcal{R} have some common operations and some common I/O parameters, that is, some pairs of operations and some pairs of parameters, respectively, are related in any generalization hierarchy in the peer ontology
Mismatch	Otherwise

Fig. 2. Classification of matches

4 Definition and deployment of semantic links

We propose optimization strategies based on the use of semantic links between services. These strategies aim at reducing the number of service comparisons according to the matchmaking models introduced above, finding all local services that fulfill the request and decreasing the network overload. Two kinds of semantic links can be established: (a) semantic links between services belonging to the same peer (*intra-peer semantic links*); (b) semantic links between services belonging to different peers (*inter-peer semantic links*). In the latter case, the related peers are referred as *semantic neighbors*. A peer p_i is a semantic neighbor of another peer p_j with respect to a service description \mathcal{S}_i if there exists a service description \mathcal{S}_j published on p_j such that \mathcal{S}_i and \mathcal{S}_j are related by an inter-peer semantic link. A *semantic link* is defined as a 4-uple:

$$sl = \langle source, destination, MatchType, GSim \rangle \quad (1)$$

where *source* and *destination* identify the services related through the semantic link, *MatchType* the match type and *GSim* the global similarity value. Semantic links are established once the semantic community is established. To do this, it sends a probe service request for each service it wants to make sharable; this probe service request contains the description of the service functional interface (categories, operations, I/O parameters). The probe service request is sent to all the other peers of the semantic community. Each peer receiving the probe service request matches it against its own service descriptions by applying the matchmaking techniques explained in the previous section. Note that, in a P2P context based on UDDI technology, a service is identified by: the *businessKey* and *serviceKey* in the UDDI Registry and the IP address of the host on which the UDDI Registry is located.

Service functional descriptions related by means of semantic links are stored in a service ontology for each peer of the semantic community. In particular, the peer knows its services, intra-peer semantic links between them and inter-peer semantic links toward its semantic neighbors. Intra-peer semantic links are exploited during service discovery on the broker that receives the request, to prune the set of candidate services and reduce the number of service comparisons. Inter-peer semantic links are used to select a subset of semantic neighbors of the broker that receives the request, making request forwarding over the community more efficient.

Pruning of candidate services. In this phase, we exploit previously established matching results and intra-peer semantic links to infer a match or a mismatch between request and candidate services. Our purpose here is to avoid the application of the hybrid matchmaking model if this is not strictly necessary, but at the same time to identify each offered service matching the request. Therefore, the expected result of the pruning process is an improvement of performances in the proposed approach. From an operative point of view, we suppose that the match type between the request \mathcal{R} and a candidate service \mathcal{S}_1 offered on a given peer has been already established. Now, intra-peer semantic links relating \mathcal{S}_1 with other offered services can be considered. For instance, if $\text{match}(\mathcal{R}, \mathcal{S}_1)$ evaluation results in a **plug-in** match and there exists a semantic link between \mathcal{S}_1 and \mathcal{S}_2 that states that $\text{match}(\mathcal{S}_1, \mathcal{S}_2)$ is a **plug-in**, we can infer that also the $\text{match}(\mathcal{R}, \mathcal{S}_2)$ is a **plug-in**. Generally speaking, given a match type between \mathcal{R} and \mathcal{S}_1 and a semantic link between \mathcal{S}_1 and \mathcal{S}_2 it can occur that:

- exactly one match type between \mathcal{R} and \mathcal{S}_2 can be inferred, therefore it is not necessary to apply the matchmaking model to evaluate the match between \mathcal{R} and \mathcal{S}_2 ;
- a mismatch between \mathcal{R} and \mathcal{S}_2 can be inferred; also in this case it is not necessary to apply the matchmaking model;
- more than one match type between \mathcal{R} and \mathcal{S}_2 can be inferred, allowing to restrict the set of possible match types;
- nothing can be inferred from the available information.

Figure 3 provides the details on all the different cases that can occur and shows the matches or mismatches that can be inferred in each case. The column headers of the table describe match types associated to a semantic link between \mathcal{S}_1 and \mathcal{S}_2 . The row headers describe possible match types between \mathcal{R} and \mathcal{S}_1 . An cell is empty if nothing can be inferred in association with a given column and row.

Selection of semantic neighbors. Given a service request \mathcal{R} sent to a peer p , the peer searches for suitable services in its own Semantic Peer Registry and retrieves a list $CS = \{\langle \mathcal{S}_1, GSim_1, mt_1 \rangle, \dots, \langle \mathcal{S}_n, GSim_n, mt_n \rangle\}$ of services with corresponding similarity values $GSim_i \geq \delta$ and match type mt_i different from **mismatch**.

If a service $\mathcal{S}_i \in CS$ presents an **exact** or a **plug-in** match with the request, then \mathcal{S}_i satisfies completely the required functionalities and it is not necessary to forward the service request to semantic neighbors with respect to \mathcal{S}_i . Otherwise,

$\text{match}(\mathcal{S}_1, \mathcal{S}_2)$ $\text{match}(\mathcal{R}, \mathcal{S}_1)$	exact	plug-in	subsume	intersection	mismatch
exact	(exact) 1.0	(plug-in) 1.0	(subsume) $GSim(\mathcal{S}_1, \mathcal{S}_2)$	(intersection) $GSim(\mathcal{S}_1, \mathcal{S}_2)$	(mismatch) 0.0
plug-in	(plug-in) 1.0	(plug-in) 1.0	- -	(plug-in OR intersection OR mismatch) -	(mismatch) 0.0
subsume	(subsume) $GSim(\mathcal{R}, \mathcal{S}_1)$	(exact OR plug-in OR intersection OR subsume) -	(subsume) -	(subsume OR intersection) -	(subsume OR intersection OR mismatch) -
intersection	(intersection) $GSim(\mathcal{R}, \mathcal{S}_1)$	(plug-in OR intersection) -	(intersection OR subsume OR mismatch) -	- -	(intersection OR subsume OR mismatch) -
mismatch	(mismatch) 0.0	(plug-in OR intersection OR mismatch) -	(mismatch) 0.0	(plug-in OR intersection OR mismatch) -	- -

Fig. 3. Exploitation of intra-peer semantic links to prune the set of candidate services.

if \mathcal{S}_i presents a **subsume** or an **intersection** match with the request, the peer p forwards the request to those peers that are semantic neighbors with respect to \mathcal{S}_i . Peer p does not consider semantic neighbors that present a **subsume** or an **exact** match with \mathcal{S}_i , because this means that they provide services with the same functionalities or a subset of \mathcal{S}_i functionalities and they cannot add further capabilities to those already provided by \mathcal{S}_i on the peer p . This phase is repeated for every $\mathcal{S}_i \in CS$. Semantic neighbors which present inter-peer links with any service \mathcal{S}_j stored on p , but not included in CS , are discarded since they are not relevant with respect to \mathcal{R} . Each selected semantic neighbor sn presents a set of k inter-peer semantic links with some services on p that are suitable for \mathcal{R} . It is described as $\langle sn, \{\langle \mathcal{S}_1, GSim_1, mt_1 \rangle, \dots \langle \mathcal{S}_k, GSim_k, mt_k \rangle\} \rangle$, where $\mathcal{S}_1 \dots \mathcal{S}_k \in CS$ and have a semantic link with some services stored on sn , featured by $GSim_1 \dots GSim_k$ similarity degree and $mt_1 \dots mt_k$ type of match, respectively. Note that this formulation holds even if sn has more than one service related to the same service $\mathcal{S}_i \in CS$.

Since the relevance of sn does not depend only on the similarity associated to the semantic links between p and sn , but also on the similarity degree between $\mathcal{S}_i \in CS$ and \mathcal{R} , the harmonic mean is used to combine these two aspects. Therefore, the relevance of a semantic neighbor sn is defined as:

$$r_{sn} = \frac{1}{k} \sum_{i=1}^{m_{sn}} \frac{2 * GSim_i * GSim(\mathcal{R}, \mathcal{S}_i)}{GSim_i + GSim(\mathcal{R}, \mathcal{S}_i)} \quad (2)$$

Relevance values are used to rank the set of semantic neighbors in order to filter out not relevant semantic neighbors (according to a threshold-based mechanism) and to further constrain the request forwarding (according to a token-based strategy). The harmonic mean is used here to combine the similarity values between the request and locally offered services with the similarity values of semantic links.

5 Experimental evaluation

Our approach is supported by the COMPAT MatchMaker Version 1.0, implemented in Java as a Web application. The MatchMaker exploits reasoning facilities of RACER OWL-DL reasoner Version 1.9. We applied the MatchMaker on a set of 480 web services specified in WSDL1.1 and covering touristic, *e*-banking, ERP and geographic application domains and a set of 10 queries formulated using both terms extracted from the peer ontology and terms that are not defined as atomic concepts of the ontology.

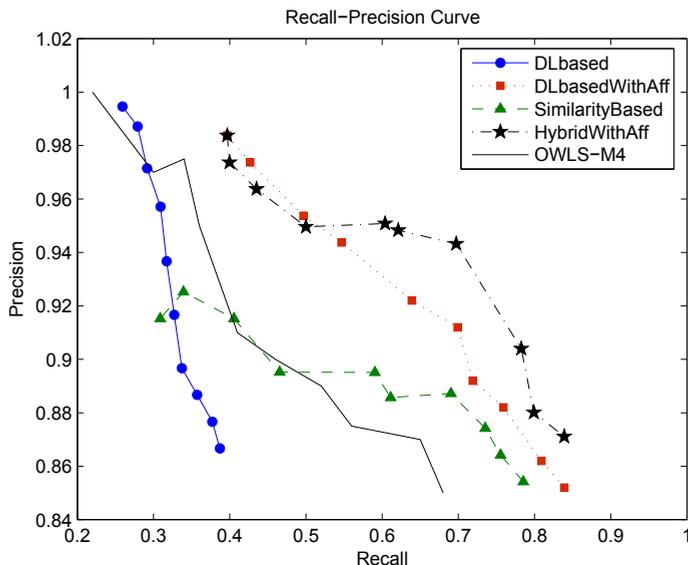


Fig. 4. Comparison of hybrid, deductive and similarity-based matching models on the basis of recall-precision performances.

Figure 4 shows the Precision-Recall curves by applying the deductive (with and without affinity-based subsumption test), the similarity-based and the hybrid matchmaking models, compared with another hybrid approach exposed in [7]. We evaluated the average values of precision and recall over the service requests by varying the global similarity threshold δ to filter out relevant results. Application of deductive matchmaking model presents high precision values, further improved by the joint use of peer ontology and thesaurus in the affinity-based subsumption test. Similarity-based matchmaking model is featured by lower precision values, although it is better than traditional IR metrics due to the use of similarity coefficients explicitly meant for service comparison. The best performances are obtained by applying the hybrid matchmaking model. Figure 5 depicts the response time for different matchmaking models, showing that the

hybrid matchmaking performs quite linearly with the number of considered services. Therefore, improvement in precision and recall offered by matchmaking model is paid in term of a higher response time with respect to other approaches. However, further experimentations aim at demonstrating that the application of the proposed optimization rules based on semantic links will improve the overall performances of the discovery approach due to the lower number of comparisons among service requests and advertisements.

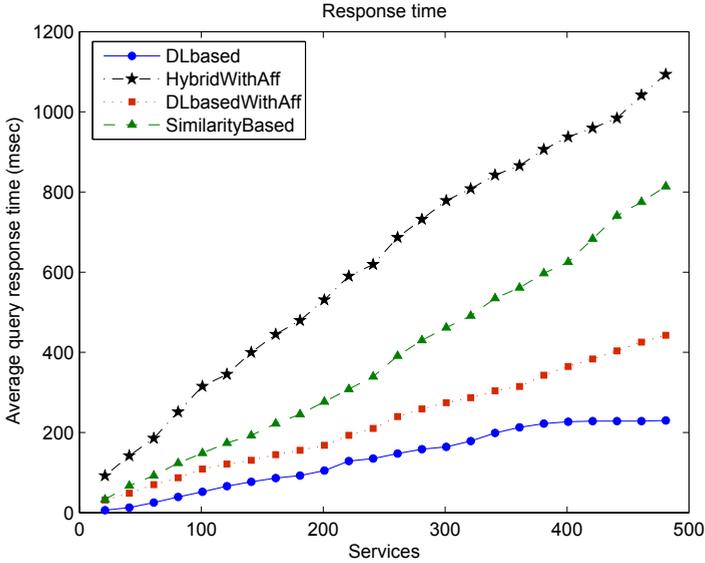


Fig. 5. Average response time of hybrid, deductive and similarity-based matching models.

6 Related work

Semantic description of services in a P2P context and semantic links between peers based on services they provide is a crucial aspect to improve effectiveness and performance of peer discovery. Several proposals have been made in literature to enable semantic-enhanced service discovery. Their main features are summarized in Table 1. Some of these approaches use centralized ontologies: GloServ [1] defines a predefined *skeleton ontology*, represented as a taxonomy of concepts each of them representing a category of services, that in turn is associated to an high level server. Each server represents a P2P network of nodes organized via a *Content Addressable Network (CAN)* [10]. These peers provide services belonging to the category represented by the ontological concept associated to their own server. The *skeleton ontology* is a centralized structure, replicated and cached in each of the high level servers.

Table 1: Comparison between approaches for P2P service semantic discovery

System	Network architecture	Semantic infrastructure	Service publishing	Service discovery
METEORS [12] focus on: - semantic overlay	hybrid P2P; special peers are introduced to handle a global ontology;	- a centralized <i>registries ontology</i> to classify peer registries; - local <i>domain specific ontologies</i> to provide service semantics; - no semantic links between peers; - semantically enriched WSDL to describe services	- service semantic descriptions are kept in semantically enhanced UDDI registries	two phases: (a) browsing of the <i>registries ontology</i> to find the proper registry; (b) deductive-based service profile matching based on I/O comparison
DAML-S for P2P [9] focus on: - service matchmaking	pure P2P	- local DAML-S ontologies to describe services; - no semantic links between peers	- service semantic descriptions are kept in local UDDI registries extended with DAML-S ontologies	deductive-based service profile matching based on I/O comparison
WSPDS [3] focus on: - semantic overlay	pure P2P	- local ontologies to provide service semantics; - semantic links between peers based on similarity of services they provide; - service semantic descriptions are made according to semantically enriched WSDL;	- service semantic descriptions are kept in semantically enhanced local UDDI registries	deductive-based service profile matching based on I/O comparison
GlobServ [1] focus on: - semantic overlay - network architecture	hybrid P2P network, with high level servers and peers in a Computer Addressable Network (CAN) organized according to a concept taxonomy	- centralized ontologies (<i>skelton ontology</i> and <i>thesaurus</i>) replicated in each high level service; - CAN lookup table to store information about peer contents inside the CAN networks; - service semantic descriptions are made according to semantically enriched WSDL	- services are provided by peers organized in the CAN networks	two phases: (a) browsing of the taxonomy of concepts to find the proper high level server; (b) keyword-based search through the CAN lookup table
WSMO-based P2P [11] focus on: - network architecture	pure P2P; network is organized as n-dimensional hypercube (according to the HyperCup model) with the purpose of improving routing efficiency	- hypercube infrastructure built according to an ontology; - peers are clustered according to an ontology concept and form a hypercube dimension; - service semantic descriptions are made according to the WSMO model	- service descriptions are kept in the WSMX registry; - service organization is flat	keyword-based matching on the non-functional service description
Artemis [2] focus on: - semantic overlay - medical domain specific services	hybrid P2P, with mediator super-peers: each peer registers to a mediator and sends it both advertisements and requests	- each peer has a coarse-grained <i>Service Functionality Ontology (SFO)</i> , to classify services, and a fine-grained <i>Service Message Ontology (SMO)</i> , to annotate services with medical concepts, based on medical information standards; each mediator super-peer keeps reference SFO and SMO and stores mappings among these and corresponding ontologies of the peers	- service descriptions are stored in UDDI registries on the mediator and annotated with ontology concepts saved in an eBXML registry on the mediator	(a) a peer sends a request to its reference mediator expressed in terms of its own ontologies; (b) the mediator uses ontology mappings to find matching services in its local registries; (c) the mediator also forwards the request to the other mediators

Service discovery is organized in two phases: (a) browsing of the taxonomy of concepts to find the proper high level server; (b) keyword-based search through the CAN lookup table.

WSPDS [3] describes a P2P network where peers have local DAML-S ontologies to provide service semantics and semantic links with other peers based on similarity of services they provide. When a request is submitted to a peer, it searches for local matching results and forwards the request to all the semantic neighbors, independently of the current request or the local results of the query. Also in WSMO-based P2P [11] the centralized ontology is used to organize peers of the P2P network as a n-dimensional hypercube (according to the HyperCup model); service semantic descriptions are made according to the WSMO model [8]. Service descriptions are stored in a registry and there is no use of semantic links between peers. METEOR-S [12] uses a centralized organization of peer registries, where service descriptions are kept in UDDI Registries semantically enhanced with local *domain specific ontologies*, while a centralized *registries ontology* is used to classify peer registries. During the discovery process, the *registries ontology* is browsed to find the proper registry to which submit the request. Sometimes, to overcome the heterogeneities between ontologies, it is required a manual-defined mediator-based architecture. For example, ARTEMIS [2] defines a network of peers, each of them has a coarse-grained functionality ontology to classify services and a fine-grained message ontology to annotate services with medical concepts, based on medical information standards. Each peer registers in a mediator super-peer the services it provides. A peer sends a request to its reference mediator expressed in terms of its own ontologies; mediator uses ontology mappings to find matching services in its local registries and also forwards the request to other mediators. Rarely semantic links between peers are considered and so the request is broadcasted on the network increasing its overload. In DAML-S for P2P [9] it is considered a network of peers, where each member stores a local DAML-S ontology to describe services; service semantic descriptions are kept in local UDDI registries extended with DAML-S ontologies. No semantic links are maintained between peers that provide similar services. Moreover, if a peer does not satisfy a request a flooding mechanism is used to find candidate services on the other peers of the network.

Original contribution of our approach relies on the application, in a flexible way, of similarity-based and deductive matchmaking models to increase precision and recall. Deductive service matching is performed with a joint use of information coming from peer ontologies and terminological aspects to deal with different reference ontologies in a P2P semantic community. Moreover, semantic links between services are exploited to speed up the discovery process and to make more efficient service request propagation on the network.

7 Concluding remarks

In this paper, we proposed the application of ontology-based strategies: (i) in an innovative service matchmaking technique, that combines different matchmaking models to improve discovery results; (ii) in the organization of services by means

of semantic links, in order to improve performances during service discovery in a semantic community built on the top of a P2P network. Heterogeneity in P2P systems is considered, without constraining peers to use the same reference ontology. Further experimentation will evaluate the impact of the proposed approach on service discovery in P2P networks according to well-known parameters (such as network overloading) and concrete applications (e.g., scientific collaboration in medicine between healthcare organizations) will be studied.

References

1. K. Arabshian and H. Schulzrinne. An Ontology-based Hierarchical Peer-to-Peer Global Service Discovery System. *Journal of Ubiquitous Computing and Intelligence (JUCI)*, 2006.
2. The ARTEMIS Project: A Semantic Web Service-based P2P Infrastructure for the Interoperability of Medical Information Systems. <http://www.srdc.metu.edu.tr/webpage/projects/artemis/>.
3. F. Banaei-Kashani, C. Chen, and C. Shahabi. WSPDS: Web Services Peer-to-Peer Discovery Service. In *Proc. of the Int. Conference on Internet Computing (IC'04)*, pages 733–743, Las Vegas, Nevada, USA, 2004.
4. D. Bianchini, V. De Antonellis, and M. Melchiori. Capability Matching and Similarity Reasoning in Service Discovery. In *CAiSE Int. Workshop on Enterprise Modeling and Ontologies for Interoperability, EMOI 2005*, Porto, Portugal, June 2005.
5. D. Bianchini, V. De Antonellis, M. Melchiori, and D. Salvi. Semantic-enriched Service Discovery. In *IEEE ICDE Int. Workshop on Challenges in Web Information Retrieval and Integration, WIRI 2006*, Atlanta, Georgia, USA, 2006.
6. D. Bianchini, V. De Antonellis, B. Pernici, and P. Plebani. Ontology-based Methodology for e-Service discovery. *Journal of Information Systems, Special Issue on Semantic Web and Web Services*, 31(4-5):361–380, June-July 2006.
7. B. Fries, M. Khalid, M. Klusch, and K. Sycara. OWLS-MX: Hybrid OWL-S Service Matchmaking. In *Proc. of First International AAAI Symposium on Agents and Semantic Web*, Arlington, VA, USA, November 2005.
8. U. Keller, H. Lausen, and D. Roman. *Web Service Modeling Ontology (WSMO)*. WSMO Working Draft, March 2004. <http://www.wsmo.org/2004/d2/v02/20040306/>.
9. M. Paolucci, K.P. Sycara, T. Nishimura, and N. Srinivasan. Using DAML-S for P2P Discovery. In *Proceedings of the Int. Conference on Web Services (ICWS2003)*, 2003.
10. S. Ratnasamy, P. Francis, M. Handley, R.M. Karp, and S. Shenker. A Scalable Content-Addressable Network. In *Proc. of the ACM SIGCOMM'01 Conference on Applications, Technologies, Architectures and Protocols for Computer Communication*, pages 161–172, San Diego, CA, USA, August 2001.
11. I. Toma, B. Sapkota, J. Scicluna, J.M. Gomez, D. Roman, and D. Fensel. A P2P Discovery mechanism for Web Service Execution Environment. In *Proc of the 2nd WSMO Implementation Workshop (WIW'05)*, Innsbruck, Austria, June 6th-7th 2005.
12. K. Verma, K. Sivashanmugam, A. Sheth, A. Patil, S. Oundhakar, and J. Miller. METEOR-S WSDI: A Scalable Infrastructure of Registries for Semantic Publication and Discovery of Web Services. *Journal of Information Technology and Management, Special Issue on Universal Global Integration*, 6(1):17–39, 2005.

Mining Time-series Sequences of Reactions for Biologically Active Patterns in Metabolic Pathways

Marenglen Biba, Floriana Esposito, Stefano Ferilli,
Nicola Di Mauro, Teresa M.A Basile

Department of Computer Science, University of Bari
Via E. Orabona, 4 - 70125 Bari, Italy
{biba, esposito, ferilli, ndm, basile}@di.uniba.it

Abstract. Large quantities of metabolic profiling data are being gathered intensively in the rapidly growing field of Metabolomics. However, such data, in order to provide knowledge, must be machine-explored by robust methods that deal with complexity and uncertainty. Symbolic machine learning methods have the power to model structural and relational complexity while statistical machine learning ones provide principled approaches to uncertainty modeling. In this paper, we apply a hybrid symbolic-statistical framework to mine time-series sequences of reactions for biologically active paths in metabolic networks. We show through experiments that our approach provides a robust methodology for knowledge discovery in Systems Biology.

1 Introduction

Metabolomics [1] is a rapidly growing field. Analytical techniques and instruments such as Mass Spectrometry (MS) and Nuclear Magnetic Resonance (NMR) for gathering and analyzing voluminous metabolic data are being intensively refined. MS is now able to detect molecules at concentrations as low as 10^{-18} molar, and high-field NMR can efficiently differentiate between molecules that are highly similar in structure. The main problem in this research area is the study of the metabolome [2] which represents the collection of all the metabolites in a biological organism. This set of molecules consists of metabolic intermediates, hormones and other signalling molecules, and secondary metabolites. All these represent the chemical fingerprints that every specific cellular process leaves behind. Thus, in order to understand how cells work it is important to explore the metabolome in a principled and robust manner. However, the separate study of the metabolome would not give a deep comprehension of the organism, because biological systems' behavior is determined by complex interactions between their building components. Therefore, an integrated approach to studying biological systems is necessary. This has given rise to the Systems Biology [3] approach to modeling biological phenomena. In Systems Biology the main problem is to uncover and model how function and behavior of the biological machinery are implemented through complex interactions among its building blocks. Metabolomics data provide precious traces of the cell's circuits

functioning, hence it is highly important for the Systems Biology approach to integrate metabolomics for a deeper understanding of biological systems [4].

Since biological circuits are hard to model and simulate, many efforts [5] have been made to develop computational models that can handle their intrinsic complexity. In this paper we focus on a particular problem of Systems Biology that concerns the modeling of metabolic pathways and the possibility to discover biologically active paths. A metabolic pathway is a sequence of chemical reactions occurring within the cell. These reactions are catalyzed by enzymes which are particular proteins that convert metabolites (input molecules) in other molecules that represent the products of the reaction. These products can be stored in the cell under certain forms or can cause the initiation of another metabolic pathway. A metabolic network of a cell is formed by the metabolic pathways occurring in the cell. It is through the metabolic networks that every single living organism carries out all its activities. Thus, pathway analysis is crucial to understand cell's behavior and machine learning methods, that are not limited to only simulate biological networks, are essential to infer knowledge from exponentially growing observation data gathered by high-throughput instruments.

Since a reaction can happen if the input molecules are available to the catalytic enzyme, a modeling framework must be able to model relations among entities. Symbolic approaches such as logic-based techniques have the potential to model relations in structural complex domains. First-order logic representations have also the advantage that models are easily comprehensible to humans. Moreover, since most part of biological systems performs its activity remaining hidden to the human modeler, machine learning techniques can play an important role in discovering latent phenomena. However, symbolic-only approaches suffer from the incapability of handling uncertainty. In models built with symbolic-only approaches, the learned rules are deterministic and do not incorporate any kind of mechanism for uncertainty modeling. On the other side, biological systems intrinsically behave in a stochastic fashion with many interactions probable to happen. Since cell's life is determined by the most probable interactions, handling uncertainty is crucial when the cell's machinery must be modeled. Statistical approaches based on the probability theory represent a valuable mechanism to govern uncertainty. However, observations of biological systems rarely reflect exactly what happens inside them. Therefore, estimation techniques are precious in order to model what we cannot observe. Statistical machine learning methods have the ability to learn probability distributions from observations and hence are suitable for modeling biological systems. On the other side, statistical-only approaches rarely are able to reason about relations and/or interactions among biological circuits as symbolic approaches do. Hence, there is strong motivation on developing and applying hybrid approaches to modeling biological systems.

The contribution of this paper is at the intersection of Systems Biology, Metabolomics and Machine Learning. We apply a hybrid symbolic-statistical framework to the problem of modeling metabolic pathways and mining active paths from time-series data. We show through experiments the feasibility of mining significant paths from metabolomics data in the form of traces of sequences of reactions.

The paper is organized as follows. Section 2 describes the problem of modeling metabolic pathways and the necessity for symbolic-statistical machine learning. Section 3 describes the hybrid framework PRISM. Section 4 describes modeling in PRISM of the *Bisphenol A Degradation* pathway of *Dechloromonas aromatica*. Section 5 presents experiments on mining stochastically generated sequences of reactions for biologically active paths. Section 6 concludes discussing related and future work.

2 Metabolic Pathways

Metabolic pathways can be represented as graphs where each node represents a chemical compound and a chemical reaction corresponds to a directed edge labeled by a protein that catalyzes the reaction. Thus, there is an edge from one compound (metabolite) to another compound (product) if there is an enzyme that transforms the metabolite into product. Figure 1. shows part of the pathway of *Bisphenol A Degradation* in *Dechloromonas aromatica* extracted from KEGG¹ database. We have chosen this pathway from the KEGG because, as we can see from Figure 1, starting from one point in the pathway there are multiple paths that can be explored. Therefore, the task of mining biologically active paths is harder because more paths should be explored in order to discover the active ones.

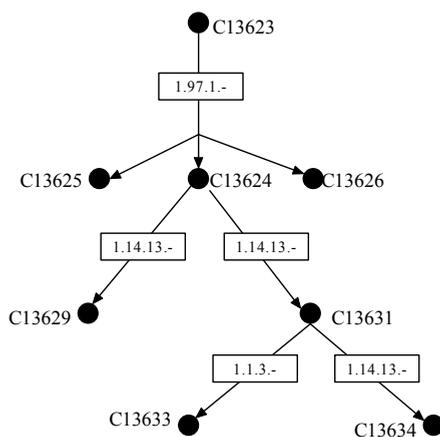


Fig. 1. Part of the metabolic pathway of *Bisphenol A Degradation* in *Dechloromonas aromatica*. The complete pathway is available from KEGG.

In order to model a metabolic pathway, a suitable framework for their simulation and mining must be able to handle relations. First-order logic representations have the

¹ <http://www.genome.jp/kegg/>

expressive power to model structural and relational problems. The metabolic pathway in Figure 1 can be easily represented in a first-order logic formalism as follows:

```
enzyme( 1.97.1.-, reaction_1_97_1_, [c13623], [c13625,c13624,c13626]).
enzyme( 1.14.13.-, reaction_1_14_13_a, [c13624], [c13629]).
enzyme( 1.14.13.-, reaction_1_14_13_b, [c13624], [c13631]).
enzyme( 1.1.3.-, reaction_1_1_3, [c13631], [c13633]).
enzyme( 1.14.13.-, reaction_1_14_13_c, [c13631], [c13634]).
```

However, this representation does not incorporate any further information about the reactions. For example, as we can see there are two competing reactions because the enzyme *1.14.13.-* catalyzes two different reactions with the same chemical compound c13624 in input. Subsequently, two enzymes, *1.14.13.-* and *1.1.3.-*, can elaborate the same input metabolites and thus two reactions compete among them. The occurring of any of the reactions determines a certain sequence of successive reactions instead of another. Hence, it is important to know which reaction among the two is more probable to happen. The most probable reaction determines the biologically active path under certain conditions. This means that under certain conditions, a biological path becomes inactive or useless and another path may become active and yield different overall products in the whole pathway. The conditions under which the reactions happen, may change stochastically due to the random behavior of the biological environment. For example, some input metabolites can suddenly be not available. Their absence can cause a certain reaction not to occur and give rise to another sequence in the metabolic pathway. Therefore, it is crucial to know how probable a certain reaction is. This situation can be modeled by attaching to each reaction the probability that it happens. This requires a first-order representation framework that can handle for each predicate that expresses a reaction the probability that the predicate is true.

The simple incorporation of probabilities is not enough to model complex metabolic networks. The conditions for the reactions to happen depend on many factors, such as initial quantity of input metabolites, changes in the physical-chemical environment surrounding the cell and many more. For this reason it is a hard task to observe all the states of the biological machinery under all the possible conditions and try to assign probabilities to reactions. Therefore there is a need for machine learning statistical methods that given certain conditions can learn distribution of probabilities from observations (the conditions here are meant as physical-chemical entities such as temperature, concentration of metabolites, entropy etc).

In order to model metabolic networks, two tasks must be performed. First, a relational model that describes the structure of the pathway must be build. There is already a large amount of accumulated knowledge about the structure of metabolic pathways such as that in KEGG and we can use all this background knowledge to skip the structure building process and concentrate on mining raw wet experimental-observational data. Indeed, graph structures are abundant but their main disadvantage in modeling cell's life is that they are static. This means that the pathway in Figure 1. does not express the stochastic dynamics in metabolic reactions. These graphs can be seen as useful static templates to interpret what can happen in the cell, but to faithfully reconstruct the cell's activity we must build a dynamic model that

represents at a certain moment and under certain conditions what happens inside the cell. Thus, in order to mine biologically active patterns in the pathway under some conditions, we must first learn a dynamic-stochastic model from sequences of reactions that have been observed under those conditions. In order to confirm the feasibility of our approach of mining biological active patterns, we will proceed as follows. We will stochastically change the conditions for the reactions to happen (Section 5 describes how this is performed). Then, under each set of conditions, we stochastically generate sequences of reactions and finally after learning probability distributions for the reactions of the pathway, we perform mining for biological active patterns by querying the dynamic model we have built.

3 The Hybrid Symbolic-Statistical Framework PRISM

PRISM (PRogramming In Statistical Modelling) [6] is a symbolic-statistical modeling language that integrates logic programming with learning algorithms for probabilistic programs. PRISM programs are not only just a probabilistic extension of logic programs but are also able to learn from examples through the EM (Expectation-Maximization) algorithm which is built-in in the language. PRISM represents a formal knowledge representation language for modeling scientific hypotheses about phenomena which are governed by rules and probabilities. The parameter learning algorithm [7], provided by the language, is a new EM algorithm called graphical EM algorithm that when combined with the tabulated search has the same time complexity as existing EM algorithms, i.e. the Baum-Welch algorithm for HMMs (Hidden Markov Models), the Inside-Outside algorithm for PCFGs (Probabilistic Context-Free Grammars), and the one for singly connected BNs (Bayesian Networks) that have been developed independently in each research field. Since PRISM programs can be arbitrarily complex (no restriction on the form or size), the most popular probabilistic modeling formalisms such as HMMs, PCFGs and BNs can be described by these programs.

PRISM programs are defined as logic programs with a probability distribution given to facts that is called basic distribution. Formally a PRISM program is $P = F \sqcap R$ where R is a set of logical rules working behind the observations and F is a set of facts that models observations' uncertainty with a probability distribution. Through the built-in graphical EM algorithm the parameters (probabilities) of F are learned and through the rules this learned probability distribution over the facts induces a joint probability distribution over the set of least models of P , i.e. over the observations. This is called *distributional semantics* [8]. As an example, we present a hidden markov model with two states slightly modified from that in [7]:

```

values(init,[s0,s1]).           % State initialization
values(out(_),[a,b]).          % Symbol emission
values(tr(_),[s0,s1]).        % State transition
hmm(L):-                       % To observe a string L:
    str_length(N),             % Get the string length as N
    msw(init,S),               % Choose an initial state randomly

```

```

hmm(1,N,S,L).           % Start stochastic transition (loop)

hmm(T,N,_,[]):- T>N,!. % Stop the loop
hmm(T,N,S,[Ob|Y]) :-   % Loop: current state is S, current time is T
  msw(out(S),Ob),      % Output Ob at the state S
  msw(tr(S),Next),     % Transit from S to Next.
  T1 is T+1,           % Count up time
  hmm(T1,N,Next,Y).   % Go next (recursion)
str_length(10).        % String length is 10
set_params :- set_sw(init, [0.9,0.1]), set_sw(tr(s0), [0.2,0.8]), set_sw(tr(s1),
[0.8,0.2]), set_sw(out(s0),[0.5,0.5]), set_sw(out(s1),[0.6,0.4]).

```

The most appealing feature of PRISM is that it allows the users to use random switches to make probabilistic choices. A random switch has a name, a space of possible outcomes, and a probability distribution. In the program above, `msw(init,S)` probabilistically determines the initial state from which to start by tossing a coin. The predicate `set_sw(init, [0.9,0.1])`, states that the probability of starting from state `s0` is 0.9 and from `s1` is 0.1. The predicate *learn* in PRISM is used to learn from examples (a set of strings) the parameters (probabilities of `init`, `out` and `tr`) so that the ML (Maximum-Likelihood) is reached. For example, the learned parameters from a set of examples can be: `switch init: s0 (0.6570), s1 (0.3429); switch out(s0): a (0.3257), b (0.6742); switch out(s1): a (0.7048), b (0.2951); switch tr(s0): s0 (0.2844), s1 (0.7155); switch tr(s1): s0 (0.5703), s1 (0.4296)`. After learning these ML parameters, we can calculate the probability of a certain observation using the predicate *prob*: `prob(hmm([a,a,a,a,a,b,b,b,b])) = 0.000117528`. This way, we are able to define a probability distribution over the strings that we observe. Therefore from the basic distribution we have induced a joint probability distribution over the observations.

4 Modeling Bisphenol A Degradation Pathway in PRISM

Since PRISM is a logic-based language, we can easily represent the metabolic pathway presented in the previous section. Predicates that describe reactions remain unchanged from a language representation point of view. What we need to statistically model the metabolic pathway is the extension with random switches of the logic program that describes the pathway. We define for every reaction a random switch with its relative space outcome. For example, in the following we describe the random switches for the reactions in Figure 1.

```

values(switch_rea_1_97_1, [rea_1_97_1( yes, yes, yes, yes), rea_1_97_1( yes, no, no,
no)]).
values(switch_rea_1_14_13_a,[rea_1_14_13_a(yes, yes), rea_1_14_13_a(yes, no)]).
values(switch_rea_1_14_13_b,[rea_1_14_13_b( yes, yes ),rea_1_14_13_b( yes, no)]).
values(switch_rea_1_1_3,[rea_1_1_3( yes, yes ),rea_1_1_3( yes, no)]).
values(switch_rea_1_14_13_b,[rea_1_14_13_c( yes, yes), rea_1_14_13_c( yes, no)]).

```

For each of the three reactions there is a random switch that can take one of the stated values at a certain time. For example, the value *rea_1_97_1(yes, yes)* means that at a certain moment the metabolite c13623 is available and the reaction occurs producing the compounds c13623, c13624 and c13625. While the other value *rea_1_97_1(yes, no, no, no)* means that the input metabolite is present but the reaction stochastically did not occur, thus the products are not produced. Below we report the remaining part of the PRISM program for modeling the pathway in Figure 1. Together with the declarations in Section 2 for the possible reactions and those of the previous paragraph for the values of the random switches, the following logic program forms a model for stochastically modeling the pathway in Figure 1. (The complete PRISM code for the whole metabolic pathway can be requested to the authors).

```

produces( Metabolites, Products ) :-
    produces( Metabolites, [], Products ).

produces( Metabolites, Delayed, Products ) :-
    ( reaction( Metabolites, Name, Inputs, Outputs, Rest ) ->
      call_reaction( Reaction, Inputs, Outputs, Call ),
      rand_sw(Call, Value),
      ((Value == rea_1_97_1( yes, yes, yes, yes );
       Value == rea_1_14_13_a( yes, yes, );
       Value == rea_1_14_13_b( yes, yes, );
       Value == rea_1_14_13_c( yes, yes, );
       Value == rea_1_1_3( yes, yes )) ->
      produces( Rest, Delayed, Products )
      ;
      produces( Metabolites, [Reaction|Delayed], Products)
      ;
      Products = Metabolites
    ).

rand_sw(ReactAndArgs, Value):-
    ReactAndArgs =..[Predicate|Arguments],
    (Predicate == rea_1_97_1-> msw(switch_rea_1_97_1, Value) ;
    (Predicate == rea_1_14_13_a-> msw(switch_rea_1_14_13_a, Value);
    (Predicate == rea_1_14_13_b->msw(switch_rea_1_14_13_b, Value);
    (Predicate == rea_1_14_13_c->msw(switch_rea_1_14_13_c, Value);
    (Predicate == rea_1_1_3->msw(switch_rea_1_1_3, Value)
    ;
    true))))). % do nothing

```

In the following, we trace the execution of the above logic program. The top goal to prove that represents the observations (sequences of reactions vastly produced by high-throughput technologies) for PRISM is *produces(Metabolites, Products)*. It will succeed if there is a pathway that leads from *Metabolites* to *Products*, in other words if there is a sequence of random choices (according to a probability distribution) that

makes possible to prove the top goal. The predicate *reaction* controls among the first clauses of the program, if there is a possible reaction with *Metabolites* in input. Suppose that at a certain moment *Metabolites* = [c13624] and thus two competing reactions can happen. Suppose one of the reaction is stochastically chosen and the variables *Inputs* and *Outputs* are bounded respectively to [c13624] and [c13629]. The predicate *call_reaction* constructs the body of the reaction that is the predicate *Call* which is in the form: *rea_1_14_13_a(_,_,_)*. This means that the next predicate *rand_sw* will perform a random choice for the switch *switch_rea_1_14_13_a*. This random choice which is made by the built-in predicate *msw(switch_rea_1_14_13_a, Value)* of PRISM, determines the next step of the execution, since *Value* can be either *rea_1_14_13_a(yes, yes)* or *rea_1_14_13_a(yes, no)*. In the first case it means the reaction has been probabilistically chosen to happen and the next step in the execution of the program which corresponds to the next reaction in the metabolic pathway is the call *produces(Rest, Delayed, Products)*. In the second case, the random choice *rea_1_14_13_a(yes, no)* means that probabilistically the reaction did not occur and the sequence of the execution will be another, determined by the call *produces(Metabolites, [Reaction|Delayed], Products)* which will try stochastically to choose the competing reaction catalyzed by the same enzyme 1.14.13.- that given the same input c13624 produces the compound c13631. If this reaction occurs, then the next reaction in the sequence will be one of the competing reactions with c13631 as input.

In order to learn the probabilities of the reactions we need a set of observations of the form *produces(Metabolites, Products)*. These observations that represent metabolomic data, are being intensively collected through available high throughput instruments and stored in metabolomics databases. In the next section, we show that from these observations, PRISM is able to accurately learn reaction probabilities through the built-in graphical EM algorithm.

5 Mining Stochastically Generated Sequences of Reactions

A certain metabolic path becomes inactive or useless under certain conditions if a certain intermediate reaction in the path cannot occur under those conditions. In this paper we are not interested in the conditions themselves (these usually are stoichiometric constraints). What is important for our purpose here, is that the conditions evolve stochastically. This means that by simulating various conditions we make possible a set of reactions instead of another, i.e. each set of conditions gives rise to a set of possible reactions that render some paths in the metabolic pathway biologically active and others biologically inactive under those conditions. In order to simulate various conditions, for each experiment we randomly assign probabilities to reactions. These probabilities represent the switches probabilities in PRISM. Thus, we have for each single experiment a set of conditions under the form of assigned reactions' probabilities (as probabilities are randomly generated and some of them may be equal to zero or in the range [0,9 - 0,999], among competing reactions one of them may not occur and this will cause some paths in the metabolic pathway to be inactive). The model constructed in this manner reflects the state of the biochemical

environment under the given conditions at a certain moment. When the reactions happen, what is caught by a high-throughput instrument is a set of metabolites concentrations and their changes. For example, if a certain reaction happens then the concentration of the input metabolites decrease and that of the product compounds increase. This change is registered as a reaction, therefore catching all the time-series changes in concentration (this is actually performed intensively and accurately by current high-throughput technologies), means registering a time-series sequence of reactions. These constitute our mining data in order to re-construct biological active and inactive paths. By simulating the built model (this corresponds to simply running the PRISM program by calling the goal *produces(InputMetabolites, Products)* where *InputMetabolites* is a bounded list with the input compounds and *Products* is a logic variable that will be bounded to the list of product compounds yielded by the series of reactions), we will have time-series sequences of reactions as if we were observing the model by high-throughput instruments.

In order to evaluate the validity of our approach we have proceeded as follows. For each experiment (each experiment has a different set of conditions, i.e. probabilities of random switches that are stochastically assigned) we have stochastically generated sequences of reactions by sampling from the previously defined model. This is made possible by the predicate *sample* of PRISM. Once the sequences have been generated, we launch the predicate *learn* of PRISM to learn the probability of each random switch from the sequences. Once the model has been reconstructed we query it over the sequences and mine biologically active paths with the predicate *hindsight(Goal)* where *Goal* is bounded to the top-goal *[InputMetabolites, Products]*. With this predicate we get the probabilities of all the sub-goals for the top-goal *Goal*. If any of these probabilities is equal to zero then the relative path of the sub-goal is biologically inactive under the given conditions. The relative path can be extracted by the predicate *probfl(SubGoal, ExplGraph)* where *ExplGraph* (explanation graph in PRISM) represents the explanation paths for *SubGoal*.

The accuracy of mining the sequences of reactions for biologically active patterns, depends on the ability to faithfully reconstruct the model from the sequences. In order to assess the accuracy of learning the probabilities of the reactions and mining really biologically active paths we adopt the following method to evaluate the learning phase for the approach of the previous paragraph. We call the initial probability distribution P_1, \dots, P_M (that represents the conditions) assigned to the clauses of the logic program the true probability distribution and call the M parameters the true parameters. Once the sequences have been stochastically generated by this model, we forget the true parameters and replace their probabilities by uniformly distributed ones. When learning starts, PRISM learns M new parameters P'_1, \dots, P'_M , that represent the learned reaction probabilities from the sequences. In order to assess the accuracy of the learned P'_i towards P_i we use the RMSE (Root Mean Square Error) for each single experiment with S sequences.

$$\text{RMSE} = \sqrt{\left(\frac{\sum_{i=1}^M (P_i - P'_i)^2}{M} \right)}$$

In this way we can measure the difference between the actual observations and the response predicted by the model. We have performed different experiments with a growing number S of sequences in order to evaluate how the number of sequences affects the accuracy and the learning time. Moreover, we wanted to test also large datasets of sequences in order to provide a robust methodology since real metabolomics datasets are in general highly voluminous. For each S we have performed 100 experiments where for each experiment the set of conditions is stochastically generated as presented above. Table 1. reports for each S the RMSE and the learning time on average for 100 experiments. We have used the version 1.10 of the system PRISM on a Pentium 4, 2.4GHz machine.

Table 1. RMSE and learning time on average for 100 experiments

S – Number of sequences	Mean of RMSE on 100 experiments	Mean learning time on 100 experiments (seconds)
100	0,13932	0,047
200	0,13593	0,068
500	0,12999	0,090
1000	0,10405	0,125
2000	0,09685	0,297
4000	0,08676	0,484
8000	0,06808	0,547
15000	0,05426	0,612
30000	0,03297	0,695
50000	0,02924	0,735
100000	0,02250	1,172

As Table 1 shows, the learning accuracy increases as more data are available and due to the tabulation techniques in PRISM, learning times increases reasonably as data dimension grows significantly. The accuracy of learning can be evaluated as good for a number of sequences between 1000 and 15.000 and excellent for a number of sequences greater than 15.000 considering that the range where probabilities fall is $[0, \dots, 1]$ and the RMSE is under 0,05. This means that the paths have been faithfully reconstructed from the sequences and thus the predicates *hindsight* and *probf* in PRISM faithfully produce the biologically active paths in the pathway. Indeed, from empirical observations, we noted that all the queries performed by these two predicates reflected the real biological paths that are supposed to have produced the sequences. For instance, we noted that anytime the probability of the reaction catalyzed by the enzyme *1.14.13.-* (with input the compound c13624 and output c13631) was stochastically assigned to be too low (from 0 to 0.05) by the conditions generation phase, then the path that involves one of the two next reactions, the one catalyzed by the enzyme *1.1.3.-* and producing in output c13633, was mined as a biologically inactive path for the given conditions. Moreover, we noted for all the experiments that by slightly changing the conditions, many inactive paths became suddenly active and vice versa. This is quite interesting since it means that we can learn from sequences how conditions evolve in order to understand what changes them and what governs their randomness.

6. Discussion and Future Work

We have applied the hybrid symbolic-statistical framework PRISM to a problem of modeling metabolic pathways and have shown through experiments the feasibility of learning reaction probabilities from metabolomics data and mining biologically active paths from time-series sequences of reactions. The power of the proposed approach stands in the description language that allows to model relations and in the ability to model uncertainty in a robust manner. Moreover, we have also shown that the symbolic-statistical framework PRISM can be used as a stochastic simulator for biochemical reactions.

The most important related work is that in [9] where a probabilistic relational formalism is used for modeling metabolic networks. The PRISM program we have presented here is syntactically quite similar to the logic program in [9], but is semantically different in the way probability distributions are defined. Stochastic Logic Programs (SLPs) [10], used in [9], assign probabilities to clauses and define probability distributions on Prolog proof trees, while PRISM programs are based on the *distributional semantics* [8] and assign probabilities to atoms as we explained in Section 3. Most of other related work is not based on symbolic-statistical approaches. In [11, 12], graph-theory based approaches are used to find common or unique sub-graphs in different pathway graphs to understand better why and how pathways differ or are similar. Other approaches are those that focus on text mining for metabolic pathways [13]. These methods have been applied to the voluminous literature on metabolic pathways to discover knowledge about the structure of the pathways. Text mining techniques focus on the structure building process trying to identify, in the accumulated experience about metabolic pathways, significant structural properties. Other approaches attempt to only stochastically simulate biochemical processes such StochSim [14] or FluxAnalyzer [15]. These are powerful tools to model the dynamic nature of cells for simulation purposes but lack machine learning abilities to infer knowledge from observations.

Although we have been able to reconstruct the model from the sequences of reactions, our approach is far from completing the real picture of a biochemical network. Much work remains to be done. First of all, we have not considered stoichiometric constraints which express quantitative relationships of the reactants and products in chemical reactions. We believe that adding these constraints to our approach will help reproduce better models. Another direction for future work regards plugging in the model other sources of data. Considering multiple sources of data can lead to better models in modeling metabolic pathways [16]. In PRISM this is straightforward because relational problems can be easily modeled due to the logic-based language. Another challenge is learning from incomplete raw metabolomic data. EM algorithms [17] are the state-of-the art for learning in the presence of missing data and since the graphical EM algorithm [7] that PRISM uses, is a version of this class of learning algorithms, we believe this will help in dealing with incomplete real datasets. In addition, in this paper we have considered a medium-sized metabolic pathway. For future work we intend to model very large metabolic pathways and hierarchical metabolic networks to see how the learning algorithms in PRISM scales for large datasets. We think the tabulation techniques used in PRISM will greatly help in dealing with a high number of paths to be explored. We also plan

to investigate other important problems using the symbolic-statistical framework PRISM and other learning capabilities such as inductive relational learning for inferring missing pathways in existing metabolic networks or reconstructing whole novel pathways from sequences of observations.

References

1. Harrigan, G. G. and Goodacre, R. (eds). *Metabolic Profiling: Its Role in Biomarker Discovery and Gene Function Analysis*. Kluwer Academic Publishers, Boston (2003).
2. Oliver, S. G., Winson, M. K., Kell, D. B. and Baganz, F. Systematic functional analysis of the yeast genome. *Trends Biotechnol.* 16 (10): 373–378, (1998).
3. Kitano, H. (editor). *Foundations of Systems Biology*. MIT Press (2001).
4. Weckwerth, W. Metabolomics in systems biology. *Annu. Rev. Plant Biol.* 54, 669–689 (2003).
5. Kriete, A. Eils, R. *Computational Systems Biology*, Elsevier - Academic Press (2005).
6. Sato, T. and Kameya, Y. PRISM: A symbolic-statistical modeling language. In *Proceedings of the 15th International Joint Conference on Artificial Intelligence*, pp.1330–1335, (1997).
7. Sato, T. and Kameya, Y. Parameter learning of logic programs for symbolic-statistical modeling. *Journal of Artificial Intelligence Research*, Vol.15, pp.391–454, (2001).
8. Sato, T. A statistical learning method for logic programs with distribution semantics. In Leon Sterling, editor, *Proc. Twelfth International Conference on Logic Programming*, pages 715-729, Kanagawa, Japan, MIT Press, (1995).
9. Angelopoulos N. and Muggleton S.H. Machine learning metabolic pathway descriptions using a probabilistic relational representation. *Electronic Transactions in Artificial Intelligence*, 6, (2002).
10. Muggleton, S.H. Stochastic logic programs. In L. de Raedt, editor, *Advances in Inductive Logic Programming*, pages 254-264. IOS Press, (1996).
11. Koyuturk, M., Grama, A., and Szpankowski, W. An Efficient Algorithm for Detecting Frequent Subgraphs in Biological Networks, *Bioinformatics*, Suppl. 1: *Proc. 12th Intl. Conf. Intelligent Systems for Molecular Biology* (ISMB'04), 200-207, (2004).
12. You, C.H., Holder, L.B., and Cook: J. Application of Graph-based Data Mining to Metabolic Pathways. *Workshop on Data Mining in Bioinformatics*, ICDM, (2006).
13. Hoffmann, R., Krallinger, M., Andres, E, Tamames, J., Blaschke, C., Valencia, A. Text mining for metabolic pathways, signaling cascades, and protein networks *Sci STKE* 283, 21 (2005).
14. Le Novère, N. & Shimizu, T. S. StochSim: modelling of stochastic biomolecular processes. *Bioinformatics* 17, 575-576, (2001).
15. Klamt S, Stelling J, Ginkel M and Gilles ED. FluxAnalyzer: exploring structure, pathways, and flux distributions in metabolic networks on interactive flux maps. *Bioinformatics* 19(2): 261-269, (2003).
16. Fiehn, O. Combining genomics, metabolome analysis, and biochemical modelling to understand metabolic networks. *Comp. Funct. Genomics* 2 (3): 155–168, (2001).
17. Dempster, A. P., Laird, N. M., and Rubin, D. B.. Maximum likelihood from incomplete data via the EM algorithm. *Royal Statistical Society*, B39(1), 1{38, (1977).

A Fast Partial Memory Approach to Incremental Learning through an Advanced Data Storage Framework

Marenglen Biba, Stefano Ferilli, Floriana Esposito,
Nicola Di Mauro, Teresa M.A Basile

Department of Computer Science, University of Bari
Via E. Orabona, 4 - 70125 Bari, Italy
{biba, ferilli, esposito, ndm, basile}@di.uniba.it

Abstract. Inducing concept descriptions from examples has been thoroughly tackled by symbolic machine learning methods. However, on-line learning methods that acquire concepts from examples distributed over time, require great computational effort. This is not only due to the intrinsic complexity of the concept learning task, but also to the full memory approach that most learning systems adopt. Indeed, during learning, most of these systems consider all their past examples leading to expensive procedures for consistency verification. In this paper, we present an implementation of a partial memory approach through an advanced data storage framework and show through experiments that great savings in learning times can be achieved. We also propose and experiment different ways to select the past examples paving the way for further research in on-line partial memory learning agents.

1 Introduction

Inductive concept learning concerns the inference of hypothesis from examples. There are two ways to process the examples: batch or incremental. The learning systems that adopt the batch paradigm simply store all the past examples and at each new incoming example just re-apply the method. This brings several disadvantages such as memory requirements for very large datasets, high computational learning times and difficulty in dealing with changing concepts over time. For these reasons, efforts have been made to built incremental learners that avoid the harmful effects of the batch approach. Incremental systems generally have lower memory requirements, lower learning time and are able to deal much better with drifting concepts.

However, in the incremental learning model the examples are distributed over the time, hence a good storage technique is of vital importance for the system efficiency. Incremental learning paradigm can be implemented using one of three memory models: *no instance memory*, *partial instance memory*, and *full instance memory*. In the *full instance memory*, such as IB1 [1], the system retains all the previous examined examples. In the *partial instance memory*, see for example FLORA [2], LAIR [3], HILLARY [4], DARLING [5], AQ-PM [6], the system memorizes only part of the past examples. Finally, the *no instance memory* model consists in

forgetting the examples as soon as the learning step is completed. Examples of such systems are Weighted Majority [7], Winnow [8], Stagger [9], Arch [10].

As pointed out in [11] there are three main reasons for studying partial memory learners. First, humans when learn from experience, store not only concept descriptions but also particular significant events and they may remember these events forever even though many new other events may happen. Second, the machine learning community has not dedicated much work to on-line partial memory systems. In fact, almost all learning systems either store all the encountered examples or none of them. Third, partial memory learners react more quickly to drifting concepts.

A fundamental issue for partial memory approach is the choice of the *representative* instances to memorize. In the partial memory incremental learning approach exploited in AQ-PM [6], the learned concepts are used to determine which training examples establish the outer bounds of a concept. The representative positive examples of a concept are those examples that lie at the boundaries of the concept in the event space, while the representative negative examples are those that constrain the concept in the event space (all other training examples can be discarded). Specifically, after the initial concepts have been learned they are used by the system for inference. If the system makes a wrong decision, then this is a signal that the concepts require refinement. The misclassified training example is included within the existing representative examples and inductive learning takes place. Once new concepts are learned, the training examples are evaluated and those determined to be representative are selected to form the new set of representative examples. The new concepts are used for inference and the new representative training examples are stored. This process repeats indefinitely.

Other systems that exploit the *partial instance memory* are: LAIR [3], that retains all the first positive examples available; HILLARY [4], that memorizes the negative instances only; DARLING [5] that associates a weight equal to 1 to each new incoming instance decreasing the weight of the past instances according to a neighbor policy and eliminating the instance from the memory as soon as the associated weight is lower than a threshold.

In this paper we focus on incremental systems that refine theories with a partial memory approach. The sub-field of machine learning where our work is positioned is that of learning logic programs from examples [12], i.e. concepts and examples are described in logic representation and in particular in first-order Horn clauses. The goal is to transform the multi-relational inductive learning system INTHELEX [13] so that it can adopt a partial memory approach. We also want to show through experiments that the partial memory implementation in learners of logic programs and systems that refine theories, can drastically reduce learning times.

To the best of our knowledge, this is the first attempt to implement a partial memory approach in a logic-based theory refining system. The most similar system is GEM-PM [11] that is an incremental learning system with partial memory that employs both refinement operators. However, the concept description language is not based on first-order logic such as INTHELEX language. Since GEM-PM is based on the AQ learning algorithm [14], it learns propositional concept descriptions, while INTHELEX learns first-order concept descriptions. Other systems such as FORTE [15] refine theories in a first-order setting but do not employ partial memory during the refinement process. Most of other work in the first-order setting [12] is related to

batch or incremental learning systems that either do not refine theories or do not employ partial memory.

The paper is organized as follows. Section 2 presents the field of inducing concept descriptions as logic programs and the incremental multi-relational learning system INTHELEX. Section 3 presents the advanced data storage framework through which the partial memory is implemented. Section 4 describes how the partial memory is implemented. Section 5 presents experiments and Section 6 concludes and presents future work.

2 Inducing Concept Descriptions as Logic Programs

First-order logic provides a theoretically robust background to reasoning and a powerful knowledge representation framework. Logic-based formalisms' semantics and their descriptive power have been exploited to induce concepts and reason about them in domains with structural and relational requirements. One of the fields that have witnessed much work is that of multi-relational learning [12]. In this research area, concepts are described as first-order Horn clauses which represent a sub-set of pure first-order logic. The learning task consists in inducing concepts given a set of positive and negative examples in the presence of an eventually available background knowledge. A concept is usually intended as a learned theory in the form of *if...then...* rules that compose it. The main operators in the learning task are those that generalize and specialize a certain rule in the theory. When a new positive example is not explained by the theory then the generalizing operator starts a search for a hypothesis (in the space of possible hypothesis usually ordered by θ -subsumption) that can explain the example. If a new coming negative example is explained by the theory, then the specializing operator identifies the responsible rule for explaining the example and tries to specialize it by searching the candidate hypothesis in the subsumption lattice. Usually multi-relational learning systems adopt one of the two operators to search the space of hypothesis. They, either start from the most possibly general hypothesis and try to specialize it (top-down) or from the most specific hypothesis and generalize it (bottom-up). Involving both operators is known as theory refinement [16] and is considered to be a very hard task.

There are three issues of multi-relational learning to be considered for the purposes of this paper. First, we are interested in the way examples are stored and considered for future learning processes. Second, how the learning process is performed, batch or incremental. Third, how the hypothesis space is searched. The current learning system INTHELEX that we refer to in this paper adopts a full memory approach, i.e. it maintains all the past positive and negative examples and whenever a hypothesis must be evaluated during search, it must be checked against all the past examples. INTHELEX is an incremental system, i.e. learning can start from scratch or from an existing theory. In the latter case, the existing theory can be generalized or specialized according to the arriving examples. Thus, the system implements refining operators. Moreover it can learn hierarchical concepts, when a concept depends on another concepts, it interrupts the search for the hypothesis for the main concept and starts searching for hypothesis for the composing concept.

To the best of our knowledge, there have not been other works on refining first-order logical theories through partial memory approaches. This is due to the fact that not many refining systems exist due to the high complexity of the refinement task. In the following we sketch the main algorithms in INTHELEX and show how the partial memory approach can be implemented.

*Procedure Generalize (E: positive example; T: theory;
M: set of all negative examples)*

```

L := Take all clauses whose concept is the same as that of E.
Consistency = false
While there are clauses in L do
  Take a clause C from L
  While (Consistency = false and there are no more generalization for C) do
    Candidate_Hypothesis = Generate a generalization of C and E
    Consistency = Check_Consistency(Candidate_Hypothesis,M)
  End While
End While
End Procedure

```

*Procedure Specialize (E: negative example; T: theory;
M: set of all positive examples)*

```

L := Take all clauses from T that participate in the derivation of the negative
example E.
Completeness = false
While there are clauses in L do
  Take a clause C from L
  While (Completeness = false and there are no more specialization for C) do
    Candidate_Hypothesis = Generate a specialization of C and E
    Completeness = Check_Completeness(Candidate_Hypothesis,M)
  End While
End While
End Procedure

```

The most expensive procedures in theory refinement are those that check the consistency and completeness towards the past examples. When a positive example is not covered by the theory the procedure *Generalize* starts a search in the space of hypothesis by generating and testing each candidate hypothesis' consistency towards all the past negative examples. Since the hypothesis that can be generated are too many, checking every candidate towards all the past examples becomes a bottleneck. The idea of the partial memory is to check the consistency only against a part of the

negative examples. On the contrary, when a negative example is covered, the theory must be specialized. The procedure *Specialize* greedily generates specializations each of which must be checked for completeness against all the past positive examples. Again, this becomes overwhelming in terms of computational times and if we are unlucky we may check for completeness $n-1$ candidates on all the available examples and then find that the n -th is the only one that is complete. With the partial memory, the number of examples to check is lower and, as we will show in the next sections, this can bring significant savings in time at no significant changes in the learning accuracy.

3 The Berkeley DB Framework and the Prolog API

BDB (Berkeley DB) [17] is an advanced framework for data management. It is an open-source project whose goal is that of producing a database library able to provide high performance data access and highly efficient data management services. It was initially developed at the University of California at Berkeley and later at University of Harvard. BDB is a tool for software developers, the only way to use it is to write code. There is no standalone server and no SQL query tool for working with BDB databases. It provides a powerful Application Programming Interface (API) that permits to develop fast and reliable embedded solutions. The basic structures of BDB are B-tree, hash table and persistent queues. Over these structures BDB offers advanced services such that multiple threads or processes can operate on the same collection of B-trees and hash tables at the same time without risk of data corruption or loss. One of the most appealing feature of BDB is the ease of deployment. The API supports many programming languages such as JAVA, C, C++ etc, so that it can be easily embedded in end-user or infrastructure applications. Moreover, since the database engine is literally in the same address space as the application, no database access needs to cross a process or network boundary. In many relational DBMSs, context switch time is an important bottleneck, and also moving data across network boundaries on different machines is even more expensive. BDB eliminates both these disadvantages. A comparison of BDB towards relational DBMSs can be found in [18].

For our purposes in this paper we are interested in some features of a data storage system. In particular, we refer to the Prolog API for BDB provided by SICStus [19]. The most important feature for storing logic terms is the possibility to store logic variables. Most DBMSs are not able to store non-ground terms or more than one instance of a term. The Prolog API for BDB permits to store variables with blocked goals as ordinary variables. This is a powerful feature in a logic-based programming system and since INTHELEX is a learning system written in Prolog we can use the above data storage Prolog API for a partial memory approach. However, as stated above, BDB does not offer a SQL-like query language. For every data management service, wrapping code must be written to provide data access. We have developed a wrapping layer in the Prolog language to implement a partial memory solution for the system INTHELEX.

In the following we describe the Prolog API of SICStus. This interface exploits the Concurrent Access Methods product of BDB. This means that multiple processes can open the same database. The environment and the database files are ordinary BDB entities that use a custom hash function. The idea is to obtain a behavior similar to the built-in Prolog predicates such as `assert/1`, `retract/1` and `clause/2`, but having the terms stored on files instead of main memory. Some differences with respect to the Prolog database are:

- The functors and the indexing specifications of the terms to be stored have to be given when the database is created.
- The indexing is specified when the database is created. It is possible to index on other parts of the term than just the functor and first argument.
- Changes affect the database immediately.
- The database will store variables with blocked goals as ordinary variables.

The db-specification (*db-spec*) defines which functors are allowed and which parts of a term are used for indexing in a database. The *db-spec* is a list of atoms and compound terms where the arguments are either + or -. A term can be inserted in the database if there is a specification (*spec*) in the *db-spec* with the same functor. Multilevel indexing is not supported, terms have to be “flattened”. Every *spec* with the functor of the indexed term specifies an indexing. Every argument where there is a + in the *spec* is indexed on.

A *db-spec* has the form of a specification-list (*speclist*):

$$\begin{aligned} \text{speclist} &= [\text{spec1}, \dots, \text{specM}] \\ \text{spec} &= \text{functor}(\text{argspec1}, \dots, \text{argspecN}) \\ \text{argspec} &= + \mid - \end{aligned}$$

where *functor* is a Prolog atom. The case $N = 0$ is allowed. A *spec* $F(\text{argspec1}, \dots, \text{argspecN})$ is applicable to any ground term with principal functor F/N .

When storing a term T , it is generated a hash code for every applicable *spec* in the *db-spec*, and a reference to T is stored with each of them. (More precisely with each element of the set of generated hash codes). If T contains ground elements on each + position in the *spec*, then the hash code depends on each of these elements. If T contains some variables on + position, then the hash code depends only on the functor of T . When fetching a term Q we look for an applicable *spec* for which there are no variables in Q on positions marked +. If no applicable *spec* can be found a domain error is raised. If no *spec* can be found where on each + position a ground term occurs in Q an instantiation error is raised. Otherwise, we choose the *spec* with the most + positions in it breaking ties by choosing the leftmost one. The terms that contain ground terms on every + position will be looked up using indexing based on the principal functor of the term and the principal functor of terms on + positions. The other (more general) terms will be looked up using an indexing based on the principal functor of the term only.

4 Implementing Partial Memory in INTHELEX

In a recent work [20] the authors presented the design and implementation of an external storage solution of logic terms by integrating BDB in INTHELEX through the SICStus Prolog API. It was shown that this upgrade resulted in much better performance against the old version of INTHELEX that used only the internal memory of the SICStus Prolog interpreter. In this solution, logic terms were stored in BDB instead of being loaded in main memory. Moreover, it was implemented a relational schema linking every example to the clause that explains it. This helps a lot in the theory refinement process because permits to check the candidate clause refinements only against the examples already covered by the clause that is being refined. Experiments in [20] showed that using the fast access of BDB and the relational schema, outperformed the old version of the system in terms of learning time.

In Figure 1 it is shown the schema implemented in [20]. In the following we briefly explain the schema. It makes possible to link clauses to examples that are covered by these clauses. Whenever one of these clauses is refined, the relational schema permits a fast access only to those examples that are related to the clause. We chose to include examples in a database and clauses in another. We maintained two separate tables for examples and descriptions in order to have the possibility to preserve useful information about descriptions that can be used either afterwards to assess and study the learning process or during the learning task for accelerating the access to data. We also designed a table with only two fields that serves as a hash table to fetch all the examples covered by a clause. In fact, the two fields represent respectively the key of the clause and the key of the example. Another table was introduced in order to maintain all the versions of the theory during the refinement steps. This could be very useful for the learning task and the previous versions of the theory could be exploited for improving the searching on the space of the hypothesis. Regarding the information about the examples, we keep useful statistics about the examples and the observations.

We can also group the examples in order to distinguish between those that have been given in input to the system at a time and those that have been given at another moment. This might be useful to study the learning task from an *example grouping* point of view, trying to analyse and distinguish those examples that have a greater impact on the learning task. As regards the clauses, we keep other information such as the generating example of the clause. This represents another precious fact that could be useful in understanding better the learning process.

The *db-spec* that results from the schema in Figure 1 is simple. The following specifications show how the tables are implemented through the *db-spec* of BDB. As it can be seen, the attributes which are marked with “+”, represent the keys of the clauses and examples, thus the database is indexed on these attributes.

```
[ examples(+,-,-,-,-), obs(+,-,-,-), max(+,-,-,-) ]
[ clauses(+,-,-,-,-,-,-,-), theory_changes(+,-,-,-,-), clauEx(+,-), pos_except(+,-,-),
neg_except(+,-,-), max(+,-,-,-,-,-,-,-)]
```

We have used the indexed attributes for our purpose of implementing a relational schema between clauses and examples but however the database engine of BDB uses these attributes for efficient fetching of the terms. This is due to the schema adopted by this database engine whose fetching mechanism is based on the indexed terms.

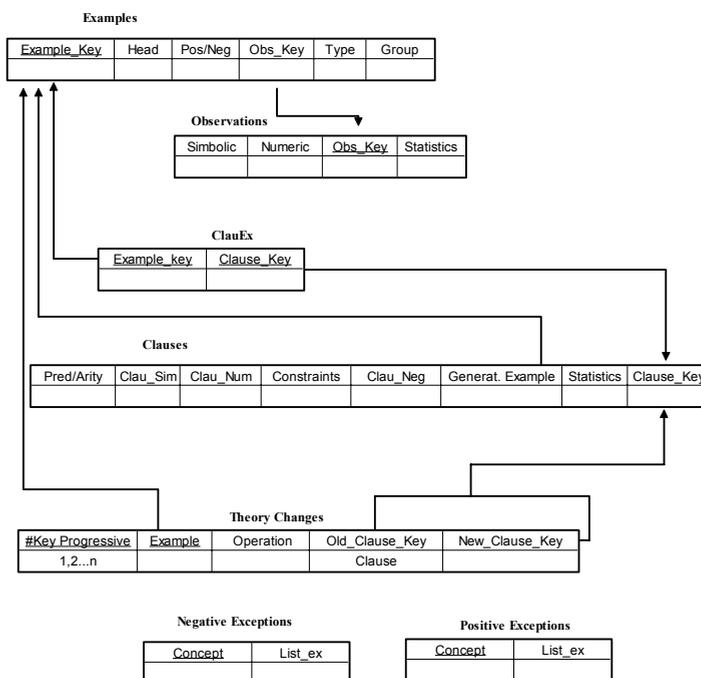


Figure 1. Linking examples to clauses and partial memory solution through the attributes *Example_Key*, *Type* and *Group*.

Concerning the implementation of the partial memory strategy, for each example we kept a progressive key, *Example_Key*, associated to each single training instance after being processed. Furthermore, the *Type* attribute was associated to each example to distinguish the observations that caused a change in the theory. With this information, the refinement operators could consider the last n observations in the order they arrived to the system, or take into account the last n observations among those that generated a theory refinement when they were considered by the system. Thus, there are different options for implementing the partial memory by simply modifying the procedures *Check_Consistency* and *Check_Completeness* of the refining operators of Section 2 such that only a portion of the past examples is considered for consistency and completeness checking.

5 Experimenting Partial Memory

The test of the performance of the system when the partial memory functionality is activated concerned the classification and understanding task on scientific paper documents [13]. In particular the classification task was performed on the class of documents belonging to three different series ICML (28- the International Conference on Machine Learning), SVLN (32- Springer Verlag Lectures Notes) and IEEEET (36 – IEEE Transactions on Pattern Analysis and Machine Intelligence). Each positive example for a concept target was considered negative for all the other concepts to be learned. The understanding task was performed on the *Author* [36+, 332-], *Page Number* [27+, 341-] and *Title* [29+, 339-] logical components of the ICML class of documents. The dataset was randomly split 33 times in a Tuning set, containing 70% of the examples, and a Test set, containing 30%, both guaranteeing a fair proportion of positive and negative examples for each class. We combined different alternatives for the partial memory and modified the procedures Check_Completeness and Check_Consistency presented in Section 2. Initially three kinds of experiments were carried out with the partial memory: **1.** in the first one approximately the last 25% of the past observations were considered; **2.** in the second experiment only the observations that modified the theory were considered; **3.** in the last experiment we took into account the observations that modified the theory among the last 25% of all the past observations. All the experiments were performed on a Pentium 4, 2.4 GHz, using SICStus Prolog 3.11.1.

As shown in Tables 1 and 2, that report the mean predictive accuracy and mean learning time in seconds for the 33 fold, in all the experiments both for the classification and understanding process, the execution time decreases significantly; as regards the percentage of predictive accuracy, there is no significant difference in the case of full memory or partial memory with the option **1**, thus we could accurately learn the theory considering only a part of the learning set. However, with the partial memory options **2** and **3** the accuracy decreases significantly.

Table 1. Classification with Partial Memory Functionality

	ICML		SVLN		IEEEET	
	Acc. %	Time(s.)	Acc. %	Time(s.)	Acc.%	Time(s.)
Full Memory	97,75	3,15	87,36	19,45	90,57	27,55
Partial Memory						
1	97,36	1,73	87,57	6,63	88,75	6,13
2	89,84	1,45	84,63	2,66	84,39	3,04
3	89,48	1,33	81,00	2,81	80,27	2,60
4	96,18	1,95	83,90	5,25	87,39	6,9

Table 2. Understanding with Partial Memory Functionality

	<i>Author</i>		<i>Page Number</i>		<i>Title</i>	
	Acc. %	Time(s.)	Acc. %	Time(s.)	Acc.%	Time(s.)
Full Memory	97,12	29,07	97,54	76,22	97,87	51,66
Partial Memory						
1	96,87	10,27	97,42	27,17	97,96	19,65
2	94,54	3,09	90,06	5,58	94,69	3,25
3	88,24	3,36	94,06	5,77	94,51	2,61
4	97,06	4,95	96,33	12,11	96,81	13,19

We performed a thorough analysis of the results with the partial memory options **2** and **3**. We discovered that on some of the 33 folds the predictive accuracy decreased drastically and this influenced the overall accuracy on the 33 folds. The reason of the decrease was that in these folds there were no negative examples that caused changes to the theory and thus the system refined the theory considering only positive examples that modified the theory. This resulted in over-generalization of the theory and in the test phase many negative examples were covered by the theory. Therefore, the natural solution to this problem was to include negative examples in the set of examples that represent the partial memory. To do this we initially included in the partial memory only the first negative example encountered. However, this did not change the situation and the predictive accuracy continued to be low. At this point, we empirically discovered that if there were some negative examples, also very few, the predictive accuracy did not decrease. For this reason, we decided to adopt the following solution. We decided to maintain in the partial memory a certain number of negative examples that constituted at every moment of the learning task a certain percentage of the overall number of examples. The procedure that updated the partial memory kept constantly at every arrival of each example, a set of negative examples. Among these, there were examples that had caused modifications of the theory and others that had not. For instance, if at a certain moment of the learning task were processed 100 examples, and there were two negative examples that had caused theory refinements and the procedure for updating the partial memory had received 5% as parameter for the percentage of negative examples, then the partial memory contained totally 5 negative examples. The remaining three negative examples were randomly chosen from those that did not modify the theory.

Due to the different number of available observations for each dataset, the percentage of negative observations was 10% for the classification process and 2% for the understanding one. The results reported in Tables 1 and 2, option **4**. for the partial memory, show an improvement of the predictive accuracy towards option **2** and **3**. From all the solutions for the partial memory, the combination of the modifying examples with a certain fixed percentage of negative examples, resulted the best strategy.

The experimental results suggest that there exists a certain number of examples that can guide the learning task and can avoid over-generalizations or under-

specializations. Therefore, it is not necessary to continue refining the theory against all the past examples if the same accuracy but at much lower time is achieved through the partial memory. It also seems highly likely that among the negative examples, there exists a sufficient set whose informative contents seems to optimally conduct the learning process. Distinguishing among the negative examples that do not modify the theory, those that more than others influence the learning task is an interesting task. In this paper, we choose them randomly to always keep in the partial memory a constant percentage of negative examples. A non-random selection would require a principled criteria. We believe information-theory based criteria such as information gain can be used. We plan to investigate this in the future.

6. Discussion and Future Work

We have implemented a partial memory solution to incremental learning and theory refinement and shown through experiments that the partial memory approach outperforms the traditional one in terms of learning times at no significant changes in predictive accuracy. The approach is implemented through an advanced term storage framework such as BDB and the relative Prolog API that is able to store variables. We have experimented various strategies for the partial memory showing that a small number of examples can guide the learning process towards optimal generalizations and specializations. Identifying this small set of examples is not always a straightforward task. In this paper, we have shown that combining a set of positive examples that modify the theory with a small number of negative examples, produces an enough training set to achieve very good results.

To the best of our knowledge, this is the first attempt to implement a partial memory approach in a logic-based theory refining system. The most similar system implementing an incremental approach with partial memory and that employs both refinement operators is GEM-PM [11] that, however learns propositional concept descriptions, while INTHELEX learns first-order concept descriptions. Most of other work in the first-order setting [12] is related to batch or incremental learning systems that either do not refine theories or do not employ partial memory.

The experiments in this paper for a refinement task, suggest that theory refinement in first-order settings can highly benefit from partial memory approaches leading to significant time savings at no significant loss of accuracy. As future work, we plan to investigate how we can identify among the modifying examples those that are more significant, i.e. examples that cause different refinements in the theory and, most probably, the ones that cause the most substantial changes are those that must be kept for future use in the learning task. Again, the implemented schema with BDB can be very useful, because it permits to keep every intermediate change in the theory and the example that caused it. In this way, we can weight an example based on its history of theory refinements and thus evaluate the “importance” of each example for the learning process.

References

1. Aha. D., Kibler. D., Albert. M. Instance-based learning algorithms, *Machine Learning* 6 37–66, (1991).
2. Widmer. G., Kubat. M. Learning in the presence of concept drift and hidden contexts, *Machine Learning* 23, 69–101 , (1996).
3. Elio. R., Watanabe. L. An incremental deductive strategy for controlling constructive induction in learning from examples, *Machine Learning* 7 7–44, (1991).
4. Iba. W., Woogulis. J., Langley. P. Trading simplicity and coverage in incremental concept learning, in: *Proceedings of the Fifth International Conference on Machine Learning*, Morgan Kaufmann, San Francisco, CA, pp. 73–79, (1988).
5. Salganicoff. M., Density-adaptive learning and forgetting, in: *Proceedings of the Tenth International Conference on Machine Learning*, Morgan Kaufmann, San Francisco, CA, pp. 276–283, (1993).
6. Maloof, M. Michalski. R. Selecting examples for partial memory learning, *Machine Learning* 41, 27–52. (2000).
7. Littlestone. N., Warmuth. M. The Weighted Majority algorithm, *Inform. and Comput.* 108 212–261, (1994).
8. Littlestone. N. Redundant noisy attributes, attribute errors, and linear-threshold learning using Winnow, in: *Proceedings of the Fourth Annual Workshop on Computational Learning Theory*, Morgan Kaufmann, San Francisco, CA , pp. 147–156, (1991).
9. Schlimmer. J., Granger. R. Beyond incremental processing: Tracking concept drift, in: *Proceedings of AAAI- 86*, Philadelphia, AAAI Press, pp. 502–507, (1986).
10. Winston. P. Learning structural descriptions from examples, in: P. Winston (Ed.), *Psychology of Computer Vision*, MIT Press, Cambridge, MA, (1975).
11. Maloof. M., Michalski. R. S. Incremental learning with partial instance memory. *Artificial Intelligence* 154, pp. 95-126, (2004).
12. Dzeroski. S., Lavrac. N. (Eds) *Relational Data Mining*. Springer, Berlin, (2001).
13. Esposito. F., Ferilli. S., Fanizzi. N., Basile. T.M.A., Di Mauro. N. Incremental Multistrategy Learning for Document Processing, *Applied Artificial Intelligence: An International Journal*, Vol. 17, n. 8/9, pp. 859-883, Taylor Francis, (2003).
14. Michalski. R. On the quasi-minimal solution of the general covering problem, in: *Proceedings of the Fifth International Symposium on Information Processing*, vol. A3, pp. 125–128, (1969).
15. Richards. B. L., Mooney. R.J. Refinement of first-order horn-clause domain theories. *Machine Learning Journal*, 19(2):95–131, (1995).
16. Mooney. R.J. Batch versus Incremental Theory Refinement. *Proceedings of the 1992 AAAI Spring Symposium on Knowledge Assimilation*, Standford, CA, (1992).
17. Berkeley DB reference: <http://www.oracle.com/database/berkeley-db>.
18. Oracle white paper: A Comparison of Oracle Berkeley DB and Relational Database Management Systems, <http://www.oracle.com/database/docs/Berkeley-DB-v-Relational.pdf>.
19. SICStus Prolog reference: <http://www.sics.se/sicstus>
20. Biba. M., Basile. T.M.A., Ferilli. S., Esposito. F. Improving Scalability in ILP Incremental Systems. *Proceedings of CILC 2006 - Italian Conference on Computational Logic*, Bari, Italy, June 26-27, (2006).

Un Protocollo di Sicurezza contro le Vulnerabilità derivanti da Software di Firma non Trusted

Francesco Buccafurri and Gianluca Lax

DIMET, Università degli Studi Mediterranea di Reggio Calabria
Via Graziella, Località Feo di Vito, 89122 Reggio Calabria, Italy
bucca@unirc.it, lax@unirc.it

Sommario La sicurezza della firma digitale si basa sulla robustezza degli algoritmi di crittografia utilizzati, sulla adeguata lunghezza delle chiavi, nonché sull'uso di un dispositivo sicuro per la sua generazione, che garantisca la segretezza della chiave privata in ogni momento dell'elaborazione. Sfortunatamente, la promiscuità nell'uso dei personal computer impiegati per generare la firma su documenti informatici non consente di considerare *trusted* il software di creazione della firma, ancorchè originale e fornito direttamente dall'ente certificatore. In questo scenario, si origina una seria vulnerabilità dei sistemi di creazione di firma digitale, sfruttando la quale software "malicious" potrebbe firmare, all'insaputa del titolare, documenti non di interesse dello stesso. Rispetto a tale vulnerabilità, recentemente documentata in letteratura, l'utilizzo dei dispositivi sicuri di firma non offrono, al momento, alcuna protezione. In questo lavoro definiamo un protocollo di sicurezza che contrasta la tipologia di attacco sopra descritta, avente reale applicabilità grazie all'utilizzo delle Java Card, già ampiamente diffuse nel mercato e compatibili con le specifiche di sicurezza richieste nel contesto della firma digitale.

1 Introduzione

La firma digitale è una procedura informatica, basata su crittografia a chiavi asimmetriche, che, applicata ai documenti informatici, permette di garantirne *autenticità* e *genuinità*. La firma è il risultato della cifratura (attraverso un algoritmo di crittografia asimmetrico) dell'hash (calcolato con una funzione hash *one-way*) del documento da firmare, realizzata attraverso l'uso della chiave segreta (posseduta esclusivamente dal titolare della firma). La firma è in tal modo collegata al contenuto del documento, cioè al valore numerico corrispondente alla sua codifica binaria e, pertanto, è superato il problema dell'immaterialità del documento informatico, che non permetterebbe di legare la firma al supporto così come avviene nel caso dei documenti tradizionali. La chiave pubblica corrispondente alla chiave segreta usata per la sottoscrizione potrà essere usata per la verifica della firma e l'associazione all'identità del titolare verrà realizzata attraverso l'intervento di una *terza parte fidata*, rappresentata dal *soggetto*

certificatore. Attraverso la garanzia di segretezza della chiave privata di sottoscrizione, i meccanismi di certificazione della chiave pubblica e la robustezza degli algoritmi hash e di crittografia utilizzati, è possibile, firmando digitalmente documenti elettronici, realizzare efficacemente le funzioni *indicativa* (chi è l'autore della firma), *dichiarativa* (cosa si sottoscrive) e *probatoria* (costituzione di un *contrassegno*, cioè di una prova critica precostituita), alla stessa stregua di quanto realizza la firma autografa per i documenti cartacei, e ottenere pertanto il requisito del *non ripudio*. Bisogna certamente considerare, tuttavia, che l'equipollenza della firma digitale, rispetto alla firma autografa e, quindi, il suo grado di efficacia probatoria, dipende strettamente dai requisiti di sicurezza con cui la firma stessa è realizzata. Lunghezza delle chiavi, algoritmo di crittografia, algoritmi hash, dispositivi utilizzati per la sua generazione, eventuale presenza di macrocodice o codice eseguibile nei documenti sottoposti alla firma, affidabilità ed autorevolezza del soggetto certificatore, sono tutti elementi che concorrono, in diversa misura, a modificare il grado di sicurezza della firma, e, di conseguenza, la sua reale efficacia probatoria. Il legislatore italiano, in accordo a quanto stabilito anche dall'UE, ha pertanto introdotto diversi tipi di firma, che differiscono per grado di robustezza, in maniera da includere, sotto il concetto di firma, anche procedure che non siano basate necessariamente sulla crittografia asimmetrica. In questo articolo, ci riferiamo tuttavia al concetto di *firma forte*, così come individuato dal legislatore europeo, recepito dalla normativa italiana proprio con la dizione giuridica di *firma digitale*. In altri termini il nostro lavoro si colloca in tutti quei contesti nei quali i requisiti di robustezza e sicurezza della firma assumono un ruolo particolarmente rilevante (per esempio, *e-government*). Contesti nei quali, cioè, eventuali debolezze della firma sono da ritenersi assolutamente intollerabili.

L'approccio seguito comunemente per garantire un adeguato livello di robustezza, approccio seguito anche dal legislatore sia a livello comunitario che a livello italiano, è basato sulla definizione della lunghezza minima delle chiavi, su restrizioni circa gli algoritmi di crittografia e le funzioni hash utilizzabili, sull'uso di dispositivi di firma esterni dedicati, detti *sicuri*, come la *Smart Card* o il *token USB*, su restrizioni circa il ruolo di eventuale macro-codice o codice eseguibile nei documenti sottoposti alla firma. Il tutto è fatto sotto l'assunzione che il software di generazione della firma (tipicamente fornito dal certificatore), il sistema operativo del PC su cui si esegue tale software, il sistema operativo e il software del dispositivo esterno utilizzato, siano tutti *trusted*. Che non implementino cioè nessun comportamento *malicious*.

Tale assunzione non è del tutto realistica, soprattutto per quello che riguarda il PC, non essendo pensabile che il PC su cui si esegue la firma di documenti sia dedicato solo a tali attività, isolato e, pertanto, protetto da eventuali attacchi informatici basati su software *malicious*. In altre parole, anche se è ragionevole assumere che la *Smart Card* sia una piattaforma *trusted*, bisogna ricordare che la *Smart Card* stessa è un computer *handicapped* [20], in quanto privo di I/O. Se il PC che si interfaccia con la *Smart Card*, allo scopo di fornire le funzionalità di I/O necessarie, non è *trusted*, non c'è modo di garantire la sicurezza del processo

di firma. Il problema, noto in letteratura¹ [10,22,1], non ammette quindi soluzioni costruite sul piano teorico.

L'approccio più significativo proposto in letteratura per mitigare questo problema si basa sulle *conditional signatures* [2], attraverso le quali la validità della firma effettuata dall'utente è subordinata alla verifica a-posteriori, da parte del sottoscrittore, del documento effettivamente firmato attraverso un terminale "sicuro".

L'approccio che invece viene seguito nel presente lavoro è meramente applicativo. Il protocollo qui sviluppato si basa cioè sull'utilizzo di specifiche tecnologie. Concettualmente esso tende a fortificare l'affidabilità del canale di comunicazione tra Smart Card e PC, utilizzando un canale di controllo autonomo, basato su tecnologie differenti (esecuzione di Applet Java), tale da rendere pragmaticamente più difficile che si realizzi una compromissione coordinata dei due canali. Tale considerazione si basa anche sul fatto che il canale alternativo è implementato attraverso Applet Java mantenute sulla piattaforma trusted, la cui autenticità può essere garantita attraverso gli usuali meccanismi utilizzati nel contesto del *code mobility*.

In particolare, viene proposta una metodologia basata sul calcolo serializzato dell'impronta hash del documento da firmare, e sull'uso della Java Card (in luogo della tradizionale Smart Card) per contrastare l'attacco sopra descritto. L'idea di fondo è dare alla Java Card, oltre al ruolo di generatore della firma, anche quella di controllore della validità del documento da firmare, e abilitare la prima funzione (cioè quella della apposizione della firma) solo dopo che il controllo è stato effettuato con successo. La metodologia è stata implementata utilizzando un simulatore di Java Card ed è stata ampiamente testata. Il contributo del presente lavoro è duplice. Da una parte si è proposto un metodo concreto per la difesa da un tipo di attacco alla firma digitale particolarmente insidioso, rendendo quindi il protocollo di firma più robusto. Dall'altra si è mostrato come l'impiego delle Java Card in luogo delle tradizionali Smart Card, programmabili solo all'origine, apre scenari applicativi molto vasti che possono chiaramente andare oltre quello della robustezza qui presentato.

2 Debolezze della Firma Digitale

In questa sezione presentiamo l'"Attacco tramite Trojan" alla firma digitale. Per comprenderne pienamente il meccanismo di funzionamento è necessario descrivere brevemente come un utente, dotato di Smart Card e di un apposito software di firma residente su un personal computer, firma un documento digitale. Tale procedura è descritta in Figura 1.

¹ Esiste anche un recente attacco costruito dal gruppo di lavoro del Prof. Bruschi del Dipartimento di Informatica e Comunicazione dell'Università degli Studi di Milano basato su questa debolezza, che permette ad un attacker di ottenere documenti firmati digitalmente all'insaputa del legittimo titolare della firma, tramite un software malicious che prende il controllo del software di firma.

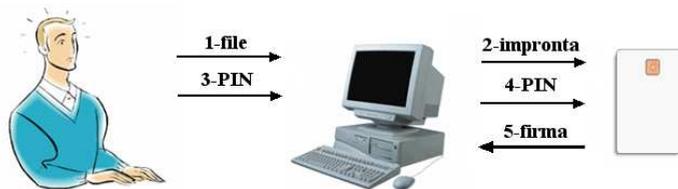


Figura 1. Procedura per la firma di un documento.

In primo luogo l'utente, attraverso il software di firma, seleziona il file contenente il documento da firmare. A questo punto il software calcola una opportuna *funzione hash* (in genere SHA-1 [17] o RIPEMD-160 [7]) del documento ottenendone così l'*impronta*. La funzione hash ha lo scopo di creare, a partire dal documento, una sequenza di bit di lunghezza fissata (tipicamente 160 bit) tale che sia praticamente impossibile trovare due documenti differenti aventi la stessa impronta. Un'ulteriore caratteristica della funzione hash è di essere *one-way*, per cui dalla conoscenza della sola impronta del documento non deve essere possibile ricavare l'intero documento o parti di esso. L'impronta così calcolata dal software di firma viene inviata alla Smart Card. Quest'ultima, abilitata dall'utente attraverso l'immissione del PIN, esegue un opportuno algoritmo di crittografia asimmetrica (tipicamente l'algoritmo RSA [19]) sull'impronta utilizzando la chiave segreta dell'utente, e generando la firma del documento. Tale firma è intrinsecamente legata al documento, perchè l'algoritmo di crittografia opera sull'impronta del documento, e all'utente, in quanto un parametro dell'algoritmo di crittografia è la chiave segreta dell'utente.

Gli algoritmi di crittografia asimmetrica godono della seguente proprietà: ciascun dato cifrato con la chiave segreta può essere decifrato con una opportuna chiave, detta chiave pubblica, il cui valore è legato a quello della chiave segreta. Questa proprietà viene sfruttata per verificare la validità della firma. Inoltre vale anche il viceversa, cioè che è cifrato con la chiave pubblica può essere decifrato con la chiave segreta corrispondente. La robustezza degli algoritmi di crittografia asimmetrici si basa sul fatto che la determinazione della chiave segreta a partire dalla conoscenza della chiave pubblica è un problema computazionalmente ritenuto non trattabile [4].

Infine, la firma così generata viene inviata dalla Smart Card al software di firma che provvede a costruire la *busta crittografica*, un documento elettronico che tipicamente segue lo standard *PKCS#7* [14] e contiene le informazioni necessarie a verificare la validità della firma, quale il documento firmato, la firma, il riferimento all'algoritmo di crittografia e alla funzione hash utilizzati, e il *certificato* dell'utente. Il certificato dell'utente è un documento elettronico che stabilisce l'associazione tra chiave pubblica e soggetto titolare del certificato. Il certificato viene emesso da un soggetto certificatore, secondo standard internazionali (tipicamente lo standard X.509 [18]) e viene firmato con la chiave privata del certificatore. Il certificatore ha anche il compito di pubblicare le liste dei certifi-

cati sospesi o revocati (dette CRL [11]), ad esempio relativi ad utenti che hanno smarrito la propria Smart Card.

La firma digitale non è esente da attacchi informatici. In questo lavoro concentriamo l'attenzione verso un attacco noto come "Attacco tramite Trojan" che consente ad un attacker di ottenere documenti firmati digitalmente all'insaputa del legittimo titolare della firma [10,22,1].

Infatti, nonostante sia garantita la robustezza degli algoritmi di crittografia e della funzione hash, la procedura di generazione della firma presenta una seria vulnerabilità. Essa risiede nel passaggio dell'impronta dal software di firma alla Smart Card, passaggio nel quale si perde il legame esplicito tra il documento e la sua impronta. Per questo motivo la Smart Card non conosce quale documento sta firmando, ma solo la sua impronta. Su questa debolezza è stato realizzato l'attacco tramite trojan. Il funzionamento dell'attacco è il seguente: un virus corrompe il software di firma trasformandolo in un software *malicious*. Ogni volta che l'utente legittimo firma digitalmente un documento, abilitando l'utilizzo della Smart Card attraverso il PIN, il virus è in grado di firmare (anche) altri documenti (le cui impronte sono inviate alla Smart Card) il cui contenuto è stato preventivamente deciso da chi ha realizzato il virus.

Si noti che il problema non è contrastabile adeguatamente agendo sull'immissione del PIN da parte dell'utente, e cioè richiedendone l'immissione, eventualmente mediante sistemi sicuri (come per esempio i metodi basati sull'approccio one-time-password [3,15]) per ogni azione di firma. Infatti non vi è un legame diretto tra PIN e hash da firmare, per cui rimarrebbe comunque la possibilità che l'hash inviato alla smart-card da firmare non sia quello voluto dall'utente.

Il presente lavoro si colloca nel contesto di questo attacco e propone un metodo per contrastarlo che si basa sulla possibilità offerta dalle nuove tecnologie, in particolare le Java Card descritte nella prossima sezione, che permettono ai dispositivi di firma di effettuare dei controlli opportuni prima di generare la firma.

3 Smart Card Multi-Applicazione

Le Smart Card sono carte di ridotte dimensioni che contengono all'interno un processore e della memoria. Vengono utilizzate in moltissimi ambiti e in particolare in quello della firma digitale. Infatti la memoria non volatile è utilizzata per la memorizzazione della chiave segreta e dei certificati digitali, mentre il processore permette l'esecuzione in-loco dell'algoritmo di crittografia utilizzato per la firma, evitando quindi che la chiave segreta, necessaria per la generazione della firma, abbandoni la Smart Card e possa così essere compromessa la sua segretezza.

Uno dei pochi svantaggi che presenta l'utilizzo delle Smart Card è rappresentato dal tempo necessario per la loro produzione, in quanto ciascuna tipologia di Smart Card deve essere opportunamente progettata e realizzata per l'applicazione cui necessita. Infatti i programmi delle Smart Card vengono in genere implementati in linguaggio assembler e una volta salvati nella ROM della card

non possono essere modificati o aggiornati. Da questo scaturiscono gli alti costi e i lunghi tempi di produzione delle carte, che portano sul mercato le Smart Card con rilevante ritardo rispetto alle richieste.

Per ovviare a questi limiti, recentemente è stata proposta una nuova tipologia di Smart Card, le Java Card, che contengono accanto all'hardware standard delle Smart Card anche alcuni componenti quali: un sistema operativo con dei metodi nativi che forniscono funzionalità di base (gestione dell'I/O, allocazione della memoria, ed altro ancora) e una *Java Card Virtual Machine* (JCVM) in grado di interpretare il bytecode delle *Java Card Applet*, applicazioni scritte in linguaggio Java per essere eseguite su Java Card.

Proprio la possibilità di eseguire queste Java Card Applet rappresenta la grossa novità introdotta dalle Java Card: tali applicazioni sono implementate in Java (sebbene sia disponibile un sottoinsieme del linguaggio Java standard) e sono indipendenti dalla piattaforma, cioè possono essere eseguite su Java Card realizzate da costruttori diversi. Inoltre è possibile caricare su una Java Card numerose applet, anche successivamente alla consegna della stessa, così come modificare o rimuove applet in qualsiasi momento utilizzano semplici procedure.

Si potrebbe obiettare che l'utilizzo di Smart Card programmabili in firmware solo all'origine, cioè le Smart Card tradizionali, dia maggiore sicurezza in quanto impedisce la compromissione del dispositivo di firma. In effetti tale considerazione ha spinto inizialmente il legislatore italiano a definire *dispositivo di firma* un "apparato elettronico programmabile solo all'origine" [8]. Tuttavia i progressi della tecnologia in campo di sicurezza hanno fatto sì che il legislatore italiano, recependo la direttiva europea [5], abbia rimosso tale vincolo [9], permettendo così l'utilizzo anche di Smart Card programmabili per la generazione della firma digitale.

Infatti esistono Java Card [21] certificate ITSEC E6-high² che soddisfano pienamente i requisiti di sicurezza attualmente richiesti [9] e fissati al livello ITSEC E3-high.

4 Una Strategia di Difesa

La vulnerabilità del sistema di firma digitale risulta evidente in quanto tali sistemi funzionano sul computer dell'utente, che non sempre è gestito in modo sicuro. Un'assunzione che può essere realisticamente adottata è che non è possibile violare la protezione intrinseca del dispositivo di firma.

La realizzazione del nostro sistema di protezione si basa essenzialmente su quest'ultimo aspetto. L'idea che sta alla base del sistema è di utilizzare una Smart Card in grado di offrire *on-board* alcune funzionalità del software di firma, e restituire la firma digitale solo ed esclusivamente se il file da firmare è il file scelto dall'utente legittimo. Ovviamente una Smart Card standard non consentirebbe di effettuare on-board le funzionalità di un software di firma. Nasce

² L'Information Technology Security Evaluation Criteria (ITSEC) [12] è uno standard indipendente e rigoroso per la valutazione e la certificazione della sicurezza dei sistemi e dei prodotti informatici.

così l'esigenza di prendere in considerazione una Java Card, ovvero una Smart Card intelligente con funzionalità di crittografia. Il sistema da noi proposto introduce un passo aggiuntivo al protocollo di firma attualmente utilizzato, nel quale l'utente deve inviare alla Java Card l'intero documento e quest'ultima ne calcola l'impronta. Il resto del protocollo rimane invariato, per cui l'utente seleziona il file da firmare attraverso il software di firma che ne calcola l'impronta e la invia al dispositivo per ricevere successivamente la firma. A questo punto però, il dispositivo, "istruito" dall'utente riguardo il documento da firmare, ha precalcolato l'impronta e restituirà la firma del documento solo nel caso in cui le due impronte, quella precalcolata dal dispositivo stesso e quella fornita dal software di firma, coincidono.

Il protocollo che implementa tale soluzione è mostrato in Figura 2, dove sono riportate rispettivamente la procedura seguita dal software di firma, denotata da Signature Software Procedure, e quella seguita dalla Smart Card, indicata con Smart Card Procedure.

Signature Software Procedure

```

Require (User, document) //Riceve il documento dall'utente
I1=SHA-1 (document) //Calcola l'impronta del documento
Require (User, PIN) //Richiede il PIN all'utente
Send (SmartCard, PIN) //Invia il PIN alla Smart Card
Send (SmartCard, I1) //Invia l'impronta alla Smart Card

```

Smart Card Procedure

```

Require (SignatureSoftware, PIN) //Riceve il PIN dal software di firma
IF check (PIN)=false THEN //Se il PIN non è corretto, termina
  STOP
Require (SignatureSoftware, I1) //Riceve l'impronta dal software di firma
*Require (User, document) //Riceve il documento dall'utente
*I2=SHA-1 (document) //Calcola l'impronta del documento
*IF I1≠I2 THEN //Se le impronte differiscono, termina
* STOP
sign=RSA (I1, KS) //Calcola la firma dall'impronta e
//dalla chiave segreta dell'utente
send (SignatureSoftware, sign) //Restituisce la firma al software di firma

```

Figura 2. Il nuovo protocollo per la firma di un documento.

Alcuni passi della Smart Card Procedure sono stati evidenziati attraverso il simbolo * e indicano operazioni implementate nella soluzione da noi proposta e che non sono previste dal protocollo di firma attuale. E' evidente che le modifiche da adottare riguardano essenzialmente il dispositivo di firma, lasciando immutato il resto del protocollo.

La soluzione proposta presenta comunque alcune difficoltà realizzative. Un primo problema riguarda come trasferire un file di dimensioni arbitrarie su un

dispositivo (Smart Card o Java Card) di risorse limitate, e come calcolare in maniera efficiente l'impronta all'interno del dispositivo. Questo è il motivo per cui l'onere del calcolo dell'impronta del documento è normalmente attribuito al software di firma.

Per risolvere questo problema è stato utilizzato un algoritmo che partendo dal file da firmare è in grado di partizionarlo in blocchi di dimensione tale da essere supportata dalla Java Card e di procedere in maniera serializzata per il calcolo dell'impronta. Nel nostro caso abbiamo scelto di implementare la funzione SHA-1 [17], una delle più utilizzate nell'ambito della firma digitale, sebbene l'esecuzione di una differente funzione hash, quale ad esempio RIPEMD-160 [7], non presenti modifiche a livello architetturale.

In particolare, l'algoritmo per il calcolo della funzione SHA-1 considera i primi 512 bit del file, li divide in 16 blocchi a 32 bit, ciascuno dei quali viene elaborato per 80 volte. L'elaborazione consiste nell'applicazione di diversi operatori booleani (xor, and, or, not e rotazioni) determinati dalla posizione dei bit all'interno del blocco e da un seme iniziale. Il seme iniziale e il tipo di operazione da applicare sono stabiliti dalla funzione hash. Il risultato della prima elaborazione determina il nuovo seme per la seconda elaborazione, e così via. Il processo viene iterato fino all'ultimo blocco del file. Nel caso in cui quest'ultimo blocco abbia dimensione inferiore a 512 bit, viene "allungato" utilizzando le regole di *padding* dettate dalla funzione hash SHA-1 (nella prossima sezione verrà descritta in dettaglio l'architettura utilizzata per realizzare tale protocollo).

In tal modo la Java Card riesce, blocco dopo blocco, a calcolare l'impronta del file scelto dall'utente. La Java Card memorizza il risultato in una struttura dati e rimane in attesa di ricevere l'impronta generata dal software di firma. Se il software di firma risiede sul PC non è stato compromesso, l'impronta del documento da firmare ricevuta dalla Java Card risulterà identica a quella memorizzata e la Java Card firmerà il file. Al contrario se il software di firma si comporta in modo malizioso tentando di far firmare un documento differente da quello scelto dall'utente, il confronto delle due impronte effettuato dalla Java Card restituirà esito negativo e la Java Card non firmerà il file, rendendo non proficuo l'attacco tramite trojan.

Il problema che è necessario ora affrontare riguarda come implementare l'applicazione che si occupa di prelevare il file scelto dall'utente e inviarlo alla Java Card per calcolarne la funzione hash. Optare per la creazione di un'eseguibile memorizzato all'interno del dispositivo ed eseguita dal personal computer risulta poco vantaggiosa. Infatti l'applicazione produrrebbe problemi di portabilità e rimarrebbe ugualmente vulnerabile ad un attacco tramite trojan, per cui risulterebbe difficile garantirne la sicurezza.

Nella soluzione da noi proposta e che verrà descritta con maggiore dettaglio nella prossima sezione, tale applicazione è implementata come Java Applet memorizzata all'interno della Java Card. L'utilizzo della tecnologia JAVA fornisce numerosi vantaggi: l'applet supera i problemi di portabilità per cui per essere eseguita è sufficiente la presenza della Java Virtual Machine sul personal computer ed inoltre è possibile garantire autenticità e integrità dell'applet attraverso il

meccanismo di firma delle applet [16]. L'utilizzo di tale soluzione rende notevolmente più difficile l'attacco tramite trojan, che deve avere ora come destinatario l'applet anziché il software di firma. Il rischio di corruzione dell'applet è a un livello significativamente inferiore rispetto a quello di corruzione del software di firma perchè è possibile sfruttare le caratteristiche di sicurezza della Java Virtual Machine e i meccanismi di autenticazione delle applet. Inoltre è possibile adottare classici meccanismi di autenticazione sicura della sessione di comunicazione tra applet e smart card, utilizzando una *one-time key*, generato dalla smart card e incluso nell'applet prima dell'invio al PC. In tal modo ogni sessione di comunicazione tra applet e smart-card, viene riconosciuta dalla smart card come iniziata realmente dall'applet, grazie all'uso della chiave condivisa. Questo meccanismo evita che una compromissione del software di firma che permetta a quest'ultimo di emulare il comportamento dell'applet, possa consentire l'invio alla smart-card di documenti diversi da quello realmente voluto dall'utente.

5 Dettagli Implementativi

In questa sezione descriviamo l'architettura del sistema che implementa la soluzione proposta e che si basa sull'utilizzo delle Java Card Applet. Ogni Java Card può caricare ed eseguire diverse Java Card Applet per permettere la presenza di più funzionalità sulla stessa card. Le Applet sono "passive", nel senso che non iniziano mai una comunicazione ma possono solo rispondere a richieste provenienti da un software attraverso il lettore della Java Card. Per motivi di sicurezza le Applet sono isolate di default una dall'altra, sebbene sia possibile farle condividere risorse richiedendolo esplicitamente.

La Java Card contiene la Java Card Virtual Machine, deputata all'elaborazione del bytecode, e l'insieme dei metodi nativi che sono implementati tramite opportuno hardware nel caso debbano eseguire elaborazioni dedicate, quali ad esempio implementazioni di algoritmi di crittografia.

Per poter realizzare la soluzione proposta sono state progettate ed implementate tre Java Card Applet, che denotiamo rispettivamente con i nomi *I/O Applet*, *Hash Applet* e *RSA Applet*.

L'architettura del sistema è mostrata in Figura 3, dove sono state evidenziate le componenti principali, la prima contiene funzionalità legate alla Java Card, la seconda contiene funzionalità relative al personal computer. In particolare, all'interno della Java Card sono visibili le tre applet e la Java Card Virtual Machine (JCVM). Nel personal computer sono evidenziati il software di firma e la Java Virtual Machine.

L'I/O Applet si occupa di richiedere all'utente il file da firmare, che è memorizzato in memoria di massa sul personal computer dell'utente. Per questo motivo l'applet interagisce con la Java Virtual Machine del personal computer, per ciò che riguarda la lettura del file. Una volta individuato il file, l'applet lo preleva a blocchi di 512 bit e lo trasferisce alla Java Card. L'esecuzione di quest'applet è effettuata in sincronia con l'Hash Applet che si occupa di prelevare il blocco di 512 bit fornito dall'I/O Applet e di elaborarlo eseguendo l'algoritmo serializ-

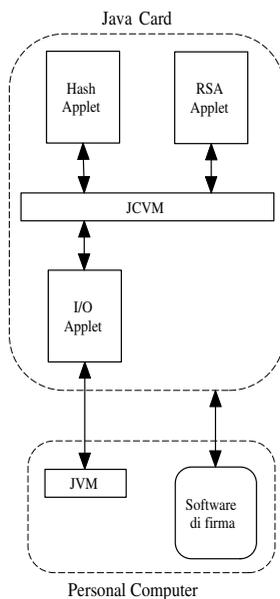


Figura 3. Architettura del sistema.

zato per il calcolo della funzione hash SHA-1 descritto nella sezione precedente. L'esecuzione alternata delle due applet termina con l'ultimo blocco del file.

Il risultato dell'elaborazione dell'Hash Applet, cioè i 160 bit dell'impronta del documento, viene memorizzato nella memoria RAM della Java Card all'interno di una opportuna struttura, che denotiamo nel seguito con B .

Per il trasferimento dei blocchi del file è stato utilizzato il protocollo APDU (Application Protocol Data Unit)[6], protocollo standard per la comunicazione tra Smart Card e lettore, che prevede la possibilità di inviare opportuni *command APDU* cui il destinatario risponde mediante *response APDU*.

La terza applet, l'RSA applet, si occupa di generare la firma a partire dall'impronta inviata dal software di firma. La realizzazione di questa applet è stata semplificata dalla presenza, all'interno dei metodi nativi della Java Card Virtual Machine, di tutte le primitive necessarie per l'implementazione dell'algoritmo RSA. Inoltre, l'utilizzo di metodi nativi ha permesso di rendere efficiente tale operazione. Questa applet, una volta ricevuta l'impronta, prima di generare la firma effettua un controllo: se il campo B è vuoto, allora avvia l'esecuzione dell'I/O Applet (che a sua volta invoca l'Hash Applet) affinché venga richiesto all'utente il documento da firmare. Una volta terminata l'elaborazione dell'applet HASH o se inizialmente B è non vuoto, l'RSA Applet confronta il contenuto dell'impronta ricevuta dal software di firma con quella calcolata e memorizzata in B e, solo nel caso esse coincidano, genera effettivamente la firma.

La validità della soluzione proposta è stata testata attraverso l'utilizzo di un emulatore software dei dispositivi di firma fisici. L'ambiente di sviluppo Ja-

va Card della SUN [13] mette a disposizione l'emulatore javacard CREF (C-language java card runtime environment) che simula in ogni dettaglio il comportamento di una Java Card. Inoltre per poter effettivamente testare la soluzione, sono state implementate ulteriori applet per la verifica del PIN immesso dall'utente e per la generazione delle chiavi (pubblica e privata) di dimensione pari a 1024 bit.

6 Conclusioni

La costante crescita e la diffusione delle applicazioni che utilizzano la firma digitale richiedono, sempre più, il rispetto di meccanismi di sicurezza, atti ad impedire a terzi l'uso improprio di questo sistema. Lo scopo di questo lavoro è di aumentare la robustezza della firma digitale, in particolare proponendo modifiche al protocollo di firma per impedire la realizzazione di attacchi basati sulla compromissione del software di firma. L'utilizzo di una Java Card ha avuto il duplice ruolo di generare la firma digitale e di controllare la validità del documento da firmare, abilitando l'apposizione della firma, solo dopo che il controllo validato dall'utente è stato effettuato con successo. La scelta di utilizzare una Java Card è fortemente sensata anche nell'ottica di sviluppi futuri, in quanto potrà essere usata per automatizzare attività legate alla firma basate sul contenuto dei documenti e sul tipo di transazione. Si pensi ad esempio ai limiti di uso del certificato, previsti dalla normativa, che restringono l'uso della firma a determinati casi. Attraverso la Java Card e attraverso i formati XML si potrebbero verificare i limiti di uso di un certificato all'atto della sottoscrizione di un documento in maniera automatica.

Ringraziamenti

Questo lavoro è parzialmente supportato dal Ministero per l'Università e la Ricerca Scientifica nell'ambito del progetto Quadrantis.

Riferimenti bibliografici

1. M. Abadi, M. Burrows, C. Kaufman, and B. Lampson. Authentication and delegation with smart-cards. In *TACS'91: Selected papers of the conference on Theoretical aspects of computer software*, pages 93–113, Amsterdam, The Netherlands, The Netherlands, 1993. Elsevier Science Publishers B. V.
2. István Zsolt Berta, Levente Buttyán, and István Vajda. Mitigating the untrusted terminal problem using conditional signatures. In *ITCC '04: Proceedings of the International Conference on Information Technology: Coding and Computing (ITCC'04) Volume 2*, page 12, Washington, DC, USA, 2004. IEEE Computer Society.
3. F. Buccafurri and G. Lax. One-time number credit cards with strong efficiency features. Technical report, Laboratorio di Informatica, 01/2007.

4. Ronald Cramer and Victor Shoup. Signature schemes based on the strong rsa assumption. In *CCS '99: Proceedings of the 6th ACM conference on Computer and communications security*, pages 46–51, New York, NY, USA, 1999. ACM Press.
5. Direttiva 1999/93/CE del Parlamento europeo e del Consiglio del 13 dicembre 1999. G.u. delle comunità europee l. 13 del 13 dicembre 1999.
6. D. Deville, A. Galland, G. Grimaud, and S. Jean. Smart card operating systems: Past, present, and future, 2003.
7. Hans Dobbertin, Antoon Bosselaers, and Bart Preneel. RIPEMD-160: A strengthened version of RIPEMD. In *Fast Software Encryption*, pages 71–82, 1996.
8. DPCM 8 febbraio 1999. Regole tecniche per la formazione, la trasmissione, la conservazione, la duplicazione, la riproduzione e la validazione, anche temporale, dei documenti informatici ai sensi dell'articolo 3, comma 1, del decreto del presidente della repubblica, 10 novembre 1997, n. 513.
9. DPCM 13 gennaio 2004. Regole tecniche per la formazione, la trasmissione, la conservazione, la duplicazione, la riproduzione e la validazione, anche temporale, dei documenti informatici.
10. Smith S. Tygar D. Gobioff, H. and B. Yee. Smart cards in hostile environments. In *Proceedings of the 2nd USENIX Workshop on Electronic Commerce*, pages 23–28, 1996.
11. R. Housley, W. Polk, W. Ford, and D. Solo. Internet x.509 public key infrastructure certificate and certificate revocation list (crl) profile, 2002.
12. ITSEC. Information technology security evaluation criteria (ITSEC): Preliminary harmonised criteria. June 1991. Document COM(90) 314, Version 1.2. Commission of the European Communities.
13. Java Card Development Kit for the Java Card.
http://java.sun.com/products/javacard/dev_kit.html.
14. Burton S. Kaliski Jr. An overview of the PKCS standards. Technical report, 1993.
15. Yingjiu Li and Xinwen Zhang. Securing credit card transactions with one-time payment scheme. 4:413–426, 2005.
16. N. V. Mehta and K. R Sollins. Expanding and extending the security features of Java. In *7th Usenix Security Symposium*, pages 159–172, 1998.
17. NIST/NSA. Fips 180-2 secure hash standard (SHS). NIST/NSA, aug 2002.
18. W. Polk R. Housely, W. Ford and D. Solo. Internet x.509 public key infrastructure, Jan. 1999.
19. R. L. Rivest, A. Shamir, and L. Adleman. A method for obtaining digital signatures and public-key cryptosystems. *Commun. ACM*, 21(2):120–126, 1978.
20. B. Schneier and A. Shostack. Breaking up is hard to do: Modelling security threats for smartcards. In *Proc. USENIX Workshop Smart Card Technology*, 1999.
21. MULTOS the High Security Smart Card OS. <http://www.multos.com>.
22. Bennett Yee and J. D. Tygar. Secure coprocessors in electronic commerce applications. In *Proceedings of the first USENIX Workshop on Electronic Commerce*, pages 155–170, 1995.

KDDBroker: Description and Discovery of KDD Services

Jessica Cellini, Claudia Diamantini, Domenico Potena

Dipartimento di Ingegneria Informatica, Gestionale e dell'Automazione,
Università Politecnica delle Marche - via Brecce Bianche, 60131 Ancona, Italy
{cellini,diamantini,potena}@diiga.univpm.it

Abstract. Web Service technologies can be profitably used in the domain of distributed Knowledge Discovery in Databases (KDD), since they provide a way to effectively share and re-use information, tools, models and domain expertise, and to compose different tools to design a KDD process. A key component of a Web Service architecture is the service broker, which provides service publishing and discovering facilities, by exploiting information stored in the UDDI registry. In this paper we present the results of an ongoing project for the design of a KDD service broker, which extends the existing standards in order to introduce semantic information on the KDD domain. In particular, the paper focuses on the semantic description and discovery of Data Mining tools. To this end, we introduce an ontology that, at this stage, represents a categorization of KDD domain in terms of tasks, methods and algorithms, and gives linkability relations among them. The ontology is then exploited for tool annotation. We discuss a possible architecture for the inclusion of the ontology in the UDDI, and present the implementation of the service broker functionalities.

1 Introduction

The goal of a Knowledge Discovery in Databases (KDD) process is to mine databases in order to identify valid, novel, potentially useful, and ultimately understandable knowledge nuggets [12]. To this end, it implies the accomplishment of different tasks, like domain understanding, data preprocessing, data mining, and knowledge interpretation/evaluation. Each of these can be realized by different and various tools. Such tools come from different research areas, like databases, machine learning, artificial intelligence, statistics and so on. Thus, a user should have various skills and expertise in order to manage all of them. In a distributed and collaborative scenario, where tools are put at disposal by different actors distributed over the Net, the design of a KDD process becomes more complex. In this scenario, there is a lack of knowledge about both the functionalities of tools and their technical details. Even if the tool is provided with a description, it is not given neither in a standard nor machine-readable form. Moreover, due to the continuous research activities in KDD and Data Mining fields, the amount of tools that a user can use for discovering knowledge is constantly growing. In [10, 11] we extensively discussed and motivated the use of

Web Service technologies in order to build an open, collaborative and distributed support environment, where resources can be easily annotated, introduced, accessed, described, composed and activated. There, the general architecture of a platform called *Knowledge Discovery in Databases Virtual Mart* (KDDVM) is also provided [16]. A core element of our platform as well as any other Web Service architecture [9, 23] is the service broker, that is a registry and a set of APIs to interact with it. According to W3C standards [28], we implement the registry by the Universal Description, Discovery and Integration (UDDI) specification. At present, a service broker is able to provide a narrow set of discovery functionalities: the search for a service is only based on its name or, at the most, on its provider. These functionalities limit the support that can be offered to a user in the KDD domain.

To give some example, let us consider a KDD consulting company, dealing with a classification problem. Consultants, on the basis of their experience and analyzing the problem, decide that a non-linear Support Vector Machine can be profitably exploited to achieve their goal. In this case, a service broker should be able to search for services implementing the specific algorithm, and besides listing services implementing “non-linear Support Vector Machine”, the broker should reply with services where the algorithm’s name is “SVM”, “SVMlight”, “RBF Support Vector Machine”, and so on. At this point, the service is activated and a model is built, but consultants decide to improve the result trying a feature extraction tool to reduce the problem dimensionality. Hence, a useful discovery functionalities could be the search for feature extraction algorithm that can be used in combination with SVM, like SVMDBA [29]. Besides these examples, others specific KDD discovery functionalities should be provided, like search services accomplishing a given task or method, search services on the basis of their input/output, as well as a search based on authors/providers reliability. Thus, in this paper we present the results of an ongoing project for the design of a KDD service broker (KDDBroker), which extends the existing standards in order to introduce semantic information on the KDD domain. To this end, we introduce an ontology that, at this stage, represents a categorization of KDD domain in terms of tasks, methods and algorithms, and gives linkability relations among them.

In the next section, we present the data schema and APIs of the UDDI standard. Section 3 gives an overview of related works, discussing the solutions proposed for extending the UDDI registry and for supplying a broker with semantic functionalities. In Section 4, we show the information needed to describe a generic KDD tool and to introduce a language to represent it in the KDDVM. Section 5 proposes KDDONTO, an ontology for KDD domain. In Section 6, we first present the service broker for the KDDVM platform, then an implementation is proposed. Finally some conclusions are given.

2 UDDI Broker

The Universal Description, Discovery and Integration (UDDI) [25] specification is an initiative to create a global, platform-independent, open framework for publishing and discovering Web Services. The UDDI standard allows businesses to describe their business services, how services can be accessed, what information they want to make public, and what information they choose to keep private. Furthermore, UDDI supplies customers with a set of standard functionalities for service discovery. In order to support the discovery process, the UDDI provides three types of information about Web Services:

- white pages: name and contact details;
- yellow pages: categorization based on business and service types;
- green pages: technical data about services;

The UDDI specifications take advantage of World Wide Web Consortium (W3C) [28] and Internet Engineering Task Force (IETF) [14] standards such as Extensible Markup Language (XML), HTTP, Domain Name System (DNS) protocols, Web Service Description Language (WSDL) and Simple Object Access Protocol (SOAP) messaging specifications. UDDI data model is defined by four basic entities: *businessEntity*, *businessService*, *tModel*, and *bindingTemplate*. Each instance of the basic entities is uniquely identified by the Universally Unique Identifier (UUID). These entities are briefly described in the following.

businessEntity:

The *businessEntity* is the principal entity of UDDI. It contains the core details of the business: its name and description, details of personal contacts within the business, identifiers used to refer to the business, categorical classifications of the business. Name and contact details define the white pages information of the UDDI registry. Categorical classifications of the business is part of the yellow pages. A *businessEntity* contains a list of services provided by the business, whose descriptions are contained in the *businessService* entity.

businessService:

Each *businessService* instance describes a single service provided by a business. A service is characterized by a name, a description, and a classification of the service that completes the yellow pages registry. It contains also a link to one or more binding templates providing technical information.

bindingTemplate:

BindingTemplate mainly represents the technical information about the access point (URL) of the service. *BindingTemplate* information is typically retrieved by an organization interested in using the related *businessService*, and exploited for dynamic invocation at runtime. *BindingTemplate* allows a publisher to suddenly change the entry point without notifying all its business partners of this change. Each *bindingTemplate* references to one or more instances of *tModel*, describing the technical information about how to access the service.

tModel:

A *tModel*, or technical model, represents two concepts within UDDI: a technical

specification for a given service type, or a model for a particular identifier or taxonomy. For examples tModels can refer to WSDL files, XML DTDs, or classification schemes. A unique tModel can be addressed by different services. A tModel gives information about its name, the name of the organization that publishes it, a list of categories that identify the service type, and pointers to technical specifications (e.g. interface definitions, message formats, and message protocols). BindingTemplate and tModel together define the green pages.

Classification of services, provided by the yellow pages, is maintained in a specific structure called *categoryBag*. The UDDI public registries provide validated support for a number of taxonomies that cover industry codes (NAICS), product and service classifications (UNSPSC), and geographical classification (ISO 3166). New domain classifications can be specified and included in the categoryBag. Inclusion of classifications is optional but greatly enhances search functionalities.

The UDDI specifications also include definitions for web services interfaces in order to allow programmatic access to the registry information (APIs). These are divided into two logical parts: discovery/inquiry and publishing APIs.

Although in the last UDDI version (3.0.2) new discovery features are introduced, at present a UDDI broker can supply user with only syntactic discovery functionalities, limiting the support that can be offered to a user in the KDD domain. As a matter of fact, users may search services only by their name (white pages) or, at the most, by a plain taxonomy (yellow pages). Thus, it is needed extending UDDI standard in order to introduce new discovery functionalities based on services capabilities.

3 Related works

In the literature, it is often discussed the need of both representing and searching for other kinds of information not provided in the standard. UDDI specification defines two ways to add new information into registry [25]. The former is to define a tModel in order to address a new service description, that is typically an extension of the WSDL one. The other way is to use a categoryBag to classify services based on their functionalities.

In [24] researchers discuss the introduction of new properties like service leasing and Quality of Service (QoS). To this end they develop another type of information, called blue pages. Properties are added into blue page registry by defining <name,value> pairs, like categoryBag. This approach is similar to one of the approaches discussed in [19], which proposes a domain oriented UDDI registry architecture and addresses new concepts such as service property schema, service relationship and service constraint. The authors introduce three possible solutions to UDDI extension, discussing advantages and disadvantages for each of them. The first solution provides a method to publish new information without modifying UDDI data model. Such solution uses tModel to refer to service properties. The main problem related to this solution is that any provider

should use the same property definition schema and should extend the discovery interface (i.e. the APIs). Moreover, it guarantees a low discovery efficiency, because the external files are distributed. Second solution considers extending UDDI data model in order to store non-functional information like QoS. Such UDDI extension allows for browsing and discovering only QoS properties. The last solution is to use an external database. This solution does not modify UDDI data model. In [19] authors also prove that this approach allows to store additional information related to service relationships and constraints with a high discovery efficiency. Our scenario is quite different and more complex compared with previous works. As a matter of fact, information we have to manage is structured and related each other. So, these approaches are inadequate to our aims.

An important research direction is to discover web services on the basis of the capabilities they provide. Note that this implies the use of some form of semantic information, that can be represented by means of ontologies. In [20], authors propose a semantic broker for InfoSleuth, an agent-based platform for information discovery and retrieval in a dynamic, open environment. This broker searches for agents that provide services to agents requesting services. To this end, the broker implements reasoning capabilities that exploit a common and shared ontology concerning both agents description and functionalities they provide. A more general approach to discovery information is that described in [8], where the authors analyze a model for the semantic interoperability of an open network system. This paper focuses on the information resource discovery problem, where autonomous businesses require a coordinated access to heterogeneous and distributed information resources. The search function is based on a semantic affinity evaluations on ontology, representing the semantics of the information resources required and in use by a given business. In [4] and [21], the problem of compiling semantic information into UDDI is discussed. In [4], in order to enhance the UDDI broker functionalities, an approach to ontology-based service discovery is proposed. In this work a service is selected through semantic-based matching algorithms according to a scored mechanism taking into account semantic relationships among services. [21] shows how services capabilities described in DAML-S can be mapped into UDDI registry, and how they can be used to perform a semantic service discovery. These works propose methodologies for semantic characterization of general *business* services, while our scope is to design an UDDI broker for the management of specific KDD information.

4 KDD service

The Knowledge Discovery in Databases process includes several steps, that implies the interaction of various tools derived from different domains. The heterogeneity of tools as well as the goal-driven and domain dependent nature of any KDD problem introduce complexity in the design of a KDD process. Such a complexity is mainly due both to the huge amount of tools the user can choose

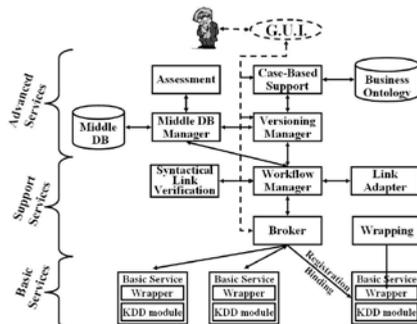


Fig. 1. Knowledge Discovery in Databases Virtual Mart: main services and their interactions.

and to the expertise needed for facing various KDD tasks. So, it is also necessary to define an environment supporting users in specific KDD activities, like:

- building a repository of KDD tools;
- easily introducing in the repository new algorithms or releases;
- browsing the tool repository and obtaining information about tools;
- managing and understanding the input dataset;
- choosing the more suitable tools on the basis of a number of characteristics like: tools performances (complexity, scalability, accuracy), the kind of data tools can be used for (textual/symbolic, numeric, structured data, sequences, ...), the kind of goal tools are written for (data cleaning, data transformation, data mining, visualization, ...) the kind of data mining task (classification, rule induction, ...);
- setting the tool parameters and activating it;
- designing the KDD process by tools composition;
- managing the KDD whole process and its versions;
- representing the models in a understandable form.

Open support environments proposed in the recent literature are based on architectures like Grid [6, 7, 27, 5] and Service Oriented Architecture (SOA) [18, 17, 13, 1]. Such proposals are mainly focused on supporting the data mining phase. In order to support users in the design of a whole KDD project, we are working on the development of an open environment based on the service-oriented paradigm, called *Knowledge Discovery in Databases Virtual Mart* [10] (see Fig. 1).

Such system is formed by three kinds of services: basic, support and advanced services. The former includes tools implementing data mining algorithms as well as any other data manipulation algorithm. Advanced services provide an "intelligent" support to user. In particular, they are mainly devoted to proactively help user in the choice of tools most suitable with her/his goals. Finally, the core layer of KDDVM contains support services, that mainly give support to browsing, composition and activation of tools. We codified the information needed to accomplish these tasks in the *Knowledge Discovery Tool Markup Language*

(KDTML)[22]. KDTML tool descriptions are XML documents where XML tags describe the characteristics of tools and their interfaces. Due to lack of space, we refer to the KDDVM project site [16] for both the general structure of a KDTML document and some examples. The language is divided into four main sections: the initial KDTML fragment contains information for tool *location and execution*, the second section describes the tool *I/O interface*, the third part lists the KDD software modules that can be linked up with the described tool (*linkable modules*), and the final section is a categorization of the KDD tool with respect to a KDD *taxonomy*. In particular, the last two parts play a leading role in the KDDVM. As a matter of fact, in order to give support to users with different degree of expertise and knowledge, it is necessary to give a description of functionalities of any KDD tools. This information is represented according to both a common vocabulary and a predefined taxonomy. Such a structure is an extension of the Data Mining taxonomy DAMON [6], which covers the other KDD tasks, like data cleaning, data transformation, data selection, visualization and so on. A taxonomical structure is a natural way to characterize the KDD tools domain in terms of the implemented task. The goal of each KDD task can be achieved by various and different methods, e.g. for the classification task a user can use a Decision Tree, a Neural Network, a Fuzzy Set Approach as well as a Genetic Algorithm. For each method are available a lot of algorithms, e.g. C4.5, ID3 and CART are Decision Tree algorithms. Finally any KDD tool is a specific implementation of a given algorithm. Furthermore, in order to achieve a KDD goal, the exploitation of a single tool is often not sufficient, rather a KDD process is typically composed of many different tools, such that the output of a tool represents the input to another one. Then, in order to support the user in such a tool composition activity, the description of a KDD tool is enriched with information about tools that can be linked up with the described one.

5 KDDONTO: a KDD ontology

This section is devoted to present the design of KDDONTO an ontology for the KDD domain that, at this stage, describes KDD algorithms and gives linkability relations among them. In the following, we first present an abstract specification of the ontology, then an OWL implementation is proposed.

Analyzing the information needs to describe a KDD tool, it turns out that the main concepts to represent are: *algorithm*, *method* and *task*. More formally, for the scope of our ontology:

- *algorithm* is a procedure through which a KDD tool is accomplished;
- *method* is a KDD methodology or technique used to achieve a specific task;
- *task* is a goal of a KDD problem.

These concepts are represented by disjoint classes.

Categorical relationships among these concepts exist. In particular, each algorithm implements a specific method, while a method can be performed by different algorithms. As regards *task* and *method*, a task can be achieved by

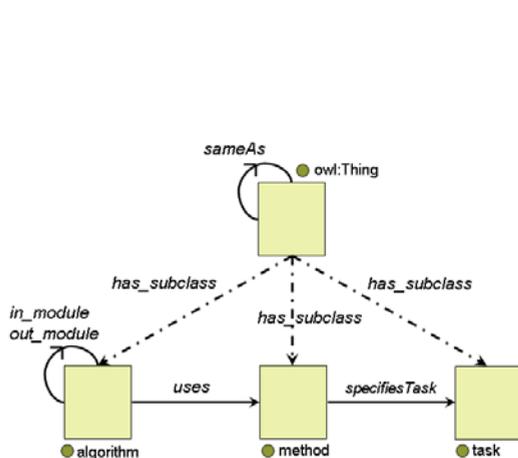


Fig. 2. An overview of the KDDONTO.

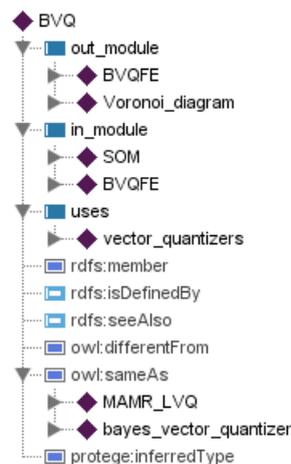


Fig. 3. Properties of BVQ algorithm.

multiple methods, and vice versa. Moreover, each algorithm can be linked to other algorithms, according to the *linkable modules* section of KDTML. Finally, in the KDD literature the same instance of a concept could be indicated with different names or acronyms (e.g. BVQ and MAMR LVQ represent the same algorithm, SVM is the acronym of Support Vector Machine). Thus, an alias relationship between different instances is also introduced. So, in KDDONTO the following properties are defined:

- *uses* between an instance of *algorithm* and the implemented instance of *method*;
- *specifiesTask* between an instance of *method* and its instance of *task*;
- *in_module* between instances of *algorithm*, the range is formed by the algorithms that are often used before the described one;
- *out_module* between instances of *algorithm*, it is the inverse of *in_module*;
- *sameAs* between instances belonging to any concept of the ontology, its range is the set of instances having the same meaning.

Figure 2 shows an overview of concepts and properties of the KDDONTO. Figure 3 shows the piece of KDDONTO related to Bayes Vector Quantizer (BVQ) algorithm, a Data Mining algorithm used in classification tasks. In particular, Figure 3 lists the BVQ properties: “BVQ” is the same of “MAMR_LVQ” and “Bayes Vector Quantizer”, the output of BVQ can be used in input to a feature extraction algorithm based on LVQ model (BVQFE) as well as to an algorithm extracting the symbolic form of the Voronoi diagram, and so on.

6 KDDBroker: a broker for KDD

This section is devoted to present the KDDBroker, a broker for the publication and discovery of KDD services. To this end, we first illustrate the approach we

use to extend UDDI registry, and then we propose an implementation of the broker.

First of all, let us remember that the main goal of a service broker is to help users in obtaining information about the tools. This information can be found in the UDDI registry, while more detailed technical information is in WSDL. Hence, the first step is to analyze the KDTML main elements, in order to evaluate whether they can be described by Web Service standards, in particular UDDI and WSDL. Information contained in the *Location and execution* section of the KDTML is standard information, so WSDL is sufficient to describe it. On the other hand, although WSDL has some tags to describe I/O, the KDTML *I/O interface* section introduces additional tags that have not a correspondence in WSDL. In order to represent all the information about the I/O of KDD tools we defined an extended WSDL (eWSDL). At this point, as usual, a tModel is used to address the eWSDL of the service. Several examples of eWSDL describing different KDD services are available at [16]. As regards *taxonomy* and *linkable modules* sections, the extension of the UDDI registry regards the use of categoryBag structure, included in the businessService entity. This structure describes a new property called “algorithm”, whose value represents the name of the algorithm implemented by the service. Furthermore, the categoryBag points to a tModel addressing the ontology where the specific algorithm is classified. In Figure 4 we show the overview of our solution. The adopted solution presents

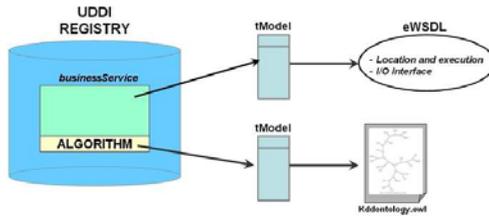


Fig. 4. Overview of UDDI architecture.

the following advantages:

- the standard UDDI structure is not modified, hence more traditional services and APIs can coexist in our broker;
- the choice of representing taxonomy information by an ontology allows us to implement reasoning capabilities;
- adding more categoryBag instances, we can describe the service by different concepts in different ontologies. The use of multiple ontologies is specially relevant in a collaborative environment, since it allows to model different pieces of domain knowledge by independent experts. However, the use of multiple ontologies leads also to consider integration and reconciliation issues;
- an external ontology can be profitably used by some advanced services of KDDVM for giving a proactive support to the user, for instance in the choice

of the most suitable tools w.r.t. her/his KDD problem as well as in tools composition.

At present the KDDVM project uses Apache as web server, Jakarta Tomcat 5.5.17 [2] as the application server, and Axis 1.4 [3] as the SOAP engine. Tomcat makes available KDD services to users, while Axis allows to receive, to translate and to send SOAP messages. In order to build the service broker we choose JUDDI 0.9 RC4 [15] and UDDI4J 2.0.5 [26], that in the literature are recommended as the best solution for implementing an UDDI-based broker over an Apache-Tomcat server. JUDDI defines the data schema of the registry, while UDDI4J is a Java class library that provides a set of APIs to interact with the registry.

We developed new APIs in order to allow the discovery and publication of specific information about KDD services. In particular, for the semantic discovery we introduce two main methods:

- *ontoInquiry*, that returns the instances of the ontology satisfying a given RDQL¹ query;
- *semInquiry*, that returns the KDD services implementing a given algorithm described according to a given ontology.

Combining these APIs, we obtain various functionalities, like: search of KDD services on the basis of methods and/or tasks, search of KDD services that implement a given algorithm, search of KDD services that can be linked up to other ones, and publication of KDD services and their properties.

Figure 5 shows a screenshot of the Service Broker Client, that provides a user-friendly interface to KDDBroker. In the example we search for KDD services implementing a Classification task, and in particular a Vector Quantizer method. As result, the broker provides the name of KDD services satisfying the above requirements, together with additional information: the WSDL address, the implemented algorithm, a brief description of service behavior, the Business providing the service, linkable algorithms and information about the QoS of the provider which, at present, is simply stated as estimated availability.

7 Conclusion

In this paper we present results of an ongoing project for the design of the KDDBroker, an UDDI-based service broker for publication and discovery of KDD services. To this end we also introduce KDDONTO, an ontology for representing the KDD domain. Including ontology in the UDDI allows us to enrich the broker with various semantic discovery functionalities. We also discussed some issues related to the current implementation of the service broker and showed a user-friendly interface designed to easily inquire the broker. This work is part of the Knowledge Discovery in Databases Virtual Mart project, a more general project for the development of an open and extensible environment where

¹ RDF Data Query Language: <http://www.w3.org/Submission/RDQL/>

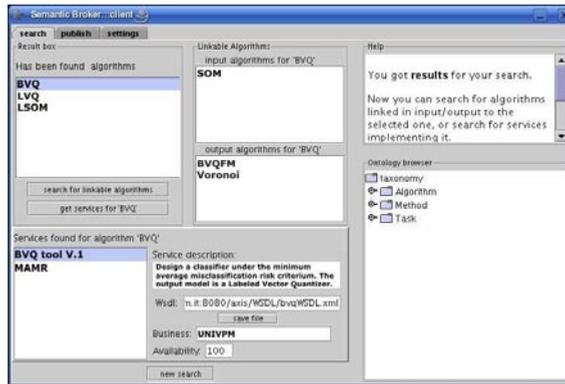


Fig. 5. An example of outcome from the Service Broker.

users can look for implementations, suggestions, evaluations, examples of use of tools implemented as services. Future work will be devoted to expand the KDD ontology with relevant concepts about the domain, and to endow the service broker with more advanced searching capabilities. Another interesting direction to investigate is to represent the whole registry by OWL-S, an OWL-based Web Service Ontology.

8 Acknowledgment

The authors would like to thank Emanuele Storti for his contribution to the implementation of the Service Broker.

References

1. Ali, A.S., Rana, O.F. and Taylor, I.J. Web services composition for distributed data mining. In *Proc. of Int. Conf. Workshop on Parallel Processing*, pages 11–18. IEEE, June 14–17 2005.
2. APACHE TOMCAT site. <http://tomcat.apache.org>.
3. AXIS site. <http://ws.apache.org/axis/>.
4. Bianchini, D. and De Antonellis, V. Ontology-based Semantic Interoperability Tools for Service Dynamic Discovery. In *Interoperability of Enterprise Software and Applications*, pages 323–333. Springer, 2006.
5. Brezany, P., Hofer, J., Wohrer, A. and Tjoa, A. Towards an open service architecture for data mining on the grid. In *Proc. of 14th Int. Workshop on Database and Expert Systems Applications*, pages 524–528, Prague, Czech Republic, Sept. 1–5 2003. IEEE.
6. Cannataro, M. Knowledge Discovery and Ontology-based services on the Grid. In *The First GGF Semantic Grid Workshop*, Chicago IL, USA, Oct. 2003.
7. Cannataro, M. and Talia, D. The Knowledge Grid. *Comm. of the ACM*, 46(1):89–93, Jan. 2003.
8. Castano, S., Ferrara, A. and Montanelli, S. Ontology-based Interoperability Services for Semantic Collaboration in Open Networked Systems. In *Interoperability of Enterprise Software and Applications*, pages 135–146. Springer, 2006.

9. Curbera, F., Duftler, M., Khalaf, R. and Nagy, W. Unraveling the Web Services Web: an Introduction to SOAP, WSDL and UDDI. *IEEE Internet Computing*, pages 86–93, March 2002.
10. Diamantini, C., Potena, D. and Panti, M. Developing an Open Knowledge Discovery Support System for a Network Environment. In *Proc. of the 2005 Int. Symposium on Collaborative Technologies and Systems*, pages 274–281, Saint Louis, Missouri, USA, May 15-20 2005.
11. Diamantini, C., Potena, D. and Smari, W. Collaborative Knowledge Discovery in Databases: a Knowledge Exchange Perspective. In *Proc. of AAAI Fall Symposium on Semantic Web for Collaborative Knowledge Acquisition*, pages 24–31, Arlington, VA, USA, Oct. 13-15 2006.
12. Fayyad, U. M., Piatetsky-Shapiro, G., Smyth, P. and Uthurusamy, R. *Advances in Knowledge Discovery and Data Mining*. AAAI/MIT Press, 1996.
13. Grossman, R. and Mazzucco, M. DataSpace: a Data Web for the Exploratory Analysis and Mining of Data. *IEEE Computing in Science and Engineering*, 4(4):44–51, July-Aug. 2002.
14. IETF site. <http://www.ietf.org>.
15. JUDDI site. <http://ws.apache.org/juddi/>.
16. KDDVM site. <http://boole.diiga.univpm.it:8080/axis>.
17. Krishnaswamy, S., Zaslavsky, A., and Loke, S, W. Internet Delivery of Distributed Data Mining Services: Architectures, Issues and Prospects. In V.K. Murthy and N. Shi, editors, *Architectural Issues of Web-enabled Electronic Business*, chapter 7, pages 113–127. Idea Group Publishing, 2003.
18. Kumar, A., Kantardzic, M., Ramaswamy, P. and Sadeghian, P. An Extensible Service Oriented Distributed Data Mining Framework. In *Proc. IEEE/ACM Int. Conf. on Machine Learning and Applications*, Louisville, KY, USA, 16-18 Dec. 2004.
19. Liu, J., Gu, N., Zong, Y., Ding, Z., Zhang, S. and Zhang, Q. Service Registration and Discovery in a Domain-Oriented UDDI Registry. In *Proc. of the 2005 the 5th Int. Conf. on Computer and Information Technology*. IEEE, 2005.
20. M. Nodine, W. Bohrer, and A.H.H. Ngu. Semantic brokering over dynamic heterogeneous data sources in infosleuth(tm). *icde*, page 358, 1999.
21. Paolucci, M., Kawamura, T., Payne, T. and Sycara K. Importing the semantic web in uddi. In *Proc. of E-Services and the Semantic Web Workshop*, 2002.
22. Potena, D. and Diamantini, C. Description of Knowledge Discovery Tools in KDTML. In *Proc. of Computer and Their Applications track of the 1st Int. conf. CCSP 2005*, pages 144–248, Kuala Lumpur, Malaysia, Nov 14-16 2005. IEEE.
23. Roy, Jaideep and Ramanujan, Anupama. Understanding Web Service. *IEEE IT Pro.*, pages 69–73, November 2001.
24. ShaikhAli, A., Rana, O.F., Al-Ali, R. and Walker, D.W. UDDIe: an extended registry for Web services. In *Applications and the Internet Workshops*, pages 85–89, 27-31 Jan. 2003.
25. UDDI site. <http://www.uddi.org>.
26. UDDI4J site. <http://uddi4j.sourceforge.net>.
27. V. Curcin, M. Ghanem, Y. Guo, M. Kohler, A. Rowe, J. Syed, P. Wendel. Discovery net: Towards a grid of knowledge discovery. In *Proc. of The Eighth ACM SIGKDD Int. Conf. on Knowledge Discovery and Data Mining*, pages 658–663, Jul 23-26 2002.
28. W3C site. <http://www.w3c.org>.
29. J. Zhang and Y. Liu. Svm decision boundary based discriminative subspace induction. *Pattern Recognition*, 1(38):1746–1758, 2005.

A Hierarchical Probabilistic Model for Co-Clustering High-Dimensional Data

Gianni Costa, Francesco Folino, Giuseppe Manco, and Riccardo Ortale

ICAR-CNR
Via Bucci 41c
87036 Rende (CS) - Italy

Abstract. We propose a hierarchical, model-based co-clustering framework for handling high-dimensional datasets. The technique views the dataset as a joint probability distribution over row and column variables. Our approach starts by initially clustering rows in a dataset, where each cluster is characterized by a different probability distribution. Subsequently, the conditional distribution of attributes over tuples is exploited to discover co-clusters underlying the data. An intensive empirical evaluation confirms the effectiveness of our approach, even when compared to well-known co-clustering schemes available from the current literature.

1 Introduction

Increasing attention has been recently paid to clustering high dimensional data, since this task is of great practical importance in several emerging application settings such as text analysis, bioinformatics, e-commerce, astronomy, statistics and psychology and insurance industry [1, 2, 11, 13]. However, clustering high-dimensional data poses some challenging issues.

Foremost, data sparseness and/or skewness as well as attribute irrelevancy and/or redundancy typically impose to look for valuable clusters within several subsets of the original attribute space. This inevitably penalizes the effectiveness of clustering and further exacerbates its time requirements, since high dimensional data tends to exhibit different clusters on distinct attribute subsets. Although standard dimension reduction techniques [6] can be used to detect the relevant dimensions, these can be different for distinct clusters, which invalidates such a preprocessing task.

Also, the identification of cohesive clusters is a major concern. In most cases, cohesion is measured in terms of the syntactic similarity of the objects in a cluster. However, several irrelevant attributes might distort the actual degree of proximity between object tuples. Moreover, clustering schemes yield *global patterns*, that do not apparently capture our general understanding of complex phenomena. Indeed, in a high-dimensional setting, specific groups of objects tend to be co-related only under certain subsets of attributes. Hence, though semantically-related, two tuples with (possibly several) differences in their attribute values would hardly be recognized as actually similar by any global

model. In principle, object cohesion is better viewed in terms of *local patterns*. Precisely, the individual data tuple can be intended as a mixture of latent concepts, each of which being a suitable collection of characterizing attributes. Accordingly, two tuples are considered as actually similar if both represent at least a same concept. Viewed in this perspective, the identification of local patterns, i.e. of proper combinations of object tuples and attributes, leads to the discovery of natural clusters in the data, without incurring into the foresaid difficulties.

Co-clustering has recently gained attention as a powerful tool, that allows to circumvent the aforementioned limitations while processing high-dimensional data. Due to its intrinsic capability at exploiting the latent relationships between tuples and their own attributes, it enables the discovery of coherent clusters of similar tuples and their interplay with corresponding attribute clusters. This has been the main motivation behind the development of a wealth of new, *ad hoc* techniques that simultaneously cluster both object tuples and their attributes.

Co-clustering techniques can be divided into the five main categories [14], discussed next. The simplest class of approaches [8, 16] is the one that applies existing clustering methods to find independent row and column partitions and then combines the results into meaningful co-clusters. Divide-and-conquer strategies, such as [9], divide the original co-clustering problem into multiple subproblems of smaller size, solve them recursively and then combine the resulting solutions into an actual solution for the initial problem. Greedy iterative algorithms [4, 3] search for co-clusters in the data matrix by progressively removing or adding rows or columns, in an attempt at maximizing some local-quality criterion. Techniques based on exhaustive co-cluster enumeration [15, 17] search for all possible co-clusters in the data matrix. Model-based techniques [7, 10, 18] assume a suitable model for the data generation process and learn estimates of model parameter values from the available data. To the best of our knowledge, the approach in [18] is the most resemblant to our proposal. However, in this regard, we emphasize that the application of the EM algorithm for learning suitable estimation of parameters is not direct, due to structural dependencies in the underlying model [18], that requires suitable approximations. To the purpose, [18] pursues the maximization of a variational approximation of data likelihood, via Generalized EM [19]. On the contrary, we assume a hierarchical model for the representation of the data generating process, that allows a more direct and natural exploitation of the EM algorithm.

In this paper, we build on probabilistic techniques to develop an innovative, model-based algorithm for the discovery of actual co-clusters in high-dimensional data. The underlying intuition is that an object tuple can be thought of as the outcome of the following hierarchical, generative process: firstly pick a distribution over latent clusters; next, choose the associated concepts; eventually, generate the individual attribute values. An EM-based clustering strategy fits the probabilistic model of the foresaid generative model to the underlying data. Precisely, the joint probability distribution over row (i.e. tuple) and column (i.e. attribute) variables is exploited to initially find tuple clusters. Then, the conditional distribution of attributes over tuples is exploited to discover actual co-

clusters, i.e. for associating concept clusters with tuple clusters. A preliminary evaluation of our approach and a comparison with consolidated co-clustering schemes in the literature seem to confirm the validity of our intuition.

The plan of the paper is as follows. Section 2 formally describes the intuition behind our model-based co-clustering approach. Section 3 discusses the details of the method employed for learning suitable estimates of model parameters from the underlying data. An intensive experimental evaluation is described in section 4, that witnesses the effectiveness of our proposal. Finally, section 4 draws some conclusions and highlights directions, that are worth further research.

2 Problem Statement and Overview of the Approach

We begin by fixing a proper notation to be used throughout the paper. Data can be represented in binary format as a boolean incidence matrix D with rows $\{y_1, \dots, y_m\}$ and columns $\{x_1, \dots, x_n\}$, where each entry d_{ij} takes values into the set $\{0, 1\}$. The implicit meaning is that tuple x_j comprises attribute y_i if and only if d_{ij} takes value 1. Let X and Y be discrete random variables ranging over the sets $\{x_1, \dots, x_n\}$ and $\{y_1, \dots, y_m\}$. We are interested in simultaneously clustering X into K disjoint clusters, and Y into H disjoint clusters. That is, we aim at finding suitable column and row mappings, respectively defined as $C_X : \{x_1, \dots, x_n\} \mapsto \{\hat{x}_1, \dots, \hat{x}_K\}$ and $C_Y : \{y_1, \dots, y_m\} \mapsto \{\hat{y}_1, \dots, \hat{y}_H\}$

In our framework, we characterize the co-occurrence matrix in probabilistic terms, by estimating the joint distribution $p(X, Y)$ between X and Y . By Bayes' rule, the distribution can be modeled as $p(x, y) = p(y|x)p(x)$. Model-based clustering methods attempt to optimize the fit between the given data and some mathematical model. Such methods are often based on the assumption that the data to be clustered are generated by one of several distributions, and the goal is to identify the parameters of each. The foundation for probabilistic clustering is a statistical model called *finite mixtures*. A *mixture* is a set of probability distributions, representing clusters that govern the format for members of that cluster.

Each cluster has a different distribution. Any particular instance actually belongs to one and only one of the clusters, whose identity is however unknown. Moreover, the clusters are not equally likely: there is some probability distribution that reflects their relative populations. Within a mixture modeling framework, the above components can be described as reported below

$$p(x) = \sum_{k=1}^K p_k(x)\alpha_k \quad p(y|x) = \sum_{h=1}^H p_h(y) \cdot \sum_{k=1}^K p(\hat{x}_k|x)\beta_{h,k}$$

where $p_k(x) = p(x|\hat{x}_k)$ is the probability of x within cluster \hat{x}_k , $p_h(y) = p(y|\hat{y}_h)$ is the probability of y within cluster \hat{y}_h , $\alpha_k = p(\hat{x}_k)$ is the probability of cluster \hat{x}_k , and $\beta_{h,k} = p(\hat{y}_h|\hat{x}_k)$ is the probability of cluster \hat{y}_h given cluster \hat{x}_k . As a consequence, the mixture $p(x, y)$ can be finally modeled as

$$\begin{aligned}
p(x, y) &= \sum_{h=1}^H p_h(y) \cdot \sum_{k=1}^K p(\hat{x}_k|x)\beta_{h,k} \cdot \sum_{k=1}^K p_k(x)\alpha_k \\
&= \sum_{h=1}^H \sum_{k=1}^K \sum_{u=1}^K p_h(y)\beta_{h,k}p_u(x)\alpha_u p(\hat{x}_k|x)
\end{aligned} \tag{1}$$

The idea in the above formula is learning latent concepts from the data as well as a collection of characterizing attribute values for each such a concept. In particular, each tuple can be seen as a mixture of various concepts, where some concepts are more or less probable according to the cluster where the tuple fits. Hence, a data tuple can be thought as the outcome of the following generative model: firstly pick a distribution over latent clusters; next, choose the concepts associated and finally generate the individual attribute values.

The clustering problem can be hence reformulated as the problem of estimating the parameters of the distributions involved in the above formula. The classical *Maximum Likelihood (ML)* Estimation technique is a way for estimating the parameters of a distribution based upon observed data drawn according to that distribution. Let Θ denote a set of parameters and let x, y be a data observed from the random variables X, Y with probability distribution $p_{X,Y}(x, y|\Theta)$, parameterized by the set of parameters Θ . The key idea in *ML* estimation is to determine the parameter θ for which the probability of observing the outcome x is maximized. Function $\mathcal{L}(\theta|X, Y) = p(X, Y|\theta)$ is the *Likelihood function* and the *Maximum Likelihood* Estimation of the parameter θ is the value which maximizes the likelihood function $\theta_{ML} = \arg \max_{\theta} \mathcal{L}(\theta|X, Y)$.

In our framework, Θ represents the set of parameters governing p_k, p_h, α_k and $\beta_{h,k}$ for each h and k . We adopt a naive assumption here, that is $\beta_{h,k} \in \{0, 1\}$ and $\sum_h \beta_{h,k} = 1$. As a consequence, $p_h(y)$ can be modeled as the probability of term y within tuple cluster k (that is, $p_h(y) = p_k(y)$). This roughly consists in associating a single concept cluster with each tuple cluster, and in modeling the probability of an attribute within a tuple cluster. Thus, the set of parameters Θ involved in the estimation are now represented by sole parameters related to (tuple) cluster \hat{x}_k (i.e., the parameters governing $p_k(x), p_k(y)$ and α_k). Moreover, the probability distribution can be rewritten as

$$p(x, y) = \sum_{k=1}^K \sum_{u=1}^K p_k(y)p_u(x)\alpha_u p(\hat{x}_k|x) \tag{2}$$

In particular, by modeling $p_k(x)$ by means of a multinomial distribution, the estimation of the above parameters can be accomplished by means of the traditional Expectation Maximization algorithm, which is described below. Thus, we can suppose that $x_i = \{n_i^1, n_i^2, \dots, n_i^m\}$, where $n_i^c \in \{0, 1\}$ for each c . A multinomial distribution models a Bernoulli's distribution in several outcomes. It is characterized by a parameter σ_c , that represents the probability that an event of class c happens. The multinomial distribution for the generic cluster \hat{x}_k

is parameterized by $\sigma_c^k = p_k(y_c)$:

$$p_k(x_i) = \prod_{c=1}^m (\sigma_c^k)^{n_i^c}$$

Thus, the model parameter Θ collects every σ_c^k and α_k , for $c = 1, \dots, m$ and $k = 1, \dots, K$.

3 Multinomial Expectation Maximization

The *Expectation Maximization (EM)* [12] algorithm is a classical technique for model-based clustering. Given the dataset and a pre-specified number of clusters, the algorithm learns, for each instance, the membership probability of each cluster and, for each cluster, its descriptive model, i.e., the parameters that govern its generative process.

The algorithm requires some initial estimates for the parameters of the mixture model; given such parameters, a single EM iteration provides new parameter estimates which are proven not to decrease the likelihood of the model. The process is repeated until convergence, i.e. the likelihood of the mixture model at the previous iteration is sufficiently close to the likelihood of the current model. More precisely, the algorithm proceeds as follows:

1. *Initialization*: $g := 0$; Set initial values $\Theta^{(0)}$ for the parameter set Θ ; compute $\mathcal{Q}^{(g)} = \log(\mathcal{L}(\Theta^{(g)}|X, Y))$.
2. *E step*: Use $\Theta^{(g)}$ to compute the membership probability $p^{(g)}(\hat{x}_k|x)$ of each object x to each cluster \hat{x}_k .
3. *M step*: Update the model parameters $\Theta^{(g+1)}$, using values computed in the E Step; compute $\mathcal{Q}^{(g+1)} = \log(\mathcal{L}(\Theta^{(g+1)}|X, Y))$.
4. If $\mathcal{Q}^{(g+1)} - \mathcal{Q}^{(g)} \leq \epsilon$, stop. Else set $g := g + 1$ and restart from step 2.

We omit here the formal derivation of the steps which are at the basis of the EM approach (details can be found in the appendix of an extended version of this paper [5]). It is worth noticing, however, that the objective of the derivation is a heuristic method for maximizing the *Log-Likelihood function*

$$\log(\mathcal{L}(\Theta|X, Y)) = \sum_{i,j} \log(p(x_i, y_j|\Theta)) \approx \sum_{i,j} z_{ik} \log \left(\sum_{k=1}^K \sum_{u=1}^K p_k(y) p_u(x_i) \alpha_u p(\hat{x}_k|x_i) \right)$$

where $z_{ik} \in \{0, 1\}$ is a random variable representing the true cluster generating the data. The latter term in the equation is transformed, for the matter of convenience, into

$$\mathcal{Q}(\Theta, \Theta^{(g-1)}) = E[\log(\mathcal{L}(\Theta|X, Y, Z))|X, Y, \Theta^{(g-1)}]$$

The E and M steps, at the generic iteration g , can be shown to be as follows:

E Step. Working on $\mathcal{Q}(\Theta, \Theta^{(g-1)})$ yields

$$p^{(g)}(\hat{x}_k | x_i) = \gamma_{ik}^{(g)} = \frac{\alpha_k^{(g-1)} p_k(x_i | \Theta^{(g-1)})}{\sum_{j=1}^K \alpha_j^{(g-1)} p_j(x_i | \Theta^{(g-1)})} \quad (3)$$

M Step. The target of this step consists in finding the best set Θ of parameters that maximizes the likelihood $\mathcal{Q}(\Theta, \Theta^{(g-1)})$. By mathematical manipulations, we obtain

$$\alpha_r^{(g)} = \frac{1}{n} \sum_{i=1}^n \gamma_{ir}^{(g)} \quad \sigma_r^{z(g)} = \frac{\sum_{i=1}^n \gamma_{iz}^{(g)} (mn_i^r + 1)}{\sum_{i=1}^n \sum_{t=1}^m \gamma_{iz}^{(g)} (mn_i^t + 1)}$$

for $r = 1, \dots, m$ and $z = 1, \dots, K$.

3.1 Parameter Initialization

Although the *EM* algorithm is guaranteed to converge to a maximum, this is a *local* maximum and may not necessarily be the same as the *global* maximum. Notice, in particular, that the first iteration needs some initial values for $\alpha_k^{(0)}$ and $\sigma_c^{k(0)}$. The better the initial choices, the higher the probability that the computed local maximum is also a global maximum.

Notice that trivial initializations, such as the one in which every mixing probability $\alpha_k^{(0)} = \frac{1}{K}$ and $\sigma_c^{k(0)} = \hat{\sigma}_c$ for each cluster k , do not work, since it poses the algorithm in an equilibrium (stall) condition. In particular, the side effect of such initializations is that the parameters (and consequently the likelihood) assume initial values and do not change throughout the next iterations.

We adopt a different strategy, which combines random sampling and *k*-Means. The idea is to select k initial instances x_1, \dots, x_k from the dataset by means of a random sampling. Then, the parameters σ_j^k relative to cluster k can be initialized by exploiting the values n_i^j derived from each instance x_i , plus laplacian smoothing (to avoid the situations where $\sigma_j^k = 0$). The α_k instead are assumed equally probable. Also, the choice of the k initial points can be strengthened by multiple executions of the *k*-means algorithm on the data, and choosing the best centroids for the estimation.

3.2 Estimating the Number of Clusters

The estimation of the correct number of clusters is accomplished by resorting to a Cross-Validation approach based on a penalized Log-Likelihood principle, as described below. We aim at finding the model parameters Θ maximizing the probability $p(\Theta | X, Y)$. By Bayes' rule,

$$P(\Theta | D) = P(D | \Theta) P(\Theta)$$

In logarithmic terms,

$$\log(P(\Theta|D)) = \log P(D|\Theta) + \log P(\Theta) = \log(\mathcal{L}(\Theta|D)) + \log P(\Theta)$$

The idea in the above formula is to counterbalance two opposing requirements: the fitting of the data and the complexity of the model. The log-likelihood function, which measures the fitting of the data to the model, increases when the value K increases: in particular, it reaches its maximum when $K = n$. By the converse, the probability of the model can be encoded by resorting to the minimum description length principle, which states that simpler models are preferable to more complex ones. Thus, the probability of a model is inversely proportional to the number of its parameters (which in turn depend from the value K). In practice, $P(\Theta)$ can be modeled as an exponential distribution w.r.t the size of Θ , i.e. $P(\Theta) = \alpha n^{-km}$ where α is a normalizing factor. Thus,

$$\log(P(\Theta|D)) = \log(\mathcal{L}(\Theta|D)) - km \log n + \log \alpha$$

The evaluation strategy hence consists in computing $\log(P(\Theta|X, Y))$ for each possible model represented by Θ , and in choosing the model where it is maximal. In particular, the strategy can be summarized as follows:

1. fix the values K_{min} and K_{max} ;
2. choose the number C of cross-validation trials;
3. for each trial c :
 - sample a subset D_{train} from D ;
 - for K ranging from K_{min} to K_{max} :
 - compute $\log(P(\Theta_K|D_{train}))^c$;
4. for each K , average the values $\log(P(\Theta_K|D_{train}))^c$ over c ;
5. choose the value K^* such that $\log(P(\Theta_{K^*}|D_{train}))^{avg}$ is maximal.

4 Experimental Evaluation

Hereafter we analyze the behavior of the framework proposed in the previous section. The analysis is performed with the main objective of assessing the quality of the identified structures, i.e. whether the discovered clusters correspond to the actual homogeneous groups in the dataset.

The effectiveness issues are extensively investigated. Experiments are conducted on both real and synthesized data. The result of each experiment is a matrix D where rows and columns are associated with their cluster of membership. Hence, a partition of the matrix in blocks where each block represents a cluster can give us a visual perception of the quality of the clustering result. Ideally, a good clustering would produce a block-triangular matrix, provided that a suitable sorting of both rows and columns is produced.

The incidence matrix also enables a simple quantitative analysis, aimed at evaluating the average density within a cluster, and to compare them with the inter-cluster density (i.e., the average density of tuples and attributes outside of their cluster of membership).

In addition, for each clustering result we computed the contingency table m , in which each column represents a discovered cluster, and each row represents a true class. The term m_{ij} corresponds to the number of tuples in D that were associated with cluster \hat{x}_j and actually belongs to an ideal class C_i . Intuitively, each cluster \hat{x}_j corresponds to the partition C_i that is best represented in \hat{x}_j (i.e., such that m_{ij} is maximal).

For lack of space, in the following we only report the results on real-life datasets. The first dataset we analyze is the *SMART* collection from Cornell¹. This collection consists of 3,891 documents organized into three main sub-collections: *Medline*, containing 1033 abstracts from medical journals; *Cisi*, containing 1460 abstracts from information retrieval papers; *Cranfield*, containing 1398 abstracts from aeronautical systems papers.

Through preprocessing, we obtained a series of datasets with increasing dimensionality, where a fixed dimensionality m was obtained by choosing the m most frequent terms. The corresponding dataset was then obtained by representing each document as a binary vector.

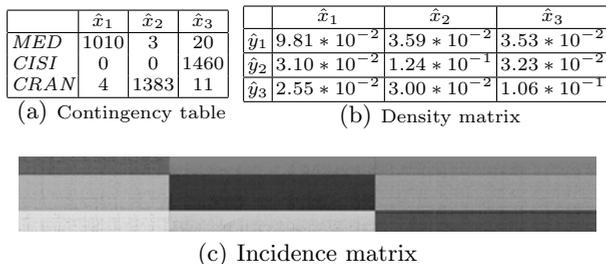


Fig. 1. Results for the *SMART* collection ($m=500$).

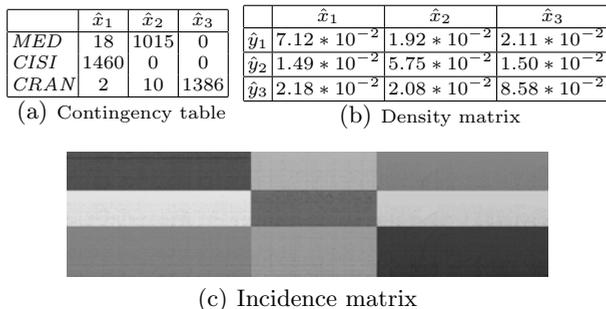


Fig. 2. Results for the *SMART* collection ($m=1K$).

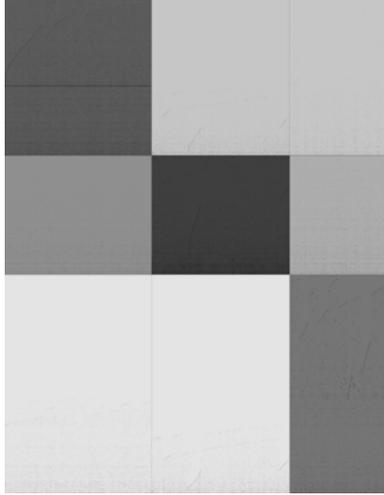
¹ <ftp://ftp.cs.cornell.edu/pub/smart>

	\hat{x}_1	\hat{x}_2	\hat{x}_3
<i>MED</i>	15	2	1016
<i>CISI</i>	1460	0	0
<i>CRAN</i>	7	1391	0

	\hat{x}_1	\hat{x}_2	\hat{x}_3
\hat{y}_1	$1.96 * 10^{-2}$	$4.79 * 10^{-3}$	$4.65 * 10^{-3}$
\hat{y}_2	$8.11 * 10^{-3}$	$3.39 * 10^{-2}$	$7.80 * 10^{-3}$
\hat{y}_3	$2.68 * 10^{-3}$	$2.54 * 10^{-3}$	$1.41 * 10^{-2}$

(a) Contingency table

(b) Density matrix



(c) Incidence matrix

Fig. 3. Results for the *SMART* collection ($m=5k$).

	\hat{x}_1	\hat{x}_2	\hat{x}_3
<i>MED</i>	4	1018	11
<i>CISI</i>	0	2	1458
<i>CRAN</i>	1387	0	11

	\hat{x}_1	\hat{x}_2	\hat{x}_3
\hat{y}_1	$2.16 * 10^{-2}$	$4.86 * 10^{-3}$	$5.04 * 10^{-3}$
\hat{y}_2	$1.39 * 10^{-3}$	$7.46 * 10^{-3}$	$1.44 * 10^{-3}$
\hat{y}_3	$2.10 * 10^{-3}$	$2.04 * 10^{-3}$	$9.98 * 10^{-3}$

(a) Contingency table

(b) Density matrix



(c) Incidence matrix (inverted)

Fig. 4. Results for the *SMART* collection ($m=10K$).

Figures 1, 2, 3 and 4 show the clustering results for increasing values of m . As we can see, compactness and separability are quite good, as also testified

by the density matrices and contingency tables. Also, the proposed approach is effective to a large dimensionality in the number of attributes. In particular, results are quite more robust than those obtained with the ITCC co-clustering algorithm [7] (an example is reported in figures 5). The latter, indeed, allow high quality results only by fixing a high number of clusters in the Y dimension.

	\hat{x}_1	\hat{x}_2	\hat{x}_3
\hat{y}_1	$6.97 * 10^{-4}$	$1.50 * 10^{-2}$	$2.52 * 10^{-4}$
\hat{y}_2	$1.30 * 10^{-2}$	$4.21 * 10^{-2}$	$1.26 * 10^{-2}$
\hat{y}_3	$5.36 * 10^{-3}$	$2.28 * 10^{-5}$	$6.10 * 10^{-5}$
\hat{y}_4	$1.48 * 10^{-2}$	$9.89 * 10^{-3}$	$3.03 * 10^{-3}$
\hat{y}_5	$1.06 * 10^{-2}$	$1.74 * 10^{-2}$	$2.03 * 10^{-2}$
\hat{y}_6	$1.33 * 10^{-2}$	$2.74 * 10^{-3}$	$6.10 * 10^{-3}$
\hat{y}_7	$6.72 * 10^{-5}$	$3.07 * 10^{-5}$	$6.02 * 10^{-3}$
\hat{y}_8	$5.05 * 10^{-3}$	$3.09 * 10^{-3}$	$1.71 * 10^{-2}$

	\hat{x}_1	\hat{x}_2	\hat{x}_3
<i>MED</i>	980	2	51
<i>CISI</i>	0	0	1460
<i>CRAN</i>	1	1390	7

(a) Contingency table

(b) Density matrix



(c) Incidence matrix (inverted)

Fig. 5. ITCC Results for the *SMART* collection ($m=10K$).

A further dataset we analyse is *Internet Ads.*, available from the UCI Machine Learning repository². The dataset contains 3,279 records and 1,554 boolean attributes. In addition, three further attributes are "categorical" in nature (although they are numeric, several values occur frequently). To summarize, the total number of possible items is 2832. This dataset represents a set of possible advertisements on Internet pages; each record represents a web page, and the features encode phrases occurring in the URL, the image's URL and alt text, the anchor text, and words occurring near the anchor text. Each record is labelled either as '*ad*' or as '*noad*'. The dataset is quite unbalanced, since there are 2,821 '*noads*' and 458 '*ads*'. Notwithstanding, separability is quite good, as it can be seen from figure 6. In particular, notice how clusters \hat{x}_7 and \hat{x}_8 represent the minority class.

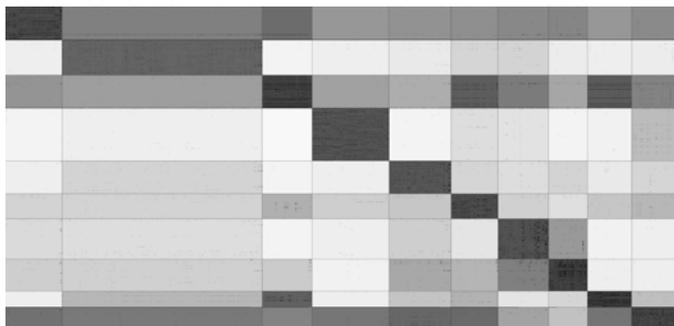
² <http://www.ics.uci.edu/mllearn/MLRepository.html>

	\hat{x}_1	\hat{x}_2	\hat{x}_3	\hat{x}_4	\hat{x}_5	\hat{x}_6	\hat{x}_7	\hat{x}_8	\hat{x}_9	\hat{x}_{10}
<i>ad</i>	13	59	17	3	54	8	147	151	6	1
<i>noad</i>	262	917	225	372	247	220	99	37	208	233

(a) Contingency table

	\hat{x}_1	\hat{x}_2	\hat{x}_3	\hat{x}_4	\hat{x}_5	\hat{x}_6	\hat{x}_7	\hat{x}_8	\hat{x}_9	\hat{x}_{10}
\hat{y}_1	$8.74 * 10^{-2}$	$9.02 * 10^{-3}$	$1.11 * 10^{-2}$	$6.22 * 10^{-3}$	$6.72 * 10^{-3}$	$6.94 * 10^{-3}$	$7.37 * 10^{-3}$	$8.82 * 10^{-3}$	$6.29 * 10^{-3}$	$7.14 * 10^{-3}$
\hat{y}_1	$3.87 * 10^{-4}$	$1.48 * 10^{-2}$	$1.22 * 10^{-4}$	$3.15 * 10^{-4}$	$7.27 * 10^{-4}$	$1.34 * 10^{-3}$	$1.44 * 10^{-3}$	$4.09 * 10^{-4}$	$3.31 * 10^{-4}$	$4.80 * 10^{-4}$
\hat{y}_1	$6.43 * 10^{-3}$	$5.86 * 10^{-3}$	$1.19 * 10^{-1}$	$5.71 * 10^{-3}$	$5.17 * 10^{-3}$	$2.67 * 10^{-2}$	$9.24 * 10^{-3}$	$5.45 * 10^{-3}$	$4.10 * 10^{-2}$	$8.97 * 10^{-3}$
\hat{y}_1	$1.85 * 10^{-4}$	$3.73 * 10^{-4}$	$4.86 * 10^{-5}$	$5.34 * 10^{-2}$	$1.43 * 10^{-4}$	$1.16 * 10^{-3}$	$9.08 * 10^{-4}$	$1.66 * 10^{-4}$	$2.01 * 10^{-4}$	$3.01 * 10^{-3}$
\hat{y}_1	$3.20 * 10^{-4}$	$1.47 * 10^{-3}$	$1.29 * 10^{-4}$	$3.35 * 10^{-4}$	$4.14 * 10^{-2}$	$1.37 * 10^{-3}$	$1.12 * 10^{-3}$	$1.43 * 10^{-3}$	$7.34 * 10^{-4}$	$1.37 * 10^{-3}$
\hat{y}_1	$1.41 * 10^{-3}$	$1.47 * 10^{-3}$	$4.98 * 10^{-3}$	$1.74 * 10^{-3}$	$1.97 * 10^{-3}$	$6.80 * 10^{-2}$	$1.51 * 10^{-3}$	$1.09 * 10^{-3}$	$1.93 * 10^{-3}$	$3.39 * 10^{-3}$
\hat{y}_1	$1.23 * 10^{-3}$	$1.13 * 10^{-3}$	$1.27 * 10^{-4}$	$3.14 * 10^{-4}$	$1.72 * 10^{-3}$	$8.77 * 10^{-4}$	$5.60 * 10^{-2}$	$6.00 * 10^{-3}$	$2.63 * 10^{-4}$	$3.06 * 10^{-4}$
\hat{y}_1	$1.78 * 10^{-3}$	$1.57 * 10^{-3}$	$1.81 * 10^{-3}$	$2.23 * 10^{-4}$	$5.37 * 10^{-3}$	$3.76 * 10^{-3}$	$9.04 * 10^{-3}$	$1.22 * 10^{-1}$	$3.61 * 10^{-4}$	$3.03 * 10^{-4}$
\hat{y}_1	$3.34 * 10^{-4}$	$3.39 * 10^{-3}$	$2.54 * 10^{-2}$	$3.50 * 10^{-4}$	$2.09 * 10^{-3}$	$4.55 * 10^{-3}$	$8.55 * 10^{-4}$	$1.32 * 10^{-3}$	$1.51 * 10^{-1}$	$2.98 * 10^{-3}$
\hat{y}_1	$1.01 * 10^{-2}$	$9.36 * 10^{-3}$	$8.92 * 10^{-3}$	$9.76 * 10^{-3}$	$1.40 * 10^{-2}$	$1.19 * 10^{-2}$	$5.59 * 10^{-3}$	$2.30 * 10^{-3}$	$9.87 * 10^{-3}$	$1.09 * 10^{-1}$

(b) Density Matrix



(c) Incidence matrix

Fig. 6. Results for the *Internet ads* dataset.

5 Conclusions and Future Works

In this paper, we defined a novel EM-based approach to the discovery of co-clusters in a high-dimensional setting. This exploits the joint probability distribution over row and column variables associated to the data co-occurrence matrix, in order to initially find row clusters. Then, the conditional distribution of attributes over tuples is exploited to discover actual co-clusters, i.e. for associating concept (i.e. column) clusters with row clusters. We studied the behavior of our algorithm and compared it against the performance of a well-known *ad hoc* co-clustering scheme. The empirical results of a preliminary evaluation show the effectiveness of our approach and, apparently, suggest that natural co-clusters can still be discovered by tuning a mono-dimensional clustering strategy.

Still, the proposed approach is based on a naive assumption that tuple clusters are associated with exactly a concept cluster. Although this assumption seems to work well in practice, it appears nevertheless a strong requirement, which is hence likely to miss some latent sub-concepts actually holding in the data. As a future development, we plan to investigate the extension of the proposed framework in order to enable multiple characterizations of a same tuple cluster, in terms of corresponding associations with as many concept clusters.

References

1. R. Agrawal, J. Gehrke, D. Gunopulos, and P. Raghavan. Automatic subspace clustering of high dimensional data for data mining applications. In *Proc. ACM SIGMOD'98 Conf.*, pages 94 – 105, 1998.
2. D. Barbará, J. Couto, and Y. Li. COOLCAT: an entropy-based algorithm for categorical clustering. In *Proc. ACM Conf. on Information and Knowledge Management (CIKM'02)*, pages 582–589, 2002.
3. A. Califano, G. Stolovitzky, and Y. Tu. Analysis of gene expression microarrays for phenotype classification. In *Proc. of the 8th Int. Conf. on Intelligent Systems for Molecular Biology*, pages 75–85, 2000.
4. Y. Cheng and G. M. Church. Biclustering of expression data. In *Proc. of the 8th Int. Conf. on Intelligent Systems for Molecular Biology*, pages 93–103, 2000.
5. G. Costa, F. Folino, G. Manco and R. Ortale. A Hierarchical Probabilistic Model for Co-Clustering High-Dimensional Data. Technical Report n.4, ICAR-CNR, 2006. Available at <http://biblio.cs.icar.cnr.it/biblio/>.
6. S.C. Deerwester et al. Indexing by latent semantic analysis. *Journal of American Society of Information Science*, 41(6), 1990.
7. I. S. Dhillon, S. Mallela, and D. S. Modha. Information-theoretic co-clustering. In *Proc. of the 9th ACM SIGKDD Int. Conf. on Knowledge Discovery and Data Mining (KDD)*, pages 89–98, 2003.
8. G. Getz, E. Levine, and E. Domany. Coupled two-way clustering analysis of gene microarray data. In *Proc. of the Natural Academy of Sciences USA*, pages 12079–12084, 2000.
9. J. A. Hartigan. Direct clustering of a data matrix. *Journal of the American Statistical Association (JASA)*, 67(337):123–129, 1972.
10. L. Lazzeroni and A. Owen. Plaid model for gene expression data. technical report, 2000.
11. A. McCallum, K. Nigam, and L. H. Ungar. Efficient clustering of high-dimensional data sets with application to reference matching. In *Proc. 6th Int. Conf. on Knowledge Discovery and Data Mining (KDD'00)*, pages 169 – 178, 2000.
12. G. McLachlan and D. Peel. *Finite Mixture Models*. Wiley, 2000.
13. L. Parsons, E. Haque, and H. Liu. Subspace Clustering for High Dimensional Data: A Review. *ACM SIGKDD Explorations*, 6(1):90 – 105, 2004.
14. S.K. Selim and M.A.Ismail. Biclustering algorithms for biological data analysis: A survey. *Transactions on Computational Biology and Bioinformatics (TCBB)*, 1(1):24 – 45, 2004.
15. A. Tanay, R. Sharan, and R. Shamir. Discovering statistically significant biclusters in gene expression data. *Bioinformatics*, 18(1):S136–S144, 2002.
16. C. Tang, Li. Zhang, I. Zhang, and M. Ramanathan. Interrelated two-way clustering: an unsupervised approach for gene expression data analysis. In *Proc. of the 2nd IEEE Int. Symposium on Bioinformatics and Bioengineering*, pages 41–48, 2001.
17. H. Wang, Wei Wang, J. Yang, and P. S. Yu. Clustering by pattern similarity in large data sets. In *Proc. of the 2002 ACM SIGMOD Int. Conf. on Management of Data*, pages 394–405, 2002.
18. G. Govaert, M. Nadif. An EM Algorithm for the Block Mixture Model. In *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(4): 643 – 647, 2005.
19. A.P. Dempster, N.M. Laird, D.B. Rubin. Maximum Likelihood from Incomplete Data via EM Algorithm. In *J. Royal Statistical Society*, 39(B): 1 – 38, 1977.

Translating Context Data in Adaptive Web Applications

Roberto De Virgilio and Riccardo Torlone

Dipartimento di Informatica e Automazione
Università degli studi Roma Tre
{devirgilio,torlone}@dia.uniroma3.it

Abstract. Recently, the literature proposes many approaches to the interoperability of Web data based on the used of formal models. We consider Adaptive Web Applications, in which a relevant requirement is the ability to capture and manipulate context information. Since context data are often heterogeneous, their translations from one representation into another is an important issue that needs to be addressed to facilitate their integration. In this paper we describe a general framework supporting the representation of a large variety of context information and the translation between different formats. Translations are specified as compositions of elementary steps, defined by means of special operations. With these operations, we show how it is possible to process automatically a translation on the models that are provided as source and target.

1 Introduction

The increasing popularity of mobile devices, such as laptops, mobile phones, and personal digital assistants is enabling new classes of applications targeting environments characterized by being dynamic, mobile, reconfigurable, and personalized spontaneously. These applications and their targeted environments raise challenging problems for application developers, as they have to be aware of the variations in the execution context such as location, time, users' activities, and devices' capabilities in order to tune and adapt the behavior and functionalities of applications. In adaptive web system, it is widely recognized that the management of context information is a fundamental requirement to take into account the limited resources of mobile systems, to select data relevant to the user, to improve the interoperability with the environment, and, in general, to make the interaction with the system truly adaptive to highly change scenarios of use.

This scenario changes the role of context information and semantics as compared to traditional information systems [8, 10], as now the physical environment immediately affects and interacts with the processing of data and communication. Unfortunately, current technologies do not fully support flexible and self-adapting models based on context. For example, if a mobile user today wants to use the computing resources of a new environment, he/she has to obtain the necessary information, assess it (format, semantics) and figure out manually how to continue his/her activities with the local resources of that new environment. This is unacceptable in pervasive computing environment and neglects the advances which have been made in other research domains dealing with

context information and semantics. Let's consider the architecture of reference in Figure 1. The *Context manager* has to integrate information expressed in different and possibly heterogeneous formats such as textual HTTP headers or more structured CC/PP or CSCP documents serialized in RDF/XML. The *Adaptation manager* has to produce the adaptation that best fits the context requirements and the *Response generator* has to produce the final response based on the adaptation of the contents. In this framework, the interoperability of such applications is an important task. Interoperability in

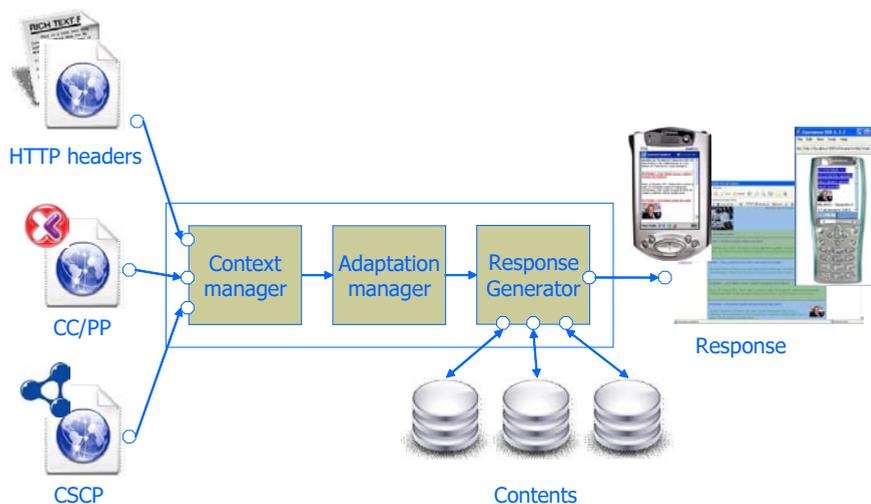


Fig. 1. An architecture of reference

heterogeneous environments is ultimately achieved via shared metadata and the overall strategy for sharing and understanding metadata consists of the automated development, publishing, management, and interpretation of models [1, 2, 6].

The most relevant context modeling approaches identify a context as a set of *profiles* [12]. It is important to understand how the set of context models of interest can be defined, in a simple way, and the individual models be specified. The Object Management Group has introduced a number of important standards such as MOF [4], UML [11], XMI [5] and CWM [9] for data modeling in various research areas. Generally the MOF is used for modeling the adaptation specifications of a web application rather than context information, often considered trivial input values. To this aim, in [3] we studied the notion of *generic profile* and we represented contexts by means of the *General Profile Model* (GPM), a conceptual model for the uniform description of the various aspects of a context. Based on GPM, we proposed a rule-based *conversion* process to translate profiles from a representation into another. However the web designer has to define and select manually the different and right translations between each couple of representations. This can be a critical problem for the interoperability of adaptive systems, because many manipulations of context information are needed. In the plethora of con-

text representations, different models exist, often just small variations of other ones. Many uses of a context involve managing the change in models and the transformation of data from one model into another. The set of possible models is potentially huge, as they are obtained by variants of constructs. It is probably the case that some combinations of variants are not meaningful, but with more constructs and more variants for each of them, we get a combinational explosion of the number of models. With this size for the space of context models, it would be hopeless to have translations between every pairs of models, as with n models we would need n^2 translations. If we assume that our GPM generalizes all the other models then we need one translation for each model (from GPM) but n can be still an unmanageable number. As a consequence it becomes meaningful to specify translations with reference to them.

This paper proposes a middleware data model that serves as a basis for the task. Our formalism permits to define different *Profile Models* to describe the primitives involved and the structure of a context model. Then we provide a mechanism to generate automatically a conversion from a Profile Model into another. The idea is to have many "basic" translations that perform *elementary steps* and can be combined to form actual conversions, but with a lot of reusability. With fine-grained decompositions, the basic steps can be reused in many other conversions. In this work, we illustrate the notion of *Mod* operation that executes an elementary step. Therefore a major issue arises: given a set of basic translations, how do we build the actual conversion we need? Or, at least, how do we verify that a given sequence of basic translations produces the model we are interested in? The idea is that all the *Mod* operations are assumed to be correct, and so, if we properly apply sequence of them, we obtain correct translations (this is sometimes called the "axiomatic approach" [1]).

The rest of the paper is organized as follows. In Section 2, we illustrate the basic notions of our context modeling approach, defining our formalism and the notions of general profile. In Section 3 we illustrate the notions of Profile Model and *Mod* operation and describe the conversion process. In Section 4 we show a practical implementation. Finally, in Section 5 we draw some conclusions and sketch future work.

2 General Profiles

A *general profile* is a description of an autonomous aspect of the context into which the Web site is accessed and that should influence the structuring and presentation of its contents. Examples of profiles are descriptions of the user, the device, the location, and so on. Our formalism presents a quite limited set of constructs, that we call *basic primitives*, to describe, in a graphical way, a conceptual representation of a context, as shown in Figure 2. Principal constructs are the *dimension* and the *attribute*. A *dimension* is a property that characterizes a profile. Each dimension is described by means of a set of *attributes*. Attributes can be *simple* or *composite*. A simple attribute has a domain of values associated with it (printable values such as string, integer, boolean and so on), whereas a composite attribute has a set of (simple or composite) attributes associated with it. It is possible to distinguish two roles for a simple attribute; it can be a *key* or an *external reference* to a component of a profile. It is possible to represent *ordered* and *unordered sequences* of attributes and a *choice* of a set of attributes, whose

basic primitives								
Profile (P)	Dimension (D)	Complex Attribute (cA)	Simple Attribute (sA)	Ordered sequence	Unordered sequence	Choice	Key (K)	External Reference (eR)
								

Fig. 2. basic primitives

instances can be chosen among instances of the attributes (for instance a <choice> of an XML-schema). Finally the *cardinality* is expressed as a pair of integer values (*Min,Max*) that corresponds to a primitive whose instances are sets of instances of the primitive associated with it (these sets must have a cardinality included between *Min* and *Max*).

Definition 1 (Profile and context) Given a set of dimensions D_1, \dots, D_n over a set of attributes $A_{i,1} \dots A_{i,k_i}$ ($1 \leq i \leq n$) respectively, a (general) profile over D_1, \dots, D_n is function that associates with each simple attribute of every dimension a value taken from its domain. A context is a collection of profiles.

As an example, Figure 3 reports a graphical representation for the context schema of a client *A* composed by profile schemes for the user and the location. For instance, a

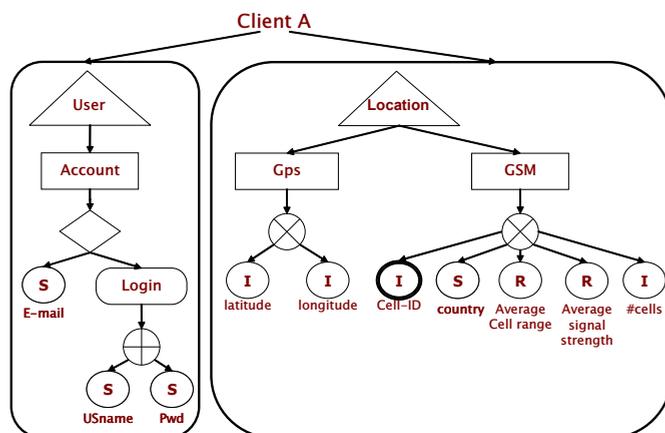


Fig. 3. Example of context

user profile of a client can be represented by means of the dimension account, described by a choice between the simple attribute e-mail (so an access without a registration) or the complex attribute login, composed by the simple attributes username and password

(so an access with a registration), and so on. The location profile presents the dimension GSM to characterize location information from the GSM cells: each cell has a unique ID, expressed by the key Cell-ID. We indicate the base type associated inside the attribute (for instance I means integer, R real and S string).

3 Conversion of Contexts

Given a source profile PI_s of a profile schema PS_s , described according to a Profile Model PM_1 , we need to generate a *target* profile (schema) PI_t (PS_t) described according to a Profile Model PM_2 , containing the same information as PI_s . This is done by a *conversion* (C) of PI_s (PS_s) from PM_1 into PM_2 , denoted as $PI_t = C_{PM_1 \Rightarrow PM_2}(PI_s)$ ($PS_t = C_{PM_1 \Rightarrow PM_2}(PS_s)$). In this section we present a mechanism to generate automatically a conversion through special *Mod* operations.

3.1 Profile Model

Through our formalism, we want to describe the model of a profile by means of the constructs involved and its structure. To this aim we introduce the following notion of *Profile Model*.

Definition 2 (Profile Model) *We define a Profile Model PM as a special Multi-Graph $\langle L, V, E, \omega, v_c \rangle$ where*

- L is the set of labels, representing the basic primitives of our formalism such as $\{\Delta, \bigcirc, \dots\}$
- V is the set of vertexes $\{v_1, v_2, \dots\}$, each one represented by a pair (OID, l) , where OID is the identifier and $l \in L$ is the label of the vertex
- E is the set of direct weighted edges (v_i, v_j) that means " v_i is composed by v_j ". The graph allows self loops such that $v_i = v_j$
- ω is the function to assign a weight to an edge in E . A weight on an edge (v_i, v_j) is a pair (min, max) that represents the type of cardinality of v_j respect to v_i . The weights admissible are in $\{(0, 1), (1, 1), (0, n), (1, n), (n, n)\}$ and it is possible to establish an ordering between two weights $\omega_1 = (a, b)$ and $\omega_2 = (c, d)$ such that $\omega_1 \leq \omega_2$ if $a \leq c$ and $b \leq d$, where it is $0 < 1 < n$.
- v_c is the vertex (OID, Δ) representing the center of the graph.

A Profile Model would describe the representation used for a profile by means of primitives involved and how these primitives are combined (or composed) in the profile. For instance, Figure 4 shows the Profile Model PM_1 . The model can articulate a profile in several dimensions. Each dimension can be composed by simple attributes of string or integer values, or by composite attributes. Each composite attribute can be composed by simple attributes of string or integer values or, recursively, by composite attributes of the same type. In PM_1 , the function ω assign to each edge the type of cardinality (I, n) : each component can be articulated by other components using the cardinality (I, I) or (I, n) . Figure 4 shows an example of profile P described by PM_1 . In a profile schema the cardinality of default is (I, I) . In a Profile Model each vertex is identified by an OID ,

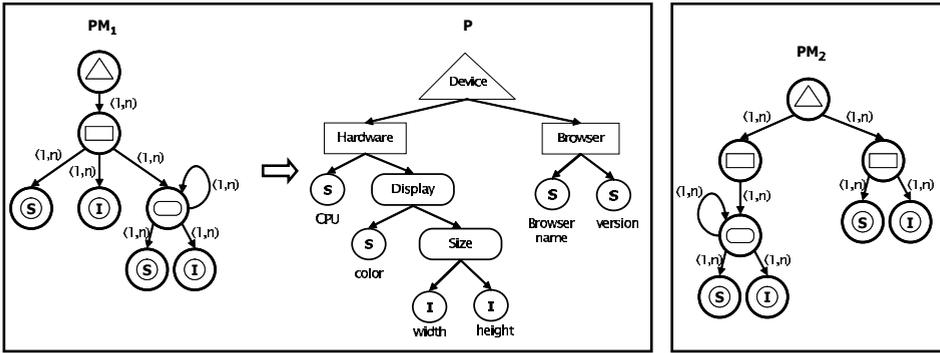


Fig. 4. An example of Profile Model

for instance generated by using Skolem functions [7], and characterized by a label, that represents a basic primitive. So the set L represents the basic primitives involved in a profile: for instance in the previous example we have dimensions, simple and composite attributes, and basic types such as string and integer. The set E represents how the primitives involved are composed in a profile. The function ω describes the minimum and maximum cardinality to articulate a component.

A fundamental aspect is that different Profile Models can be compared making use of a subsumption relationship, denoted by \triangleleft . Intuitively, a Profile Model PM_1 subsumes a Profile Model PM_2 if PM_1 represents all the types of profiles that PM_2 can represent, and more others. More precisely, we first say that two vertexes $v_1 = (OID_1, l_1)$ and $v_2 = (OID_2, l_2)$ are similar, denoted by $v_1 \approx v_2$, if $l_1 = l_2$. Then two paths of vertexes are similar if the vertexes of the pathes, in order, are similar. The subsumption relationship is then defined as follows.

Definition 3 (Subsumption of Profile Models) *Given two Profile Models $PM_1 = \langle L_1, V_1, E_1, \omega_1, v_{c_1} \rangle$ and $PM_2 = \langle L_2, V_2, E_2, \omega_2, v_{c_2} \rangle$, we say that PM_1 is subsumed by PM_2 , $PM_1 \triangleleft PM_2$, if for each vertex $v_1 \in V_1$ it exists a vertex $v_2 \in V_2$ such that for each edge $(v_1, v') \in E_1$ it exists the edge $(v_2, v'') \in E_2$ so that $(v_1, v') \approx (v_2, v'')$ and $\omega_1((v_1, v')) \leq \omega_2((v_2, v''))$.*

As an example, given the Profile Models reported in Figure 4, we have that $PM_2 \triangleleft PM_1$.

We can define *meet*, *join* and *difference* between Profile Models. Intuitively, given two Profile Models PM_1 and PM_2 , the *meet* (\sqcap) represents the greatest Profile Model that is able to generate the productions (types of profile) in common between PM_1 and PM_2 , the *join* (\sqcup) the least Profile Model that is able to generate all the productions of PM_1 and PM_2 , and the *difference* ($-$) the greatest Profile Model that is able to generate the productions of PM_1 not in common with PM_2 .

Definition 4 (Meet of Profile Models) *The meet of two Profiles PM_1 and PM_2 , denoted by $PM_1 \sqcap PM_2$, is a Profile Model PM such that, $PM \triangleleft PM_1$, $PM \triangleleft PM_2$ and, for each Profile Model $PM' \neq PM$ such that $PM' \triangleleft PM_1$, $PM' \triangleleft PM_2$, it is the case that $PM' \triangleleft PM$.*

Definition 5 (Join of Profile Models) The join of two Profile Models PM_1 and PM_2 , denoted by $PM_1 \sqcup PM_2$, is a Profile Model PM such that, $PM_1 \triangleleft PM$, $PM_2 \triangleleft PM$ and, for each Profile Model $PM' \neq PM$ such that $PM_1 \triangleleft PM'$, $PM_2 \triangleleft PM'$, it is the case that $PM \triangleleft PM'$.

Definition 6 (Difference of Profile Models) The difference of two Profile Models PM_1 and PM_2 , denoted by $PM_1 - PM_2$, is a Profile Model PM such that, $PM \triangleleft PM_1$, $PM \cap (PM_1 \cap PM_2) = \emptyset$ and, for each Profile Model $PM' \neq PM$ such that $PM' \triangleleft PM_1$, it is the case that $PM' \triangleleft PM$.

Figure 5 shows some example of meet, join and difference.

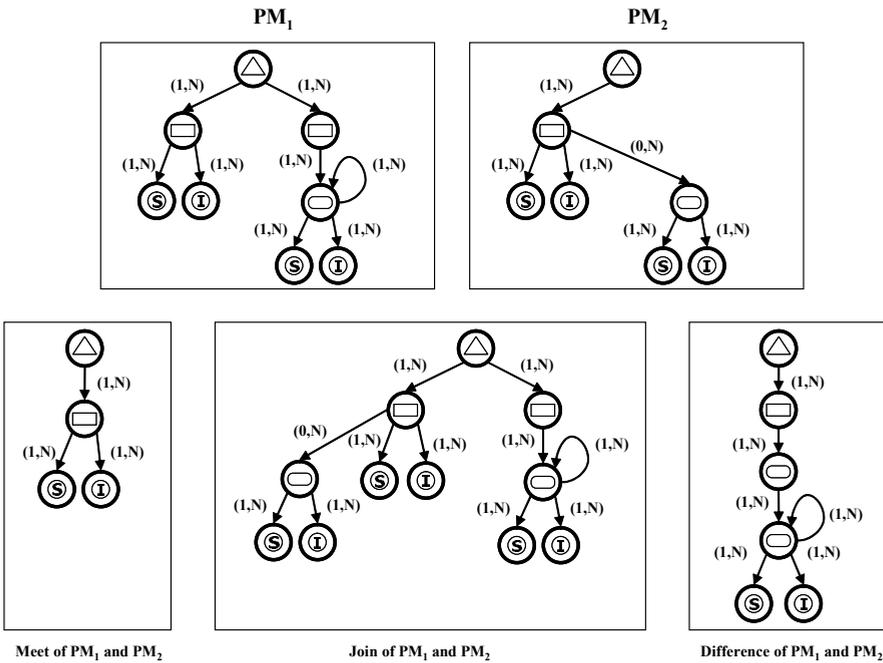


Fig. 5. Example of operators (\cap) , (\sqcup) and $(-)$

The comparison between two Profile Models can be measured by the following idea of distance.

Definition 7 (Distance between Profile Models) Given two Profile Models $PM_1 = \langle L_1, V_1, E_1, \omega_1, v_{c1} \rangle$ and $PM_2 = \langle L_2, V_2, E_2, \omega_2, v_{c2} \rangle$, and the difference $PM = PM_2 - PM_1 = \langle L, V, E, \omega, v_c \rangle$, the distance of PM_1 from PM_2 , $\delta(PM_1, PM_2)$, is given by $|L_2 - L_1| + |E|$.

The distance $\delta(PM_1, PM_2)$ measures how much the Profile Model PM_1 can't interpret the Profile Model PM_2 . In other words, this distance evaluates how many primitives and combinations of them in PM_2 are not comprehensible by PM_1 . For instance,

in the Figure 5 the distance of PM_2 from PM_1 , $\delta(PM_2, PM_1)$, is 6. In this case PM_2 can't interpret the composition of composite attributes in other ones (the self-loop).

3.2 Mod (Γ)

An operation $Mod(\Gamma)$ takes as input a profile (schema) P_1 according to a Profile Model PM_1 and returns a profile (schema) P_2 according to a Profile Model PM_2 containing the same information of P_1 , denoting such as $\Gamma_{PM_1 \rightarrow PM_2}(P_1) = P_2$. A Mod operates an elementary translation of a profile (schema) from a representation into another, for instance of some primitives. For instance the Figure 6 represents a Mod operation to translate a profile schema from a Profile Model that articulates the schema on n levels, making use of composite attributes, into a Profile Model that articulates the schema on 2 levels, making only use of simple attributes (and external references) and no composite ones. Intuitively given a Mod operation $\Gamma_{PM_i \rightarrow PM_j}$, if we apply the operation to a

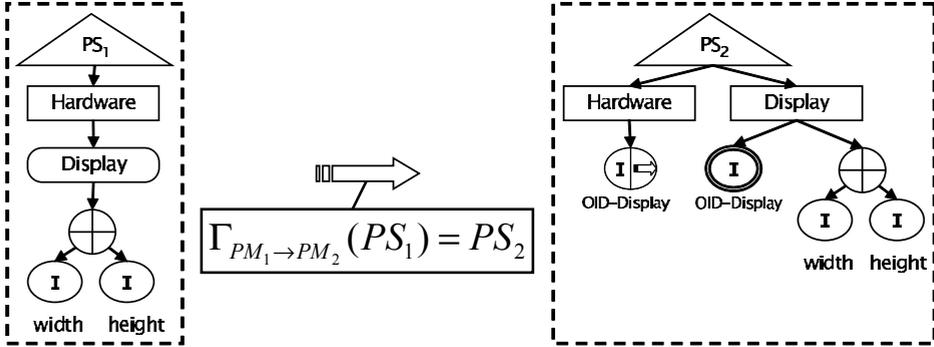


Fig. 6. An example of Mod operation

profile P described by a Profile Model PM such that $PM_i \triangleleft PM$ then the operation will translate the part P' of P described by PM_i into P'' described by PM_j . The resulting profile will be described by a Profile Model such as $(PM - PM_i) \sqcup PM_j$. So we can denote also the following $\Gamma_{PM_1 \rightarrow PM_2}(PM) = (PM - PM_1) \sqcup PM_2$.

Let's consider a prefixed set of Mod operations $F = \{\Gamma_{PM_1 \rightarrow PM_2}, \Gamma_{PM_3 \rightarrow PM_4}, \dots\}$. From F , we can define a special graph $MG = \gamma(F)$, as follows

Definition 8 (Graph of Profile Models) We define Model Graph on a set of Mod operations F as a graph $MG = \{N_{MG}, E_{MG}\}$ not connected, directed and acyclic, where

- N_{MG} is the set of nodes representing Profile Models
- $E_{MG} = \{L_{MG}, U_{MG}\}$, where
 - L_{MG} is the set of direct edges (n_i, n_j) such that $\exists \Gamma_{n_i \rightarrow n_j} \in F$
 - U_{MG} is the set of dashed edges (n'_i, n'_j) such that $n'_i \triangleleft n'_j$

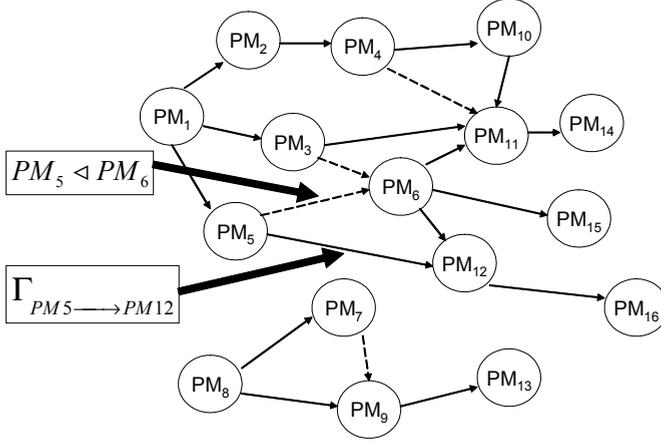


Fig. 7. An example of Model Graph

Figure 7 shows an example. Moreover, we introduce the following notion of *grafted node* in a Model Graph MG , to compare a Profile Model PM with MG .

Definition 9 (Grafted Node) *Given a model graph MG and a model set PM , a node $PM' \in N_{MG}$ is a grafted node respect to PM if $PM' \triangleleft PM$ and we say that PM is comparable with MG .*

In MG a path $\{PM_1, PM_2, \dots, PM_{n-1}, PM_n\}$ represents a set of elementary translations that we can apply sequentially to a Profile Model PM comparable with MG , such that $\Gamma_{PM_{n-1} \rightarrow PM_n}(\dots(\Gamma_{PM_1 \rightarrow PM_2}(PM)))$. For instance in Figure 8 the Profile Model PM , comparable with the Model Graph through the grafted nodes PM_1 , PM_6 and PM_8 , defines three paths: if we select the path T_1 the resulting Profile Model is $T_1(PM)$.

3.3 Automatic generation of a Conversion

Given a set of Mod operations F and two Profile Models PM_1 and PM_2 , a conversion \mathcal{C} from PM_1 to PM_2 is performed by a sequence of Mod operations from F .

Definition 10 (Conversion) *Given two Profile Models PM_1 and PM_2 and a set of Mod operations F , a conversion \mathcal{C} from PM_1 into PM_2 is a sequence $\{\Gamma_1, \Gamma_2, \dots, \Gamma_n\} \in F$ such that $\delta(\Gamma_n(\Gamma_{n-1}(\dots(\Gamma_1(PM_1))))), PM_2) = 0$.*

We have to select the sequence of Mod operations of F such that the Profile Model resulting by applying the sequence to PM_1 has distance zero from PM_2 . Therefore a conversion \mathcal{C} from PM_1 into PM_2 through F can be generated automatically as follows.

1. we generate a Model Graph MG from F . For the nature of the Mod operations, the Model Graph is not connected, directed and has to be acyclic.

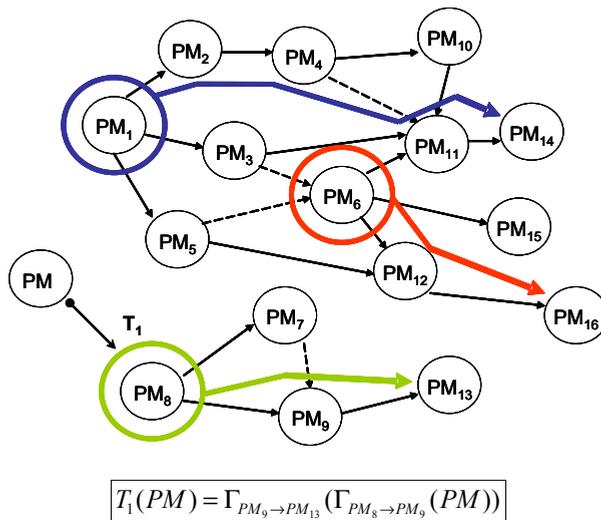


Fig. 8. Elementary Translations in a Model Graph

2. we set the *current* Profile Model PM_c to PM_1 , the *target* Profile Model PM_t to PM_2 and the sequence S of Mod operation to \emptyset .
3. we have to select one or more paths in MG . So we search in MG the set of grafted nodes $Gn = \{gn_1, gn_2, \dots, gn_k\}$ respect to PM_c , ordered respect the increasing distance from PM_t ;
4. we operate a backtracking strategy on each gn_i of Gn .
5. we select a path p of MG starting from gn_i as follows: (i) from a node select the outgoing edge that applied to PM_c returns the Profile Model with the lowest distance from PM_t , (ii) stop the selection if we arrive in a node without outgoing edges or each outgoing edges makes greater the distance from PM_t .
6. add the selected path p to S and recalculate with p the new PM_c
7. if $\delta(PM_c, PM_t) = 0$ then return S , otherwise select a new set of grafted nodes in MG respect to PM_c and iterate the backtracking.
8. the fault condition of the backtracking is to obtain a set of grafted node empty. So we go back considering S without the last path added and selecting another grafted node, as shown in Figure 9.
9. if it is not possible to find a sequence S such that $\delta(PM_c, PM_t) = 0$, then the web engineer has to write the missing elementary translations and update the set F .

4 Implementation

It was extended the tool presented in [3], introducing a module to define Profile Models. Figure 10 shows a screenshot of the user-friendly interface. The Mod operations are written as production rules by the syntax described in [3]. Profile Models are implemented using RDF/XML syntax and the Mod operations as a set of RDF inference rules by the Jena framework (<http://jena.sourceforge.net/inference/>).

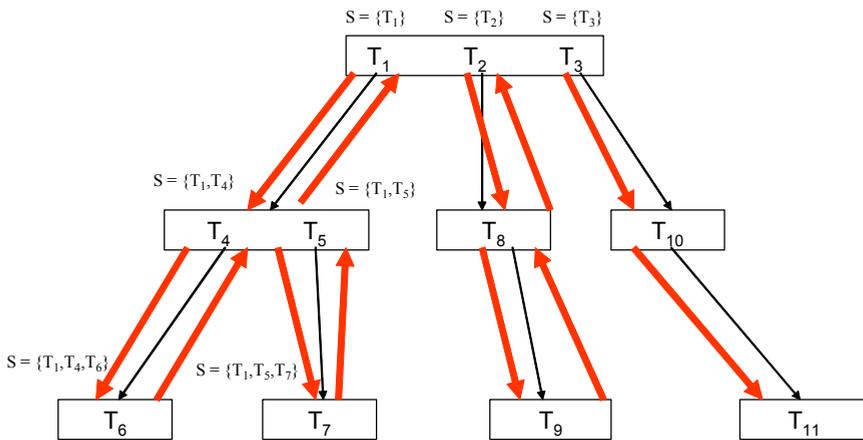


Fig. 9. The backtracking strategy

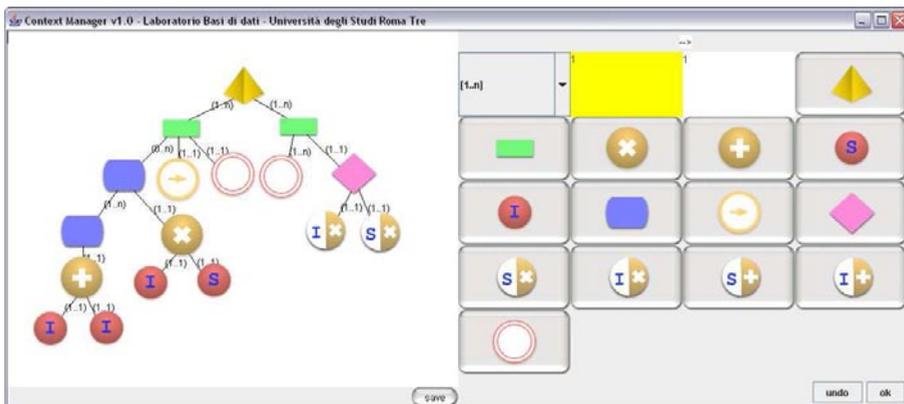


Fig. 10. User-interface of the tool

5 Conclusions and Future Work

In this paper we proposed a data model that consists of a general abstraction of existing formats that allows to define different context models. The data model proposed is enriched by special operators (Mod) to embed the conversion of context information from one representation into another. The conversion process is resulting by a composition of elementary translation steps. We illustrated a technique to reason and generate automatically a conversion by the form of a sequence of Mod operations. The results presented in this paper are subject of further conceptual and practical investigation. From a conceptual point of view we believe that it is possible to define a framework for *context management*, as a new approach to manipulating context information. We are defining an algebra of operators (a lattice) to embed the main functionalities of context management. This framework can be an important support in different application scenarios such as contexts translation, contexts integration or contexts classification by means of clusters. From a practical point of view, an important issue strongly suggests that an implementation of context management would provide major programming productivity gains for a wide variety of context management problems. Of course, to make this claim compelling, an implementation is needed.

References

1. P. Atzeni and R. Torlone. Management of multiple models in an extensible database design tool. In *EDBT '96: Proceedings of the 5th International Conference on Extending Database Technology*, pages 79–95, London, UK, 1996. Springer-Verlag.
2. P. A. Bernstein. Applying model management to classical meta data problems. In *CIDR '03: Proceedings of the Conf. on Innovative Database Research*, 2003.
3. R. De Virgilio and R. Torlone. Modeling heterogeneous context information in adaptive web based applications. In *ICWE '06: Proceedings of the 6th international conference on Web engineering*, pages 56–63, New York, NY, USA, 2006. ACM Press.
4. M. J. Emerson, J. Sztipanovits, and T. Bapty. A mof-based metamodeling environment. *Journal of Universal Computer Science*, 10(10):1357–1382, 2004.
5. O. M. Group. XML metadata interchange (XMI) v2.0., 2005.
6. R. Hull and R. King. Semantic database modeling: survey, applications, and research issues. *ACM Comput. Surv.*, 19(3):201–260, 1987.
7. R. Hull and M. Yoshikawa. Ilog: declarative creation and manipulation of object identifiers. In *Proceedings of the sixteenth international conference on Very large databases*, pages 455–468, San Francisco, CA, USA, 1990. Morgan Kaufmann Publishers Inc.
8. R. Kaschek, K.-D. Schewe, B. Thalheim, and L. Zhang. Integrating context in modelling for web information systems. In *WES*, pages 77–88, 2003.
9. E. Medina and J. Trujillo. A standard for representing multidimensional properties: The common warehouse metamodel (cwm). In *ADBIS '02: Proceedings of the 6th East European Conference on Advances in Databases and Information Systems*, pages 232–247, London, UK, 2002. Springer-Verlag.
10. R. Pitrik. An integrated view on the viewing abstraction: Contexts and perspectives in software development, ai, and databases. *Journal of Systems Integration*, 5(1):23–60, 1995.
11. J. Rumbaugh, I. Jacobson, and G. Booch. *The Unified Modeling Language Manual*. Addison Wesley, 1998.
12. T. Strang and C. Linnhoff-Popien. A context modeling survey. In *Proc. of Int. Workshop on Advanced Context Modelling, Reasoning and Management*, 2004.

Autonomic Deadlock Prevention in Self-Managing Databases

Domenico Di Giulio

IBM Software Group – Rome Tivoli Laboratory
d.digiulio@it.ibm.com

Abstract. One of the most complex activities owned by database administrators is the removal of deadlock conditions. Existing database systems provide tools to support DBAs in determining the root causes of deadlocks, but they expose no automatic capability of avoiding that a deadlock condition will not occur again. The removal of deadlocks is left to the DBA, and typically requires a change in the code of an application or procedure, in order to modify the policy it uses to lock resources. A solution is proposed here to provide an RDBMS with the capability of analyzing deadlock conditions and identifying actions to be applied in the future to avoid the same conditions. The solution instructs the RDBMS to identify transactions that have caused deadlocks in the past and apply the right actions in time to avoid the same deadlock conditions.

1 Introduction

Any RDBMS that sets the goal of minimizing the work required to manage the database and reduce (or eliminate) the need for relying on experienced DBAs can hardly leave the problem of deadlock analysis and removal unaddressed. This problem often requires a lot of time to be solved because it can be difficult for a human to analyze the sequence of actions that have led multiple transactions to cause a deadlock. After this time has been spent, however, it is extremely common for the DBA to apply a quite simple solution (such as the addition of a proper table lock in the early stages of a transaction) to avoid that the same type of deadlock will appear in the future, when the same workload will be run again. If considering this, it looks quite strange that existing database systems do not provide any type of *automatic* analysis of deadlock conditions and any mean for identifying and applying actions that avoid deadlocks, instead of expecting someone to *wire* these actions in the application code.

The issue that is usually raised to *give up* any deadlock avoidance strategy is that an algorithm that avoids deadlock conditions, by making sure that the database system is moving always between *safe* states (i.e. states from which no deadlock can be generated), needs to be aware in advance of all the resources that each process may need to acquire until the end of its work. This type of information, which is certainly not known, is the big missing piece that leaves deadlock avoidance into the pages of the books on RDBMS theory.

However, for a database system that serves requests coming from an application, which always executes the same tasks in the same way, saying that the behavior of fu-

ture transactions cannot be predicted is very far from being true. By simply observing the workload that it constantly receives from an application, an RDBMS could *learn* in time what this behavior is, and anticipate conditions that in the past have led to deadlocks with the execution of the proper actions that remove the root causes of known deadlock conditions.

The goal of this paper is to show how the RDBMS can take actions to reduce or eliminate the risk of deadlocks by simply observing the database workload, and learn from the past deadlocks how to lock resources in advance in order to avoid their root causes.

2 The Deadlock Problem

The deadlock problem is a very standard and known one in the world of databases and multi-processing in general.

A **deadlock** is a situation wherein two or more competing processes or execution flows cannot complete their work because they need to acquire resources that are currently held by other processes. In the first steps of this science, E. G. Coffman ([1]) described the four necessary conditions for a deadlock to occur:

1. **Mutual Exclusion**: a resource is either assigned to a process or it is available.
2. **Hold and Wait**: a process that already holds resources may ask for new ones.
3. **No Preemption**: it is not possible to force a process to release the resources it is currently holding.
4. **Circular Wait**: two or more processes create a circular chain where each process waits for a resource held by the next one in the chain.

In a database system, processes execute concurrent transactions, and require access to rows or tables. If the requests involve some kind of exclusive access to requested resources (*mutual exclusion*), concurrent transactions may be blocked, waiting to acquire exclusive locks on rows or tables held by other transactions, and creating a loop (*circular wait*) where no transaction can make any progress.

This can take place because every transaction is allowed to request access to new rows or tables while it already holds a *lock* over other ones (*hold and wait*) and because the RDBMS always waits for transactions to commit their work before allowing other transactions to access the resources they hold (*no preemption*).

2.1 Deadlock Handling Techniques

Deadlock handling solutions can be categorized as follows (even if more complex ones propose a “mixed” approach, trying to mix up the added value of the three basic strategies for deadlock removal):

- **Deadlock detection**: instead of trying to avoid or remove the root causes of deadlocks, database systems typically track the current allocation of resources

to running transactions, detect circular wait conditions and, when a deadlock is found, choose a transaction that is forced to roll back its work, releasing all resources. The theory of deadlock detection actually states that the killed process should be restarted, but this is never done in the real world, because it would require not only to restart the database transaction, but also to restore the initial state of the process that was executing it, to let it run again.

- **Deadlock prevention:** prevention techniques try to avoid deadlocks by removing one of the four necessary conditions just described. These types of solution address the problem by coding applications in such a way that one of the four Coffman conditions is not met. For instance, applications should be coded in such a way that mutual exclusion in accessing resources is never needed, or that every process releases all the resources it holds before requesting new ones, or allowing some kind of resource preemption. But the most known way of preventing deadlocks is to take care of acquiring resources always in the same order, so that the circular wait condition is not possible. If the application is coded in a way that concurrent transactions always access required resources in a predefined order, which is certainly feasible but requires attention at coding time, circular waits are never created and deadlock is prevented.
- **Deadlock avoidance:** these techniques try to remove the occurrence of deadlocks by granting resources to requestors only if the resulting state of the system is *safe*, which means that a deadlock cannot be generated from that state even if all processes request all resources they need at the same time. A well-known implementation of deadlock avoidance techniques is the *banker's algorithm* by E. W. Dijkstra ([2]), which relies on the knowledge of all the resources a process may request during its lifetime, and uses this knowledge to make sure that every request takes the system in a state where all the following ones cannot generate a deadlock cycle. While being interesting in theory, the banker's algorithm is usually considered to be impractical because the source of information it needs is not available in a real environment.

In the RDBMS world, deadlock avoidance is considered almost impossible, since it requires knowledge of unavailable information. If deadlocks are not prevented at the application level, for example by locking tables in a predefined order, they are completely ignored until they are detected. Existing database systems can detect a deadlock and roll back a transaction involved in the wait loop, but cannot *learn* from the history of past deadlocks how to reduce the risk that the same deadlocks will occur in the future.

The solution proposed here relies on a methodology for avoiding deadlocks by *dynamically* applying the principles of deadlock prevention, with particular focus on the removal of circular wait conditions.

3 Autonomic Deadlock Prevention

From the RDBMS perspective, a transaction is just a sequence of SQL statements requiring to *select*, *insert*, *update* or *delete* rows in database tables. To execute this sequence, the RDBMS is required to know which locks must be acquired and when, and to handle issues related to concurrent requests for the same resources from multiple transactions.

Complexity of database locks can be very different for different database systems, but generally a database lock can be defined by the following:

1. **TYPE:** defines the type of resource that is being locked (e.g. locks could be acquired on *rows*, *blocks*, *tables* or *tablespaces*).
2. **TARGET:** identifies the object of the requested type that is being locked (e.g. locks can be held on table *X*, block *Y* or row *Z*).
3. **LEVEL:** identifies the degree of concurrency allowed when the resource is locked (e.g. *shared*, *intent-exclusive*, *exclusive*).

Existing database systems can be very different in the *types* and most of all in the *levels* allowed for the locks they provide, but typically fall within this general model and also describe a table of compatibility of lock requests with locks granted on the same resources. This table shows that in some cases lock requests of a specified level cannot be satisfied if locks of higher levels are held by other transactions on the same resources, or on resources of enclosing types. For instance, a transaction may not be allowed to get an *exclusive* lock on some table's row when the table is locked in *exclusive* mode by some other transaction.

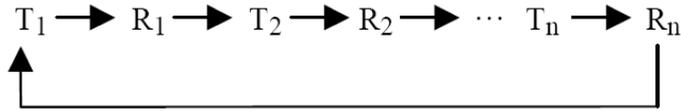
To show the basic idea of autonomic deadlock prevention, however, let's pick a simplified model, where locks can be acquired only on database tables, and only in *exclusive* mode. As described later in this paper, the main idea can be extended to cover the most general model that has been just introduced.

With this simplified model, a database transaction T_i asks in time for a series of table locks:

$$T_i = L_{i1}, L_{i2}, L_{i3}, \dots L_{in}$$

Each lock is related to a specific database table, and represents a request for accessing that table in *exclusive* mode, that is, without allowing other transactions to get access to that table in any way.

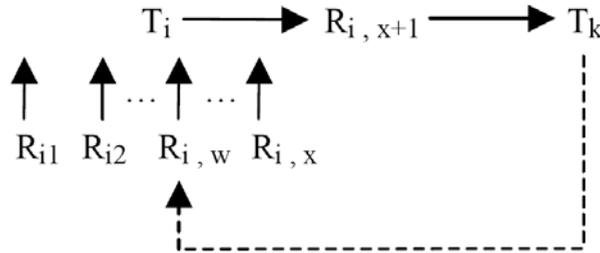
Let's focus now on what happens when a deadlock condition occurs. The deadlock can be represented by a resource allocation graph, describing the set of existing transactions that already hold resources and are waiting to acquire other resources in the same graph (in the following, the word *resource* is used with the meaning of *table*, for the reasons just described):



As it is known, the meaning of arrows in resource allocation graphs is the following: a pending request for resource R_X from transaction T_Y is represented with a directed arrow that goes from T_Y to R_X , while an arrow in the opposite direction means that resource R_X is currently allocated to transaction T_Y .

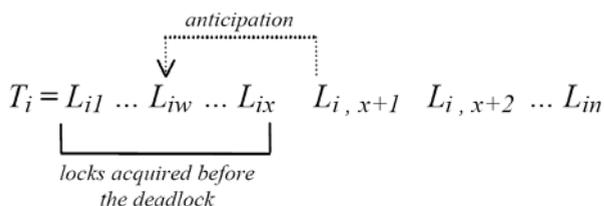
Existing database systems are able to detect loops in the resource allocation graph and select a transaction to roll back, in order to break the loop within which it is involved, but do not analyze the loop to *learn* how it could be avoided in the future.

If the transaction to be rolled back (or any other one involved in the deadlock) is considered, it can be easily found that it currently holds a number of resources, and is waiting to acquire a lock over the next resource in its locking sequence:



As shown in the diagram above, if transaction T_i is involved in the deadlock it must be blocked somewhere in the sequence of table locks it needs to complete its work, waiting for the next one to be acquired. Let x be the blocking position: T_i is already holding a lock over resources $R_{i1} \dots R_{ix}$, and is currently waiting to acquire resource $R_{i,x+1}$. However, this resource is currently held by transaction T_k , which in turn is waiting to acquire, directly or through means of other transactions in the loop, one of the resources held by T_i . Let R_{iw} be this resource: the deadlock could be avoided if, at the time T_i requests R_{iw} , resource $R_{i,x+1}$ is acquired as well, making sure that either both resources are granted or none of them is acquired by the requesting process. This choice is consistent with the recommendations of deadlock prevention techniques: a simple change in the order of locks acquired by a transaction is often the simplest solution to prevent a deadlock.

The main point in this solution is that *the database system can automatically decide to anticipate resource locking*, by analyzing deadlock cycles as described and finding out which lock must be anticipated to the early stages of a transaction, and what is the position in the locking sequence where anticipation must be applied. In the example above, when detecting the deadlock and finding that T_i must be rolled back, the RDBMS can also identify the correcting action to be taken in the early stages of T_i , and keep track of a rule that states: “wait for $R_{i,x+1}$ to be acquired together with R_{iw} as soon as T_i asks for R_{iw} in its locking sequence”.



It is easy to understand that what has been described implicitly relies on the following assumptions:

- From the locking perspective the future expected workload of a database system can be predicted by analyzing the past workload, since tasks are done by applications always in the same way. As it will be described in a while, this assumption is true only if the locking model is simplified in a way that the *target* of fine-grained locks is not considered when comparing locking sequences of different transactions.
- “Known” transactions can be identified in their early stages, and in any case in time for the corresponding recorded actions to be taken in order to avoid the deadlock conditions they have met in their past executions. It is possible to take a probabilistic approach to solve the problem of transaction identification, leading the RDBMS to gradually *learn* in identifying and correcting transactions from its past experience.

In the remainder of this paper, both assumptions are considered, and solutions are proposed to ensure that they are met in a real environment.

3.1 Repeatability of Locking Sequences

Principles of autonomic deadlock prevention can be applied to a real database system only if the database workload can be considered *repeatable*. If a database is used to serve requests from a SW application, there’s no reason why its future workload should be different from the past, since SW applications typically execute the same tasks in the same way. However, for the solution described to be applicable, repeatability of transactions must be guaranteed from a *locking* perspective, which unfortunately is not that obvious if *fine-grained* locks jump into the game.

In other words, transaction repeatability is obvious when the locking model is the simple one we have used so far, which allows only table locks in *exclusive* mode, but is not so evident with real database locking models, where *blocks* or *rows* can be locked as well, since many executions of the same task may acquire locks of the same *type* but on different *targets*. As an example, an application that manages personal employee data for an organization or company may use an “Update Personal Record” task to modify employee records. This task could be used by an employee to search for his record using a personal code, modify personal data and update the record on the database.

Such a task would probably update some kind of EMPLOYEE table without requiring to lock the whole table, but rather acquiring an exclusive lock over a specific row of that table. Therefore multiple executions of this task would probably result in multiple transactions acquiring locks of the same *type* (row) on different *targets* (different employees). In such a case it would be completely useless to anticipate to an early stage of a transaction that updates row *X* an exclusive lock on row *Y* just because the same transaction was involved in a deadlock when updating row *Y*. At the same time, it could be very difficult for the RDBMS even to understand that the two transactions are actually two different instances of the same flow, but working on different data.

This brings to the following rule, which must be applied when implementing automatic deadlock prevention techniques:

“When recording a locking sequence, the database will not keep track of the locked target for lock requests whose type is smaller than table (these will be called fine-grained lock requests)”. (R1)

This rule makes sure that two transactions generated by the same execution flow on different data will show the same locking sequence. Therefore, repeatability of locking sequences is guaranteed.

In addition to this, to make sure that the RDBMS is taking the right choice when anticipating a lock to the early stages of a transaction, the type of the anticipated lock cannot be smaller than *table*. This brings to the following rule:

“When the type of the lock request to be anticipated is smaller than table, the database will actually anticipate a lock of the same level on the table that encloses the requested resource”. (R2)

The combination of rules R1 and R2 allows to consider automatic lock anticipation techniques applicable to database systems with a very general locking model, including *fine-grained* lock requests.

While the application of these rules generate a sub-optimal usage of resources, it allows to remove the root causes of a deadlock by automatically applying the correcting action that in many cases would be manually implemented in the application code by a DBA or developer, leading to a similar sub-optimal usage of resources. The first goal of the solution is to automate the process of analyzing and removing recurring deadlocks, which can be accomplished if optimal usage of resources is not critical. This remains a key assumption for the solution presented to be applicable.

3.2 Identifying and Correcting Transactions

The process of analyzing a deadlock loop and finding which lock request can be anticipated to avoid the same loop in the future is only a part of the problem of automatic deadlock prevention. The RDBMS must be able to identify the next instance of the same execution flow to understand when the correcting action must be applied.

In addition to this, in some cases correcting actions may generate new deadlocks when applied, and if this happens the application of the same actions in the future should be discouraged.

Rather than trying to address the identification problem with complex pattern-matching algorithms (which is certainly a possible choice), a probabilistic approach can be proposed, which relies on past decisions to decide whether or not a partial locking sequence can be recognized as a known and complete one. The same approach is also used to address the problem of deciding whether or not a recorded action to be applied for an identified transaction is likely to be successful or should be discarded, since it would potentially create new deadlock loops.

As described earlier, as soon as a deadlock loop is analyzed, the RDBMS can keep track of the need for anticipating a lock request to a specific “early” stage of one of the transactions involved. While doing this, the RDBMS could also associate to this information a couple of probability assessments:

$P^{(id)}(T, w)$	chance that a transaction whose locking sequence from step 1 to step w matches steps 1... w of transaction T will finally match the whole sequence tracked for T
$P^{(succ)}(T, L, w)$	chance that transaction T will close without being involved in deadlock loops if lock L is anticipated to step w of its locking sequence

Both $P^{(id)}(T, w)$ and $P^{(succ)}(T, L, w)$ are initialized to 50%, without trying to predict their actual values based on any assumption. Then, the database system adjusts these values as soon as it takes decisions to anticipate locks and looks at the results of these decisions.

In order to do that, whenever a running transaction moves to the next step of its locking sequence, the database system tries to match steps 1... w with the ones tracked for other transactions that have been involved in deadlocks in the past. Among all matches found with transactions requiring to anticipate a lock at step w , the transaction T with the maximum value of $P^{(id)}(T, w)$ is considered, and this value is compared to a configurable threshold $H^{(id)}$, to decide whether or not the current transaction should be considered as a new occurrence of T . If $P^{(id)}(T, w) > H^{(id)}$, the database considers the lock L that should be anticipated at the current step w , and the associated value of $P^{(succ)}(T, L, w)$. If this value is higher than a configurable threshold $H^{(succ)}$, the RDBMS decides to apply the correcting action, acquiring lock L at step w , and keeping track of this.

Then, the database must keep monitoring the whole execution of the transaction and take further actions as follows, also updating values of $P^{(id)}$ and $P^{(succ)}$:

1. If the affected transaction is finally found to be an occurrence of transaction T , the current value of $P^{(id)}(T, w)$ must be raised appropriately. Then, three different cases may be found:
 - 1a). The affected transaction ends regularly. In this situation, the value of $P^{(succ)}(T, L, w)$ must be raised to confirm that anticipating lock L at step w is a good choice to avoid deadlocks for transaction T .

The update of $P^{(succ)}(T, L, w)$ must reflect how relevant the current execution was; that is, $P^{(succ)}(T, L, w)$ must be raised more if the degree of concurrency, and therefore the chance of deadlock occurrence, was higher.

- 1b). The affected transaction gets involved in a deadlock before step x , for which anticipation of lock L was previously determined. In this case, if the resource on which lock L has been acquired is within the deadlock loop, anticipation of L may be the cause of the deadlock and must be discouraged: the current value of $P^{(succ)}(T, L, w)$ must be decreased, and the anticipated lock must be immediately released. If on the other hand the resource on which L has been acquired is out of the deadlock loop, $P^{(succ)}(T, L, w)$ can be left unchanged.
 - 1c). The affected transaction gets involved in a deadlock after step x . If this happens, the anticipation of L cannot be the cause of the deadlock, and the action taken is the same as for case 1a). The new deadlock can be analyzed to track the need for anticipating a new lock at the current step in the locking sequence of transaction T .
2. If the affected transaction is finally found to be different from T , the current value of $P^{(id)}(T, w)$ must be decreased, and if a match is found with another transaction T' that has generated a deadlock in a past execution, the value of $P^{(id)}(T', w)$ must be raised instead of $P^{(id)}(T, w)$. In addition to this, the anticipated lock must be immediately released to avoid unnecessary resource usage.

A simple way of updating the current values of $P^{(id)}$ and $P^{(succ)}$ is to calculate an exponential moving average, using a configurable smoothing factor. For instance, the n -th update of the chance $P^{(X)}$ associated to event X can be calculated from the $(n-1)$ -th update of the same chance using the following expression:

$$P_n^{(X)} = \alpha P_{n-1}^{(X)} + (1 - \alpha) E_n^{(X)} \quad \alpha \in [0, 1]$$

$$E_n^{(X)} = \begin{cases} 1 & \text{if event } X \text{ has occurred} \\ 0 & \text{if event } X \text{ did not occur} \end{cases}$$

This very simple model can be used to update both $P^{(id)}$ and $P^{(succ)}$, with a difference in how the smoothing factor α must be calculated. Actually, while for $P^{(id)}$ this factor can be left constant, for $P^{(succ)}$ it must be lower for higher degrees of concurrency, so that the new value of $P^{(succ)}$ is raised more if more transactions were running concurrently with T without incurring in a deadlock. To achieve this goal, α can be calculated with the following expression, where c is the number of transactions that were running concurrently with T :

$$\alpha = \beta / (1 + c) \quad \beta \in [0, 1]$$

If $c=0$, the smoothing factor α has exactly the same value of a configurable factor β , which should be very close to 1 (but lower), while if c increases, the value of α is progressively lower, so that the new value of $P^{(succ)}$ is raised more with respect to the previous one. This choice makes sure that only significant cases of deadlock prevention will actually confirm that the lock anticipation is to be considered a good choice. On the other hand, if $\beta < 1$ (but still $\beta \approx 1$), the anticipation of a lock is gradually discouraged if there's no concurrency issue that confirms it is needed through positive values of c , which is meaningful if considering that in databases with poor or no concurrency there should be no need to apply lock anticipation.

Of course, both values of α and β should be configurable by the DBA (good defaults could be 80% and 99% respectively), as well as both values of thresholds $H^{(id)}$ and $H^{(succ)}$ (a careful approach suggests not null values for both of them). Any choice automatically applied by the RDBMS should be evidently logged for problem determination, and it should be possible to disable the whole solution at any time, switching from a “*learn and react*” mode to a “*learn and notify*” approach, where the RDBMS just logs (or informs the DBA in any other way) of the possible lock anticipation actions that could be applied to reduce the risk of deadlocks, without applying them.

3.3 Performance and Fine-Grained Locks

While the solution described may appear heavy at a first glance, because it forces the database to look for transaction matching and correcting actions at any time a transaction asks for a new lock, it can be easily shown that the additional burden is actually acceptable. This is because all of the locking and transaction information required by the database system in order to implement lock anticipation are actually already held in memory, since they support the synchronization of transactions, which is a responsibility of the RDBMS.

The locking sequences of transactions to be compared with the running ones (because they generated a deadlock in the past) must be also kept in memory, so that identification and correction of transactions can take place without slowing down the system. Values of $P^{(id)}$ and $P^{(succ)}$ must be stored persistently on the file system, but this can happen at regular times rather than whenever they are changed. Also, all the information required to know which locks can be anticipated in order to correct database transactions is stored on disk only at the time a deadlock is detected.

On the other hand, it has to be noticed that acquiring anticipated locks on database tables means decreasing the overall degree of concurrency and may not be acceptable for systems with strong performance requirements. For the same reason, the solution described should be applied only when deadlock prevention is critical and optimal access to database resources is not a key requirement.

Alternatively, techniques for extending the solution described to cover *fine-grained* locking should be considered, so that the negative impact of table locks could be removed. As an example, when the database system detects a deadlock that involves rows locked in exclusive mode from different tables, it should investigate the existing relationship between these rows, so that the deadlock could be prevented by anticipating locks on the proper rows rather than locking an entire table.

Transactions are supposed to lock rows from different tables when they are linked together by some kind of relationship. However, this relationship is defined by the meaning of the operation executed, and could be difficult to identify. For instance, a transaction might need to update rows in different tables not only when these rows are linked together by referential integrity, but when they include information related to the same domain or the same information at different aggregation levels, or even when one of the tables involved is some kind of log or control table that needs to be properly updated for every database operation.

With respect to this problem, the database system could implement one of two different approaches, which could be summarized as follows:

1. Try to discover the relationship that exists between rows extracted from different tables when they are locked by the same transaction.
2. Keep track of the rows involved in a deadlock, and anticipate locking of these rows only when the transaction appears to be *exactly* the same.

By discovering the relationship between rows locked by the same transaction, it would be possible to anticipate appropriate locks when it is running again on different data. For instance, the database system could navigate foreign keys to find the path that links involved rows (probably passing through intermediate tables), and then it could decide to anticipate locking of the row (or the subset of rows) that is reached by navigating the same path when the transaction is running again, but starting from different data. This way, the database would be actually discovering how a transaction could be considered repeatable *individually* for each transaction.

Notice that if no relationship is discovered through foreign keys, it could be possible to investigate other types of relationships, for instance by looking for matching values on columns that are not linked by referential integrity constraints, or applying other techniques from the data mining field. But even this could fail when the relationship to be detected is complex, or when dealing with a transaction that performs multiple operations that are not related together in any way.

In these cases, the second approach may be considered. With this choice, the database system always keeps track of the *target* of involved locks, and applies anticipation techniques only when detecting transactions that work exactly on the same *target*. This makes sure that anticipated locks are meaningful in the context of affected transactions, but increases the amount of information collected by the database, and limits the application of the solution described to cases when the system is exactly repeating the same steps that in a past occasion have led to a deadlock.

4 Conclusions and Future Work

In this paper, a methodology has been proposed to automatically prevent deadlocks in a database system, which starts from analyzing the root causes of deadlock conditions, and determines the correcting actions to be applied to the involved transactions the next time they are run again, to avoid these conditions.

The solution relies on the assumption that the workload expected for a database system can be determined from its past history, which is acceptable for databases serving requests from SW applications.

With this assumption, it has been shown how a simple solution can be proposed to *dynamically* apply the same type of actions that a DBA (or developer) usually needs to *statically* define into the code of a stored procedure or application. These actions are often related to the simple anticipation of a lock in the early stages of a transactional piece of code that was found to be deadlock prone in its past executions. Locks can be anticipated at the table level, to make sure that transactions can be considered *repeatable*, but extensions of this basic approach can be considered to replace table locks with *fine-grained* ones, improving performance.

The next step is trying to get from this very general methodology a proposed algorithm, whose value can be tested through a SW prototype, so that its strengths and weaknesses can show up and be addressed. The result could be a certified solution to increase the self-managing features of an existing database system.

References

1. E.G. Coffman, M.J. Elphick and A. Shoshani. System Deadlocks. *ACM Computing Surveys*, 1971.
2. E.W. Dijkstra. The Mathematics Behind the Banker's Algorithm. *Selected Writings on Computing*, 1982.
3. E.W. Dijkstra. Cooperating Sequential Processes. *Programming Languages*, 1965.
4. S.S. Isloor and T.A. Marsland. The Deadlock Problem: An Overview. *IEEE Computer*, 1980.
5. S. Bensalem and K. Havelund. Scalable Dynamic Deadlock Analysis of Multi-threaded Programs. *Haifa Verification Conference*, 2005.
6. R. Agrawal, M. J. Carey, and L.W. McVoy. The Performance of Alternative Strategies for Dealing with Deadlocks in Database Management Systems. *IEEE Transactions on Software Engineering*, 1987.
7. P. A. Bernstein and E. Newcomer. Principles of Transaction Processing. *Morgan Kaufmann*, 1997.
8. H. F. Korth. Locking primitives in a database system. *Journal of the ACM*, 1983.

Approximate Query Answering and Ranking for Semantic Knowledge Bases

Nicola Fanizzi, Claudia d'Amato, Floriana Esposito

Dipartimento di Informatica, Università degli Studi di Bari
Campus Universitario, Via Orabona, 4 – 70125 Bari, Italy
{fanizzi|claudia.damato|esposito}@di.uniba.it

Abstract. We present a method for retrieval in knowledge bases expressed in Description Logics, founded in the *instance-based learning* and the *disjunctive version space* approach. The method can be employed to answer to class-membership queries, even though the answers are not logically entailed by the knowledge base, e.g. there are some inconsistent assertions due to heterogeneous sources. In addition, a ranking for the inductively produced answers can be also derived by exploiting the voting procedure embedded in it. The method has been implemented and tested in an experimentation, where we show that it is sound and effective.

1 Introduction

In the perspective of the Semantic Web [3], knowledge bases require specific forms of representation and reasoning which allow for processing data semantics through logic-based inference. Description Logics (DLs) [1] have become the theoretical foundation for representing knowledge in such a context. Moreover they are endowed by reasoning services. Anyway, purely logic methods may fail or fall short when data sources are distributed and potentially incoherent. This has given rise to alternative logic methods, such as *non-monotonic*, *paraconsistent* [10], *approximate* reasoning [11], *case-based reasoning* [7] and inductive forms such as inductive *generalization* [4, 9].

Instance-based methods are known to be both very efficient and fault-tolerant compared to the classic logic-based methods, being noise always a danger in contexts where knowledge is distributed and acquired from heterogeneous sources. Two different kinds of noise can be distinguished: 1) noise that may be introduced by inconsistency in the knowledge base; 2) noise due to incorrect knowledge that does not strictly cause inconsistency, nevertheless it may yield wrong conclusions with respect to the intended meaning of the concepts in considered domain.

Answering to a query, namely finding the extension of a query concept, can be cast as a problem of establishing the class membership of the semantically annotated individuals in a KB. Based on our past experience on distance-based inductive methods [5, 6], an instance-based framework for DLs was devised to

derive (by analogy) inductive conclusions from the knowledge base, possibly also some which were not previously logically deducible. For the best of our knowledge no other works exist that solve the query answering problem by the use of an inductive approach.

We elaborated a relational form of the *Nearest Neighbor* (NN) approach [13]. The baseline idea for solving the query answering task is that similar individuals, by analogy, should likely belong to similar concepts.

These ideas have been further extended by considering another form of inductive inference: the *disjunctive version space* approach [15], adapted to a DL framework. In this setting, the neighborhood of an individual with respect to a target concept is determined on the grounds of its similarity to other training individuals in the knowledge base which are known to belong to that concept. Instead of using a similarity measure, like in the basic NN approach, the notion of neighborhood is based on class-membership queries performed on a training set of individuals. Specifically, an individual is said to belong to the neighborhood of a positive example, when it belongs to the conjunction of the concepts that differentiate that instance from each negative example. In turn, each such a concept can be regarded as a disjunction of features that separate a positive from a negative example. Thus belonging to the neighborhood of positive instances of a target concept gives a criterion to decide on the membership of an individual. The procedure is not crisp, since a number of mistakes can be tolerated, blaming them to the noise in the data. Since the decision is made on the grounds of a majority vote, the outcomes of this vote may be exploited to estimate the strength of the decision, which gives also a way to rank the likelihood of the class-membership for individuals in the answer set.

The remainder of the paper is organized as follows. We briefly survey the basics of the DLs representation and related inference service in Sect. 2. Then, the NN method adapted to the DLs standards is introduced in Sect. 3 and further refined in Sect. 4, integrating a disjunctive version space approach. The method has been implemented so that some preliminary experimental results with real ontologies can be presented in Sect. 5. Finally, we conclude proposing further developments of the presented methods.

2 Representation and Inference Services

As reference representation language, \mathcal{ALC} logic [14] has been chosen, since it is considered a good compromise between expressiveness and computational effort (required by the inference services). This logic adopts constructors supported by the standard Web ontology languages (see [1] for a thorough reference). Actually, the methods presented in the next sections may be made less language dependent through suitable approximations. In this section the basics of \mathcal{ALC} logic and inference are briefly recalled.

In DLs, concept descriptions are defined in terms of a set N_C of *primitive concept* names and a set N_R of *primitive roles*. Complex descriptions are built using primitive concepts and roles and the constructors in Table 1. The semantics

Table 1. \mathcal{ALC} constructors and their meaning.

Name	Syntax	Semantics
<i>top</i>	\top	$\Delta^{\mathcal{I}}$
<i>bottom</i>	\perp	\emptyset
<i>negation</i>	$\neg C$	$\Delta^{\mathcal{I}} \setminus C^{\mathcal{I}}$
<i>conjunction</i>	$C_1 \sqcap C_2$	$C_1^{\mathcal{I}} \cap C_2^{\mathcal{I}}$
<i>disjunction</i>	$C_1 \sqcup C_2$	$C_1^{\mathcal{I}} \cup C_2^{\mathcal{I}}$
<i>exist. restr.</i>	$\exists R.C$	$\{x \in \Delta^{\mathcal{I}} \mid \exists y \in \Delta^{\mathcal{I}}((x, y) \in R^{\mathcal{I}} \wedge y \in C^{\mathcal{I}})\}$
<i>value restr.</i>	$\forall R.C$	$\{x \in \Delta^{\mathcal{I}} \mid \forall y \in \Delta^{\mathcal{I}}((x, y) \in R^{\mathcal{I}} \rightarrow y \in C^{\mathcal{I}})\}$

of the concept descriptions is defined by an *interpretation* $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$, where $\Delta^{\mathcal{I}}$ is a non-empty set, the *domain* of the interpretation, and the *interpretation function* $\cdot^{\mathcal{I}}$ maps each $A \in N_C$ to a set $A^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}}$ and each $R \in N_R$ to $R^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}$. Complex description are interpreted as shown in Table 1.

A *knowledge base* $\mathcal{K} = \langle \mathcal{T}, \mathcal{A} \rangle$ contains a *TBox* \mathcal{T} and an *ABox* \mathcal{A} . \mathcal{T} is a set of concept definitions¹ $C \equiv D$, meaning $C^{\mathcal{I}} = D^{\mathcal{I}}$, where C is atomic (the concept name) and D is an arbitrarily complex description defined as above. \mathcal{A} contains assertions on the world state, e.g. $C(a)$ and $R(a, b)$, meaning that $a^{\mathcal{I}} \in C^{\mathcal{I}}$ and $(a^{\mathcal{I}}, b^{\mathcal{I}}) \in R^{\mathcal{I}}$. Moreover, normally the *unique names assumption* is made on the ABox individuals. These are denoted with $\text{Ind}(\mathcal{A})$.

In this context the most common inference is the semantic notion of *subsumption* between concepts:

Definition 2.1. *Given two concept descriptions C and D , D subsumes C , denoted by $C \sqsubseteq D$, iff for every interpretation \mathcal{I} it holds that $C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$. When $C \sqsubseteq D$ and $D \sqsubseteq C$, they are equivalent, denoted with $C \equiv D$.*

Example 2.1. An instance of concept definition in the proposed language is:

$$\text{Father} \equiv \text{Male} \sqcap \exists \text{hasChild}.\text{Person}$$

which corresponds to the sentence: "a father is a male (person) that has some persons as his children".

The following are instances of simple assertions:

$$\text{Male}(\text{Tom}), \text{Male}(\text{Bill}), \text{hasChild}(\text{Tom}, \text{Bill}).$$

Supposing that $\text{Male} \sqsubseteq \text{Person}$ is known (in the T-Box), one can deduce that: $\text{Person}(\text{Tom})$, $\text{Person}(\text{Bill})$ and then $\text{Father}(\text{Tom})$.

Given these primitive concepts and roles, it is possible to define many other related concepts:

$$\text{Parent} \equiv \text{Person} \sqcap \exists \text{hasChild}.\text{Person}$$

and

$$\text{FatherWithoutSons} \equiv \text{Male} \sqcap \exists \text{hasChild}.\text{Person} \sqcap \forall \text{hasChild}.\neg \text{Male}$$

¹ The cases of general axioms or cyclic definitions will not considered here.

It is easy to see that the following relationships hold: $\text{Parent} \sqsupseteq \text{Father}$ and $\text{Father} \sqsupseteq \text{FatherWithoutSons}$. \square

Another important inference is *instance checking*, that is deciding whether an individual is an instance of a concept [8, 1].

Conversely, it may be necessary to solve the *realization problem* that requires finding the concepts which an individual belongs to, especially the most specific one:

Definition 2.2. *Given an ABox \mathcal{A} and an individual a , the most specific concept of a w.r.t. \mathcal{A} is the concept C , denoted $\text{MSC}_{\mathcal{A}}(a)$, such that $\mathcal{A} \models C(a)$ and for any other concept D such that $\mathcal{A} \models D(a)$, it holds that $C \sqsubseteq D$.*

Unfortunately, for many non-trivial DL languages, such as \mathcal{ALC} , the exact MSC may not be always expressed with a finite description [1] interpreted with the descriptive semantics presented earlier, yet it may be approximated [4, 2], which is satisfactory for inductive approaches. Generally an approximation of the MSC is considered up to a certain depth p . The maximum depth p has been shown to correspond to the depth of the considered A-Box, as defined in [12].

3 The Nearest Neighbor Procedure Applied to DLs

In this section, the basics of the Nearest Neighbor approach [13] are recalled, showing how to exploit a classification procedure for inductive reasoning and retrieval.

Nearest Neighbor is a lazy-learning approach, so called as the learning phase is reduced to simply memorizing training instances of the target concepts that are pre-classified by an expert. All computational efforts are performed during the classification phase, where a notion of similarity for the instance space is employed to classify a new query instance.

Such an approach can be used in the SW context to classify a new instance with respect concepts in an ontology by analogy with its neighbors. Specifically, given an ontology, a classification method can be employed for assigning an individual with the concepts it is likely to belong to. These individuals are supposed to be partially described by assertions in the ABox, thus classification may induce new assertions by analogy, which cannot not be inferred by deduction.

The classical Nearest Neighbor approach can be explained as in the following. Let x_q be the instance that must be classified. Using a similarity measure (or any other distance function), the set of the k pre-classified instances nearest with respect to x_q is selected. The objective is to learn an estimate of a hypothesis function for the target concept membership $h : \text{TI} \mapsto V$ from a space of training instances TI to a set of values $V = \{v_1, \dots, v_s\}$ standing for the classes to be assigned.

In its simplest setting, the algorithm approximates h for x_q on the ground of the value that h assumes for the training instances in the neighborhood of x_q , i.e. the k closest instances to the new instance in terms of a dissimilarity measure.

Precisely, this instance is assigned a class according to the value which is *voted* by the majority of instances in the neighborhood. This setting takes into account similarity only when selecting the instances to be included in a neighborhood.

A more general setting is based on weighting the vote according to the distance of the query instance from the training instances:

$$\hat{h}(x_q) := \operatorname{argmax}_{v \in V} \sum_{i=1}^k w_i \delta(v, h(x_i)) \quad (1)$$

where \hat{h} is the estimated hypothesis function, δ is the *Kronecker symbol*, namely a function that returns 1 in case of matching arguments and 0 otherwise, and, given a distance measure d , the weights w_i can be defined as $w_i = 1/d(x_i, x_q)$ or $w_i = 1/d(x_i, x_q)^2$. In the case of the \mathcal{ALC} DL, the similarity measures defined in [6] could be employed.

Note that the hypothesis function \hat{h} is defined only extensionally, therefore the k -NN method does not return an intensional classification model (a function or a concept definition), it merely gives an answer for new query instances to be classified, using the procedure mentioned above.

It should be observed that a strong assumption made in this setting is that it can be employed to assign the query instance to a class from a set of values which can be regarded as a set of pairwise disjoint concepts. This is a simplifying assumption that cannot be always valid. In our setting, indeed, an individual could be an instance of more than one concept.

Let us consider a value set $V = \{C_1, \dots, C_s\}$, of possibly overlapping concepts C_j ($1 \leq j \leq s$) that may be assigned to a query instance x_q . If the classes were disjoint as in the standard setting, the decision procedure defining the hypothesis function is the same as in Eq. (1), with the query instance assigned the *single* class of the majority of instances in the neighborhood. In the general case, when the pairwise disjointness of the concepts cannot be assumed, another classification procedure has to be adopted. A possible solution can be to decompose the multi-class classification problem into smaller binary classification problems (one per target concept).

The multi-class classification problem is not the only issue that has to be solved in order to apply the Nearest Neighbor approach to the SW context. Another problem is also related to the Closed World Assumption (CWA) usually made in the knowledge discovery context, since, on the contrary, in the SW context, the Open World Assumption (OWA) is usually made.

To deal with the OWA, the absence of information on whether a certain training instance x belongs to the extension of a concept C_j should not be interpreted negatively, it should rather count as neutral information. Thus, a ternary value set has to be adopted as codomain for the h_j functions, namely $V = \{-1, 0, +1\}$, where the values denote, respectively, membership, non-membership and absence of information²:

² In the following, \vdash is a proof-theoretic inference operator provided by a DL reasoner.

$$h_j(x) = \begin{cases} +1 & \mathcal{K} \vdash C_j(x) \\ -1 & \mathcal{K} \vdash \neg C_j(x) \\ 0 & \textit{otherwise} \end{cases}$$

The checks could have been pre-computed for the KB, therefore the overall complexity of the procedure depends on the number $k \ll |\text{Ind}(\mathcal{A})|$, that is the number of times the distance measure is computed in order to assign class/es to the query instance.

It is important to note that, since the presented procedure is based on a majority vote of the individuals in the neighborhood, it is less error-prone in case of noise in the data (i.e. incorrect assertions in the ABox), therefore it may be able to give an answer, requiring only the correctness of the training instances classification, while in such a situation, a purely logic-based approach can give any answer.

4 Building Intensional Local Models

Another related analogy-based method for performing retrieval and query answering with respect an ontological knowledge base can be derived from the notion of *Disjunctive Version Space* [15]. Differently from the NN approach that is based on distances, in the following method, the population of an individual's neighborhood is performed inducing definitions for the query concept on the grounds of its examples and counterexamples available in the knowledge base.

4.1 The Disjunctive Version Space Procedure

Given a query concept Q , for each training instance $x \in \text{TI}$ such that $Q(x)$ holds (positive example for Q), a hypothesis H_Q^x may be generated (*on-the-fly*) by considering the subset of counterexamples for Q , denoted $\bar{E}_Q = \{\bar{x} \in \text{TI} \mid \mathcal{K} \vdash \neg Q(\bar{x})\} \subseteq \text{TI}$ and finding a maximally discriminating description $D(x, \bar{x}) \in \mathcal{L}$ for each $\bar{x} \in \bar{E}_Q$. Specifically, for maximally discriminating description of an individuals \bar{x} that is the counterexample of an individual x we mean the most specific concept (w.r.t. the subsumption relation) to which \bar{x} belongs to and that does not cover x , namely x is not instance of such a concept description. Thus, the hypothesis H_Q^x may be regarded as the conjunction of such $D(x, \bar{x})$'s, to be induced varying the counterexamples \bar{x} in \bar{E}_Q :

$$H_Q^x = \prod_{\bar{x} \in \bar{E}_Q} D(x, \bar{x})$$

In order to produce a(n approximation of) description $D(x, \bar{x})$, one possibility is considering the difference

$$D(x, \bar{x}) := \text{MSC}^P(x) - \text{MSC}^P(\bar{x})$$

where p is a fixed depth that may depend on the ABox depth (see Sect. 2) and the symbol $-$ denotes Teege’s *difference operator* for DL descriptions [16]. In the case of \mathcal{ALC} , we have:

$$D(x, \bar{x}) := D_x \sqcup \neg D_{\bar{x}}$$

where $D_x = \text{MSC}^p(x)$ and $D_{\bar{x}} = \text{MSC}^p(\bar{x})$.

Now, for each training individual x , the individual under classification x_q will belong to x ’s neighborhood w.r.t. Q iff it belongs to the related hypothesis H_Q^x . The *neighbor instance set* of x_q w.r.t. Q is defined as follows

$$N_Q(x_q) := \{x \in \text{TI} \mid \mathcal{K} \vdash H_Q^x(x_q)\}$$

The approximate query answering procedure can be defined as a majority vote for the values $V = \{-1, 0, +1\}$:

$$\hat{h}_Q(x_q) := \operatorname{argmax}_{v \in V} \sum_{x \in N_Q(x_q)} \delta(v, H_Q^x)$$

Yet, in this case the procedure may be biased by the different numbers of training instances in $N_Q(x_q)$ voting for the other values. Hence, we rather consider the proportions of votes over the total number of training individuals classified with the three values of V :

$$\hat{h}_Q(x_q) := \operatorname{argmax}_{v \in V} \sum_{x \in N_Q(x_q)} w_Q^v \cdot \delta(v, H_Q^x) \quad (2)$$

where the weighting factor $w_Q^v = \#(v, N_Q(x_q)) / \#(v, \text{TI})$ denotes the count of neighbor instances w.r.t. x_q voting for value v for the query concept Q over the total number of training individuals belonging to the same class.

4.2 Discussion

As suggested in [15], the procedure can be parameterized on precision and recall. Indeed it may be made more noise-tolerant by admitting an amount of consistency errors (say ε) in deciding whether a training instance belongs to the neighborhood: there may be up to $\varepsilon \cdot |\bar{E}_Q|$ cases, i.e. a number of counterexamples $\bar{x} \in \bar{E}_Q$, for which $x_q \notin D(x, \bar{x})$ and yet the membership to a neighborhood will be assumed as acceptable ($x \in N_Q(x_q)$).

Besides, the method can be tuned also w.r.t. the completeness, by adjusting the specificity of $D(\cdot, \cdot)$ according to a number of features to be considered (say M) as a separation between positive and negative instances.

Better and language-independent definitions of $D(\cdot, \cdot)$ can be considered, that may be based only on the available assertions, as the algorithm only requires to know whether a new individual belongs to the neighborhood or not and this can be specified also with no involvement of the concept level. Namely, in order to extend the applicability to more expressive languages than \mathcal{ALC} , an alternate way for building the discriminating definitions $D(\cdot, \cdot)$ has to be defined.

The method adopted here is suitable for logics endowed with a notion of difference and a further approximation had to be made on the construction of the MSC's. Nevertheless, the algorithm is not extremely language-dependent: any other method that can induce concept descriptions which are able to explain a positive instance and rule out a single negative one would be acceptable.

The presented method mainly depends from the complexity of the *instance checking* operator for the chosen DL. It is used both for determining the counterexamples of a given query concept Q and for computing the (approximation of) MCS of an individuals.

4.3 Ranking

This instance-checking procedure provides an inductive (approximate) answer to whether individual x_q belongs to the concept denoted by value v , given the nearest training individuals in $N_Q(x_q)$. In order to rank the likelihood of the answer, the *argmax* argument in Eq. (2) is maximized. We use these values to estimate the likelihood of each value. Then these values should be normalized:

$$\hat{P}(x_q|Q) = \frac{\hat{h}_Q(x_q)}{\sum_{v' \in V} \sum_{x \in N_Q(x_q)} w_Q^{v'} \cdot \delta(v', H_Q^x)} \quad (3)$$

Then the ranking measure of the answer x_q w.r.t. query Q is:

$$\text{RankMeasure}(x_q, Q) = \hat{P}(x_q|Q)$$

5 Experiments

We present the outcomes of experiments carried out for testing the feasibility of the method illustrated in the previous section. Its implementation was tested on answering queries w.r.t. four ontologies drawn from the Protégé library³, endowed with different numbers of individuals, namely: the FSM, SURFACE-WATER-MODEL, SCIENCE, and NEWTESTAMENTNAMES. Some are expressed in larger DLs than \mathcal{ALC} . This affected the construction of the MSC's approximations, which turned out to be more general than those that could be produced in the original DLs.

FSM is an $\mathcal{SF}(D)$ KB describing finite state machines. It is made up of 20 concepts, 10 object properties, 7 datatype properties. SURFACE-WATER-MODEL is an $\mathcal{ALC}(D)$ ontology describing water quality models. It is based on the *Surface-water Models Information Clearinghouse* of the US Geological Survey. It deals with numerical models for surface water flow and water quality simulation. These models are classified according to their availability, domain, dimensions, and characteristic types. It is made up of 19 concepts, 9 object properties. SCIENCE is an $\mathcal{ALCIF}(D)$ KB and describes scientific facts. It is made up of 74 concepts, 70 object properties. The NEWTESTAMENTNAMES is an $\mathcal{SHIN}(D)$ KB which

³ <http://protege.stanford.edu/download/ontologies.html>

describes facts related to the New Testament (*Semantic Bible Project*). It is made up of 47 concepts, 27 object properties.

The classification method presented in the previous section was applied to each test ontology, by generating 15 random queries based on the concepts and roles therein; each query is a complex concept made up of a variable random number (from 2 up to 11) of (primitive and defined) concepts found in the knowledge base.

A naïve retrieval procedure required, for each test query, every individual is considered to determine if it belongs to the answerer set (+1) or not (-1), or it is neutral 0, i.e. unknown answer) w.r.t. the ontology. Specifically, for each training individual an MSC approximation was pre-computed and assigned to the set of examples or counterexamples w.r.t. the query concept. Each test individual is then classified applying the method presented in the previous section. For the smaller knowledge bases leave-one-out cross validation procedure was used, while for the larger ones a 10-fold cross validation was performed. We intended to assess whether our method is able to retrieve instances correctly, i.e. its performance was compared to the relevance determined by an expert, whose role was made by a reasoner⁴. Additionally, it should also be able to induce by analogy new (previously unknown) class-membership assertions that cannot be logically inferred.

Particularly, for each ontology and for each concept, four rates have been computed: *match rate*, *omission error rate*, *commission error rate*, *induction rate*. The match rate is the proportion of instances retrieved exactly as a reasoner would do. The omission error is related to completeness. It measures the amount of relevant individuals w.r.t. a certain query (i.e. the answer is ± 1) that were not retrieved (answer 0). The commission error is related to soundness. It measures the amount of individuals whose relevance was mismatched, i.e. they were retrieved when they belonged to the negation of the query concept or vice-versa. The induction rate measures the amount of individuals found as relevant (answer ± 1) even though the expert cannot give an answer (i.e. the reasoner returns **unknown**). Thus, commission error may be more harmful than omission error. A high induction rate means that the procedure was actually able to suggest new assertions that are likely to be valid and can be validated by a knowledge engineer.

Tab. 2 reports the experimental results. Per each ontology, we report the average rates for the measures discussed above and also the interval of values assumed for the 15 random queries during the cross validation experiments.

Primarily, by looking at table, it is important to note that, for every ontology, the commission error was null on average and the variance is also quite low. This means that the classifier has never made critical mistakes because no individual has been deemed as an instance of a concept while really it is an instance of a disjoint class. Also the omission error rate is almost null. The highest value was observed on the FSM ontology, likely caused by the very few individuals in it.

⁴ We employed PELLET: <http://www.mindswap.org>

Table 2. Results of the experiments.

ONTOLOGY			<i>match rate</i>	<i>induction rate</i>	<i>omission err. rate</i>	<i>commission err. rate</i>
1ST EXPERIMENT	FSM	<i>avg.</i>	0.92	0.00	0.08	0.00
		<i>range</i>	0.84 - 1.00	0.00 - 0.00	0.00 - 0.16	0.00 - 0.00
	SURFACE-WATER-MODEL	<i>avg.</i>	0.92	0.07	0.01	0.00
		<i>range</i>	0.57 - 1.00	0.00 - 0.43	0.00 - 0.03	0.00 - 0.00
SCIENCE	<i>avg.</i>	0.94	0.06	0.00	0.00	
	<i>range</i>	0.04 - 1.00	0.00 - 0.96	0.00 - 0.00	0.00 - 0.00	
NEWTTESTAMENTNAMES	<i>avg.</i>	0.98	0.00	0.02	0.00	
	<i>range</i>	0.78 - 1.00	0.00 - 0.00	0.00 - 0.22	0.00 - 0.00	
2ND EXPERIMENT	FSM	<i>avg.</i>	0.81	0.00	0.19	0.00
		<i>range</i>	0.30 - 1.00	0.00 - 0.00	0.00 - 0.68	0.00 - 0.03
	SURFACE-WATER-MODEL	<i>avg.</i>	0.54	0.46	0.00	0.00
		<i>range</i>	0.02 - 1.00	0.00 - 0.99	0.00 - 0.01	0.00 - 0.00
SCIENCE	<i>avg.</i>	0.98	0.02	0.00	0.00	
	<i>range</i>	0.71 - 1.00	0.00 - 0.29	0.00 - 0.00	0.00 - 0.00	
NEWTTESTAMENTNAMES	<i>avg.</i>	0.45	0.55	0.00	0.00	
	<i>range</i>	0.01 - 1.00	0.00 - 0.99	0.00 - 0.00	0.00 - 0.00	

The performance is comparable to the reasoner, as the high match rate values show. Yet this yields low induction rates.

As such, the method appears to be sound. As regards the completeness, we observed during the first experiment that the amount of individuals that vote for the **unknown** class w.r.t. the query was generally high for these ontologies, thus biasing the final decision. Although this may be satisfactory compared to the reasoner's performance, the final goal is to overcome the inherent incompleteness due to the OWA and try to induce the real classification of an individual (w.r.t. the intended meaning of the domain modeled by an ontology). Therefore, we tweaked the procedure, by decreasing the impact of the individuals classified as unknown during the voting phase. Specifically, it is modified by answering **unknown** only when the number of neighboring positives and negatives is balanced; in the rest of the cases, the new procedure gives an answer to a binary classification problem depending on the their majority.

Again, the method proved sound (null commission error rate). As regards completeness, we observed in two cases a shift from the match rate towards the induction rate: i.e. the system actually suggested a classification, even in presence of a high rate of individuals classified as **unknown**. Indeed, for the SURFACE-WATER-MODEL and the NEWTTESTAMENTNAMES ontologies, the incompleteness is caused by the lack of information about concept disjointness, thus yielding no counterexamples which were needed to assess the membership to the target query. For the other cases, the outcomes are almost similar to those observed in the previous experiment. Namely, or the FSM ontology this should be caused by the number of disjointness axioms (46) which greatly helps the procedure.

Moreover, most of the its individuals are instances of a single concept. This latter situation applies also the SCIENCE ontology.

Concluding, we have observed that the proposed method is often able to induce new assertions in addition those that were already logically derivable from the knowledge base. Particularly, an increase in prediction accuracy was observed when the instances are homogeneously spread and information about concept disjointness is available. Besides, the method confirmed its tolerance to noise as a very low commission error was observed. In order to assess the compliance with the real relevance of an individual (*intended meaning*) w.r.t. the test queries probably a human expert would be more suitable than a reasoner.

6 Conclusions and Future Work

A merely deductive approach to retrieval may fall short in the real cases of heterogeneous knowledge bases integrating distributed knowledge sources. That was the reason for investigating other forms of retrieval based on different membership decision procedures. We have present a method for retrieval in knowledge bases expressed in Description Logics, founded in the *instance-based learning* and the *disjunctive version space* approach.

An instance-based learning method applied to DL representations that may serve to predict/suggest missing information about individuals in a knowledge base. Besides, the procedure is robust to noise and never made commission errors in the experiments that have been carried out so far.

Future work will concern a further investigation on ways to make the method more language-independent, so to apply it to more expressive DL languages, as those implemented in OWL. Moreover, we are studying the possibility of providing, together with each individual classification, also an estimate of its probability. Besides, the application of the method to the problem of Semantic Web Service discovery and retrieval is also foreseen.

References

- [1] F. Baader, D. Calvanese, D. McGuinness, D. Nardi, and P. Patel-Schneider, editors. *The Description Logic Handbook*. Cambridge University Press, 2003.
- [2] F. Baader and R. Küsters. Non-standard inferences in description logics: The story so far. In D. Gabbay, S. S. Goncharov, and M. Zakharyashev, editors, *Mathematical Problems from Applied Logic. New Logics for the XXIst Century*, volume 4 of *International Mathematical Series*. Kluwer/Plenum Publishers, 2005.
- [3] T. Berners-Lee, J. Hendler, and O. Lassila. The semantic web. *Scientific American*, May 2001.
- [4] W. Cohen and H. Hirsh. Learning the CLASSIC description logic. In P. Torasso, J. Doyle, and E. Sandewall, editors, *Proceedings of the 4th International Conference on the Principles of Knowledge Representation and Reasoning*, pages 121–133. Morgan Kaufmann, 1994.
- [5] C. d’Amato, N. Fanizzi, and F. Esposito. A dissimilarity measure for \mathcal{ALC} concept descriptions. In *Proceedings of the 21st Annual ACM Symposium of Applied Computing, SAC2006*, volume 2, pages 1695–1699, Dijon, France, 2006. ACM.

- [6] C. d'Amato, N. Fanizzi, and F. Esposito. Reasoning by analogy in description logics through instance-based learning. In G. Tummarello, P. Bouquet, and O. Signore, editors, *Proceedings of Semantic Web Applications and Perspectives, 3rd Italian Semantic Web Workshop, SWAP2006*, volume 201 of *CEUR Workshop Proceedings*, Pisa, Italy, 2006.
- [7] M. d'Aquin, J. Lieber, and A. Napoli. Decentralized case-based reasoning for the Semantic Web. In Y. Gil, V. Motta, E. Benjamins, and M. A. Musen, editors, *Proceedings of the 4th International Semantic Web Conference, ISWC2005*, volume 3279 of *LNCS*, pages 142–155. Springer, 2005.
- [8] F. M. Donini, M. Lenzerini, D. Nardi, and A. Schaerf. Deduction in concept languages: From subsumption to instance checking. *Journal of Logic and Computation*, 4(4):423–452, 1994.
- [9] F. Esposito, N. Fanizzi, L. Iannone, I. Palmisano, and G. Semeraro. Knowledge-intensive induction of terminologies from metadata. In F. van Harmelen, S. McIlraith, and D. Plexousakis, editors, *ISWC2004, Proceedings of the 3rd International Semantic Web Conference*, volume 3298 of *LNCS*, pages 441–455. Springer, 2004.
- [10] P. Haase, F. van Harmelen, Z. Huang, H. Stuckenschmidt, and Y. Sure. A framework for handling inconsistency in changing ontologies. In Y. Gil, V. Motta, E. Benjamins, and M. A. Musen, editors, *Proceedings of the 4th International Semantic Web Conference, ISWC2005*, volume 3279 of *LNCS*, pages 353–367. Springer, 2005.
- [11] P. Hitzler and D. Vrandečić. Resolution-based approximate reasoning for OWL DL. In Y. Gil, V. Motta, E. Benjamins, and M. A. Musen, editors, *Proceedings of the 4th International Semantic Web Conference, ISWC2005*, number 3279 in *LNCS*, pages 383–397. Springer, 2005.
- [12] T. Mantay. Commonality-based ABox retrieval. Technical Report FBI-HH-M-291/2000, Department of Computer Science, University of Hamburg, Germany, 2000.
- [13] T. Mitchell. *Machine Learning*. McGraw-Hill, 1997.
- [14] M. Schmidt-Schauß and G. Smolka. Attributive concept descriptions with complements. *Artificial Intelligence*, 48(1):1–26, 1991.
- [15] M. Sebag. Delaying the choice of bias: A disjunctive version space approach. In L. Saitta, editor, *Proceedings of the 13th International Conference on Machine Learning, ICML96*, pages 444–452. Morgan Kaufmann, 1996.
- [16] G. Teege. A subtraction operation for description logics. In P. Torasso, J. Doyle, and E. Sandewall, editors, *Proceedings of the 4th International Conference on Principles of Knowledge Representation and Reasoning*, pages 540–550. Morgan Kaufmann, 1994.

Uno Strumento Web di Esplorazione Interattiva Multidimensionale dei Dati Geo-Riferiti

Sonia Fernandes Silva e Paolo Menicori

Etruria Telematica Srl, Strada di Basciano, 22 - Monteriggioni 53035 Siena, Italia
{sonia.silva,menicori}@etelnet.it

Abstract. *Spatial OLAP* è la tendenza nel mercato *business intelligence* che utilizza principalmente rappresentazioni geografiche nell'analisi esplorativa dei dati multidimensionali geo-riferiti contenuti in un *DataWarehouse* (DW). L'utilizzo esclusivo di rappresentazioni cartografiche non permette però di sfruttare tutte le potenzialità che un DW può esprimere, ad es. l'analisi temporale. In questo lavoro si propone uno strumento web per l'esplorazione multidimensionale di dati geo-riferiti, le cui componenti visuali (mappa e diagrammi grafici) sono interattivamente sincronizzati. Diversamente dalle proposte esistenti, lo strumento consente agli utenti un'alta interattività, durante l'esplorazione multidimensionale sul web, per promuovere il suo utilizzo a chi ha bisogno di analizzare informazioni sommarie sulla rete. Questo articolo presenta i principali meccanismi interattivi grafici dell'interfaccia web e l'approccio tecnologico utilizzato nell'analisi multidimensionale dei dati geo-riferiti. Viene utilizzato il DW turistico dell'Osservatorio Economico della provincia di Siena, che contiene serie storiche statistiche relative alle presenze turistiche nelle strutture ricettive del territorio.

1 Introduzione

La tecnologia OLAP (OnLine Analytical Processing) [6] permette un rapido accesso, da un *DataWarehouse* (DW), ai dati aggregati in diverse prospettive multidimensionali (conosciuti nella letteratura come "cubi multidimensionali"), per facilitare processi decisionali in molte applicazioni (statistiche, finanziarie, ecc). Più specificamente, gli strumenti OLAP permettono un'esplorazione interattiva dei cubi multidimensionali sui quali sono applicati operatori di accesso/manipolazione ai cubi, quali slicing, dicing, rolling-up, drilling-down, ecc [13]. I cubi sono presentati in una rappresentazione tabulare [8], o come diagrammi grafici (chart statistici) per un'analisi visuale dell'informazione quantitativa [10].

D'altra parte, uno studio ha valutato che oltre l'80% delle informazioni delle basi dati pubbliche sono georiferibili in modo diretto (ad es.: direttamente collegabili ad unità territoriali quali Provincia, Regione, ecc) o indiretto (collegabili ad entità geografiche tramite l'indirizzo, il codice postale, ecc.) [11]. Viceversa, si usa anche definire entità geografiche in conformità a regole di *business*, quali l'area di vendita.

L'integrazione dei dati multidimensionali e geo-riferiti è vista come la più promettente tecnologia di informazione di appoggio alle decisioni, caratterizzando in

questo modo nella sua struttura interna un DW spaziale. Tuttavia, la tecnologia OLAP non è ottimizzata per approfondire l'analisi geografica. Viceversa i sistemi di informazioni geografici (SIG) non sono ottimizzati per approfondire l'analisi OLAP.

Spatial OLAP (SOLAP) [3], [18] è la tendenza attuale nel mercato *business intelligence* che accoppia le tecnologie SIG e OLAP in sistemi di supporto alla decisione. Tale tecnologia agisce come piattaforma che include la visualizzazione e navigazione geografica a supporto dell'analisi esplorativa multidimensionale dei dati geo-riferiti. Permette inoltre un'esplorazione interattiva spazio-temporale [1], [18] in molte applicazioni emergenti (ad es. sistemi di comunicazione mobile).

SOLAP utilizza principalmente la mappa choropleth [5], [16] per l'analisi della distribuzione quantitativa su entità poligonali, per evidenziare tendenze o anomalie geografiche riguardo ad un determinato fenomeno statistico. Tuttavia, tale metodo non sfrutta tutte le potenzialità (es.: l'analisi temporale) esprimibili da un DW ai fini del processo decisionale. La ricerca si è quindi concentrata nello sviluppo di strumenti innovativi [1], [4], [7], [9] che integrano la rappresentazione geografica con nuovi paradigmi di interazione e visualizzazione, in modo da rivelare automaticamente *patterns* nei dati (clusters, correlazione, tendenze, anomalie, ecc) durante l'analisi esplorativa multidimensionale.

Ispirandosi alle principali funzionalità individuate da questi prototipi, questo lavoro consiste nello sviluppo di uno strumento di esplorazione multidimensionale dei dati geo-riferiti, le cui componenti grafiche (mappa geografica e diagrammi grafici) sono interattivamente sincronizzate e in grado di fornire rapidamente informazioni di sintesi, principalmente quelle riferite alla dimensione temporale. L'idea principale è che tale strumento consenta agli utenti di esplorare tali dati sul web, per promuovere il suo utilizzo a chi ha bisogno di analizzare informazioni sommarie sulla rete. Nella pratica si sta sviluppando uno strumento web di accesso ed analisi visuale del DW turistico che sta dietro all'osservatorio economico della provincia di Siena, contenente serie storiche relative alle presenze turistiche nelle strutture ricettive del territorio.

Diversamente dalle proposte esistenti, tutte le interazioni sull'interfaccia web richiedono il minimo supporto del server. Tale caratteristica è resa possibile attraverso la combinazione di due tecniche: la tecnica di *basemaps* [21] per un'alta interattività con la mappa geografica sul web, dove la conoscenza geografica è codificata nell'immagine *raster* della mappa e trasmessa al *web client*; e l'utilizzo dei cubi multidimensionali codificati nella struttura dei dati *Statistics Tree* [12]. Tale struttura è stata estesa con un meccanismo di indicizzazione spazio-temporale, per un'alta interattività durante l'analisi spazio-temporale dei dati sommarie sul web. Questo articolo presenta i principali meccanismi interattivi grafici dell'interfaccia web e l'approccio tecnologico (*basemaps* + cubi spazio-temporali) che consente un'alta interattività ed un accesso immediato all'informazione di sintesi del DW turistico.

L'articolo è così strutturato: la Sezione 2 descrive la struttura multidimensionale del DW utilizzato e il metodo adottato nell'analisi esplorativa multidimensionale. La Sezione 3 illustra le componenti grafiche e i principali meccanismi interattivi dell'interfaccia web. La Sezione 4 riassume l'approccio tecnologico adottato nella analisi esplorativa dei cubi geo-riferiti sul web. La sezione 5 descrive brevemente le conclusioni e i lavori futuri.

2 Struttura Multidimensionale del DW Turistico

Lo strumento web è il risultato di un lavoro di ricerca sperimentale nel quale viene utilizzato il DW turistico dell'osservatorio economico (OE) della provincia di Siena (la struttura del DW descritta in questo articolo è più semplificata rispetto alla struttura originale). I dati turistici contengono un volume significativo di dati storici riferiti agli arrivi e presenze turistiche (indicatori statistici) nelle strutture ricettive (alberghiere ed extra-alberghiere) della provincia senese, che possono essere analizzate nelle diverse prospettive (dimensioni) e suoi corrispondenti livelli di aggregazione (Figura 1), quali:

- **Dimensione Spaziale**, rappresentante le strutture ricettive aggregate nelle seguenti unità territoriali: Comune, Circondario (gruppi di Comuni confinanti), Risorsa Turistica (gruppo di Comuni che fanno parte di una tipologia turistica, ad esempio, località termale, ecc) e APT (gruppi di Comuni riferenti a un'unità amministrativa turistica);
- **Dimensione Storica (temporale)**, rappresentante la componente temporale mensile, stagionale, annuale di arrivi/presenze turistiche; e
- **Dimensioni Tematiche (descrittive)**, che rappresentano la tipologia delle strutture, dalla qualità (numero di "stelle" nella tipologia alberghiera, di "spighe" in quella agrituristica, ecc.) alla tipologia generale (struttura alberghiera e extra alberghiera); e provenienza della clientela turistica, dal dettaglio (regione italiana, nazione, ecc.) al generale (provenienza italiana e straniera). La provenienza turistica, nonostante sia di natura geografica, è considerata come descrittiva nel DW.

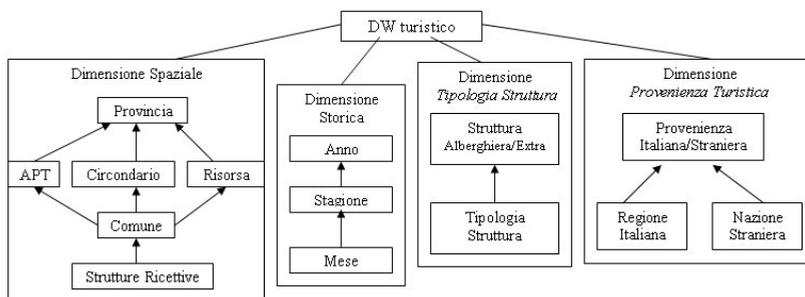


Fig. 1. Livelli di Aggregazione delle Dimensioni del DW Turistico

L'astrazione dei dati è un metodo efficace per fornire la visione sommaria di una grande quantità di dati a diversi livelli di dettaglio. In base al lavoro descritto in [20], tale astrazione rappresenta un *reticolo di cubi multidimensionale dei dati*. Tale reticolo è strutturato all'interno di una combinazione gerarchica dei livelli d'astrazione delle dimensioni coinvolte, dove ogni nodo caratterizza un cubo multidimensionale e ogni arco rappresenta un *drill-down/roll-up* di una specifica dimensione. Ad esempio, considerando la struttura gerarchica delle dimensioni del DW turistico illustrata nella Figura 1, il reticolo parte dal cubo più astratto (<provincia, anno, tipologia generale, provenienza generale>) ai cubi più dettagliati (<struttura, mese, tipologia struttura, regione italiana>, <struttura, mese, tipologia struttura, nazione>).

Le proiezioni sommarie di ogni cubo, a sua volta, compongono un *grafo di viste sommarie*, con numero di nodi corrispondente a $2^n - 1$ dimensioni del cubo [14], dove gli archi mostrano le possibilità di derivazione. Ad esempio, la vista sommaria $\langle \text{comune}, \text{tipologia} \rangle$ del cubo $\langle \text{comune}, \text{anno}, \text{tipologia}, \text{nazione} \rangle$ può essere derivata dalle viste $\langle \text{comune}, \text{anno}, \text{tipologia} \rangle$ e $\langle \text{comune}, \text{tipologia}, \text{nazione} \rangle$. Partendo da questo grafo di viste, possono essere eseguite diversi tipi di analisi. Secondo [20], l'analisi multidimensionale dei dati si effettua, in senso astratto, "navigando" il reticolo in modo interattivo (così da incrementare o decrementare i livelli di astrazione dimensionali con cui le misure vengono analizzate).

Lo scopo principale di questo lavoro è arrivare ad analizzare da remoto le tendenze storiche degli indicatori tematici del DW turistico sul territorio senese, alle diverse granularità spaziale e temporale. Tale scopo nasce dalla richiesta di arricchire la strategia precedente di accesso all'informazione sommaria dei dati turistici dell'OE sul web, insoddisfacente per eseguire analisi OLAP, perché restrittiva. Il precedente approccio espone infatti analisi già preconfezionate (le richieste devono essere progettate a priori) e quindi non flessibili, non permettendo l'analisi da nuove e diverse prospettive. L'interfaccia web che implementa questa originaria strategia (Figura 2) esegue interrogazioni *on-line* direttamente sui dati di dettaglio e il calcolo delle informazioni sommarie è fatto automaticamente ad ogni richiesta.

Comune	2004	2005	%	2004	2005	%	2004	2005	%	2004	2005	%
AREZZANO	27	0	-100%	945	0	-100%	1475	0	-100%	4944	0	-100%
BIANCOLEONE	46	0	-100%	725	0	-100%	841	0	-100%	2743	0	-100%
BUCCHINICCI	36	0	-100%	415	0	-100%	595	0	-100%	1449	0	-100%
CALCI	46	0	-100%	972	0	-100%	1285	0	-100%	4924	0	-100%
CALTANICASSA	87	0	-100%	1835	0	-100%	4783	0	-100%	16012	0	-100%
CALTANICASSA S. GIOVANNI	104	0	-100%	1440	0	-100%	2772	0	-100%	5959	0	-100%
CALTANICASSA S. MARCO	53	0	-100%	687	0	-100%	1084	0	-100%	2795	0	-100%
CANTALICE	33	0	-100%	283	0	-100%	329	0	-100%	1213	0	-100%
CANTALICE S. MARCO	245	0	-100%	15178	0	-100%	30787	0	-100%	106474	0	-100%
CARRARA	38	0	-100%	823	0	-100%	1504	0	-100%	4259	0	-100%
CARRARA S. MARCO	39	0	-100%	1134	0	-100%	1774	0	-100%	3926	0	-100%
CARRARA S. MARCO S. MARCO	58	0	-100%	931	0	-100%	2048	0	-100%	4957	0	-100%
CARRARA S. MARCO S. MARCO S. MARCO	84	0	-100%	894	0	-100%	1596	0	-100%	5982	0	-100%
CARRARA S. MARCO S. MARCO S. MARCO S. MARCO	90	0	-100%	1085	0	-100%	2847	0	-100%	4721	0	-100%
CARRARA S. MARCO S. MARCO S. MARCO S. MARCO S. MARCO	114	0	-100%	2010	0	-100%	4456	0	-100%	13386	0	-100%
CARRARA S. MARCO S. MARCO S. MARCO S. MARCO S. MARCO S. MARCO	47	0	-100%	999	0	-100%	2288	0	-100%	6874	0	-100%
CARRARA S. MARCO	28	0	-100%	779	0	-100%	269	0	-100%	743	0	-100%
CARRARA S. MARCO	14	0	-100%	294	0	-100%	830	0	-100%	2045	0	-100%
CARRARA S. MARCO	38	0	-100%	1073	0	-100%	1713	0	-100%	4077	0	-100%
CARRARA S. MARCO	19	0	-100%	174	0	-100%	48	0	-100%	134	0	-100%
CARRARA S. MARCO	80	0	-100%	1034	0	-100%	2448	0	-100%	6842	0	-100%
CARRARA S. MARCO	81	0	-100%	1274	0	-100%	4492	0	-100%	12132	0	-100%

Fig. 2. Proiezione Sommaria per Comune (Rappresentazione Tabulare)

Ad esempio, partendo dalla selezione della voce "Report & Statistiche" nel frame sinistro dell'interfaccia web, appare la restrizione selettiva di indicare uno specifico anno e mese. Il dataset risultante sarà la proiezione sommaria delle misure aggregate per comune e mese selezionato in due anni consecutivi. Tale proiezione sommaria è visualizzata in una rappresentazione tabulare, come illustra la Figura 2. Ovviamente tale rappresentazione non è adeguata per poter evidenziare automaticamente tendenze e anomalie spaziali o temporali rispetto a un fenomeno statistico. Partendo da queste limitazioni, si stanno sviluppando due applicativi web come interfacce di accesso e analisi visuale del DW turistico: uno finalizzato all'analisi esplorativa

multidimensionale dei dati sommari; l'altro finalizzato all'accesso e analisi dei dati di dettaglio relativi alle singole strutture ricettive. Questo articolo descrive soltanto il primo applicativo web. A questo fine, si è pianificato di implementare su web la strategia di esplorazione multidimensionale di [20], in due tappe. La tappa iniziale permetterà da remoto all'utente di impostare un'interrogazione analitica al DW. Ciò avviene specificando i seguenti parametri in una pagina *web*: misure statistiche, dimensioni tematiche e corrispondenti livelli di aggregazione, statistiche (somma, ecc), e periodo storico. La tappa successiva permetterà all'utente di analizzare interattivamente i dati sommari risultanti di tali interrogazioni, attraverso un'interfaccia web. I dati sommari sono visualizzati al livello più alto della granularità spaziale e temporale, e possono essere esplorati progressivamente, attraverso meccanismi interattivi grafici. La prossima sezione descrive l'interfaccia web (nominata *Geo-Multiviews*) utilizzata nella seconda tappa.

3 Interfaccia Web

La interfaccia *Geo-Multiviews* contiene una barra degli strumenti (con menu, *checkboxes*, ecc) e le seguenti componenti grafiche (Figura 3):

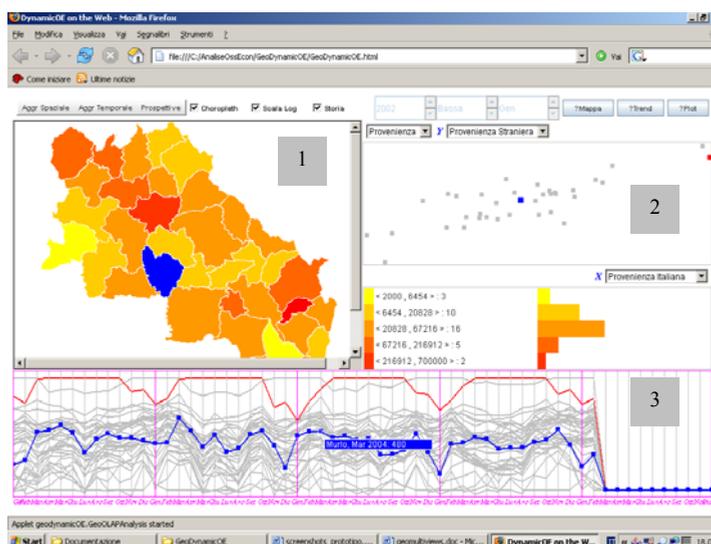


Fig. 3. Interfaccia Web *Geo-Multiviews*.

- 1) La mappa choropleth e la corrispondente legenda istogramma, dove gli oggetti geografici sono le unità territoriali colorate in tono graduale e le diverse intensità dei colori codificano la scala dei valori di una statistica. La legenda riassume la distribuzione statistica dell'indicatore analizzato. Al momento esiste solo la classificazione degli intervalli unici in modalità lineare e logaritmica (questa ultima rende la distribuzione statistica più espressiva nella mappa e nei diagrammi grafici).

- 2) Il diagramma di dispersione (*scatterplot*), che rappresenta la distribuzione statistica bidimensionale di tutti gli oggetti geografici (rappresentati visivamente come punti) in accordo con le dimensioni tematiche del DW. Tale diagramma permette di paragonare simultaneamente indicatori relativi a due membri di una stessa prospettiva (ad es., *presenze turistiche: residence X alberghi 5 stelle*).
- 3) Il diagramma serie storica, (*timeline*) per l'analisi interattiva del *trend* evolutivo di tutti gli oggetti geografici (rappresentati visivamente come linee nel grafico) in riferimento a una misura statistica, durante il periodo storico specificato nella pagina web della tappa iniziale.

Le figure qui illustrate sono alcuni esempi di come analizzare dinamicamente la reportistica multidimensionale del DW turistico in riferimento ai livelli di aggregazione tematici *tipologia di qualità* delle strutture ricettive (alberghi 5 stelle, residence, ecc) e *provenienza generale* (italiana/straniera) nel periodo dal 2002 al febbraio/2006. La figura 3 illustra una schermata di un *report* interattivo che si riferisce alla distribuzione (logaritmica) degli arrivi di tutte le strutture ricettive a livello Comunale e mensile, considerando tutte le provenienze turistiche, e con la distribuzione bidimensionale che discrimina la provenienza in straniera e italiana.

3.1 Aspetti Interattivi

Le componenti grafiche sono *interattivamente sincronizzate* [2], [17]. Quando l'utente muove il mouse su un oggetto geografico in una specifica componente grafica (poligono nella mappa, linea nella serie storica), lo stesso oggetto è evidenziato (in colore blu) in altre componenti. Tutte le interazioni sull'interfaccia web sono in tempo reale perché richiedono il minimo supporto del server. Ciò è reso possibile attraverso la implementazione di due tecniche innovative (descritte nella prossima sezione) che permettono la creazione dinamica di diversi *report* grafici di sintesi a un tempo di risposta costante, tramite meccanismi interattivi che includono:

- *Drill-Down/Roll-up* Spaziale e Temporale: la mappa *choropleth* e la serie storica sono entrambe navigabili da un livello geografico (temporale) di aggregazione a un altro più dettagliato e vice-versa, mantenendo gli stessi livelli di aggregazione nelle dimensioni tematiche e temporale (spaziale) [18]. Ad esempio, la Figura 4 illustra una sequenza di un *drill-down* temporale degli arrivi comunali nelle strutture ricettive agrituristiche.
- *Classificazione Tematica Orientata al Tempo*: la classificazione tematica è istantaneamente modificata tramite la selezione singola/multipla dei membri delle dimensioni tematiche coinvolte (nel menu "*Prospettive*" nella barra degli strumenti). Partendo dalla statistica scelta, la mappa *choropleth* e lo *scatterplot* sono modificati in base ad un determinato istante (ad es., *maggio 2005*). Il *checkbox* "*Storia*" (nella barra degli strumenti), se attivo, visualizza i valori cumulativi dell'indicatore analizzato nel periodo storico specificato. La successione di periodi diversi ha un effetto di animazione nella mappa e nello *scatterplot*. Ad esempio, la Figura 5 illustra una sequenza istantanea di arrivi agrituristiche (a livello Circondario) nella alta stagione di 2003-2004.

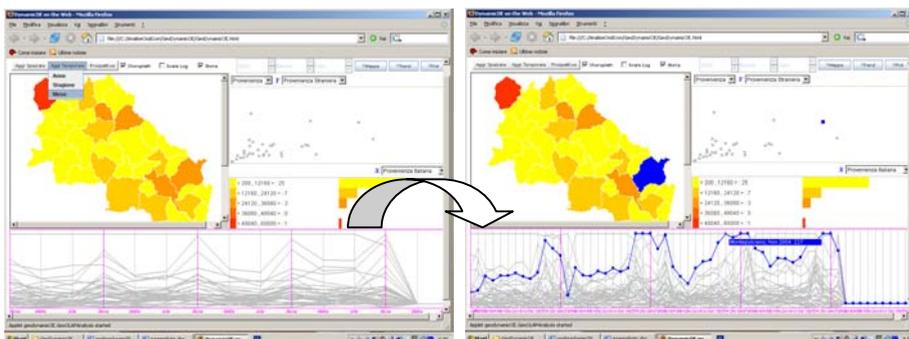


Fig. 4. Sequenza di un *Drill-Down Temporale* (*Stagione* → *Mese*)

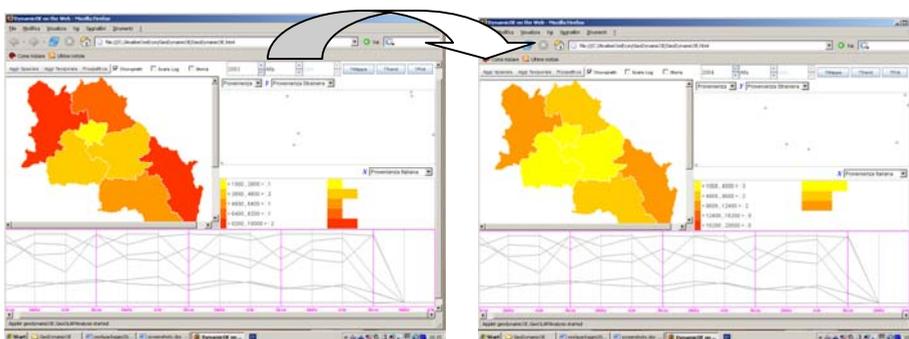


Fig. 5. Sequenza di *Classificazione Istantanea* (*Alta Stagione 2003* → *Alta Stagione 2004*)

- Filtraggio Bidimensionale e Periodico: Si può eseguire dei filtri bidimensionali e temporali degli oggetti geografici attraverso il disegno di un'area di interesse (*box*) nei diagrammi grafici. Ad esempio, la figura 6 illustra una sequenza interattiva di un filtraggio sulla serie storica mensile dei Comuni che hanno avuto un alto valore di arrivi nelle strutture agrituristiche nell'anno 2004 (disegno di un *box* dal gennaio al dicembre 2004 nella serie storica). La funzionalità *choropleth* (nella barra degli strumenti) è stata disabilitata nella terza sequenza per meglio evidenziare i Comuni filtrati.

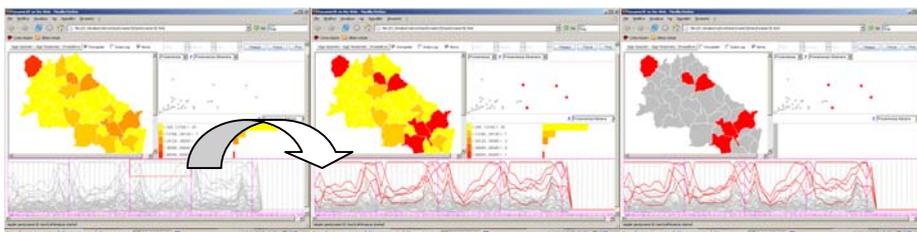


Fig. 6. Sequenza Interattiva di un Filtraggio sulla Serie Storica

- Dettaglio a richiesta: è sempre disponibile un'identificazione immediata dell'informazione statistica dell'oggetto geografico evidenziato nella mappa e nei

diagrammi grafici. Ad esempio, selezionando i piccoli quadrati di una specifica linea nella serie storica, si ottiene il valore preciso della statistica in una specifica data (*Comune Murlo* con 484 arrivi in marzo 2004, nella figura 4). Si può anche paragonare progressivamente diversi oggetti geografici rispetto ad uno specifico oggetto selezionato (evidenziato in rosso nella Figura 4).

4 Analisi Esplorativa dei Cubi Geo-Riferiti sul Web

Per sviluppare lo strumento web descritto in precedenza, si sono verificate le possibilità tecnologiche relative ai meccanismi di accesso e analisi storica dei *cubi multidimensionali geo-riferiti*, cioè, cubi di dati in cui dimensioni e misure spaziali includono le figure geometriche georiferibili (al momento questo lavoro non considera le misure spaziali). Gli strumenti SOLAP si basano principalmente sull'implementazione di funzionalità geografiche nelle analisi OLAP. Tuttavia, la combinazione di queste funzionalità con paradigmi di interazione e visualizzazione esplorativa è ancora agli albori in ambiente web. Si è quindi adottata una soluzione alternativa, basata sulla personalizzazione di una combinazione di tecnologie.

Per dare una risposta alla crescente domanda di forte interattività con la visualizzazione cartografica, la modalità oggi più comune di accesso remoto ai dati geo-riferiti è tramite *Internet Map Services*. L'interazione dinamica con la mappa avviene in genere tramite l'invio di dati geografici vettoriali. Tuttavia, tale approccio tecnologico è dipendente da fattori legati alla tecnologia internet (variazione degli scenari di connessione, banda di rete, ecc.) che possono imporre restrizioni ad una forte interattività con la mappa durante l'analisi esplorativa dei dati geo-riferiti. Abbiamo quindi adottato la soluzione descritta in seguito che è più ottimizzata.

4.1. *BaseMap*: Immagine Raster con Conoscenza Geografica

Nella tecnica di manipolazione dell'immagine descritta in [21], la conoscenza geografica è codificata nell'immagine *raster* della mappa (nominata *basemap*), attraverso i colori RGB, e trasmessa al *web client*. Esiste quindi una tripletta RGB per ogni oggetto geografico, generata da funzioni (*basemap encoding*) che calcolano un colore univoco partendo da un valore intero (l'indice del singolo oggetto geografico facente parte di un geodataset), ed esiste una funzione (*basemap decoding*) che restituisce un valore intero partendo dalla tripletta RGB.

Secondo [21], il formato GIF è il migliore per una *basemap* perché ha un alto rapporto di compressione (quindi tempi accettabili per il trasferimento via rete), mantenendo una buona qualità dell'immagine. Tuttavia, un'immagine GIF può supportare al massimo 256 colori (quindi, può codificare al massimo 256 oggetti geografici). Nella rappresentazione *basemap* delle mappe che eccedono 256 oggetti geografici, [21] propone un meccanismo aggiuntivo di codifica di colori in 2 livelli: ogni colore univoco del *basemap* del primo livello punta a un gruppo di colori univoci nel secondo livello. In questo modo, due immagini *basemaps* in livelli possono codificare teoricamente 65536 ($256 * 256$) oggetti geografici.

La Figura 7 illustra le immagini *basemaps* delle mappe poligonali a livello di *Circondario* e *Comune* della provincia senese e le immagini *basemaps* in 2 livelli che codificano le strutture ricettive della provincia (il colore nero rappresenta il *background* della mappa e i confini poligonali).



Fig. 7. *Basemaps* della Provincia Senese a livello di *Circondario*, *Comune* e *Struttura Ricettiva*

La tecnica di *basemap encoding/decoding* non soltanto permette di realizzare un'applicazione *client* più semplificata, ma rende anche possibile l'interattività dinamica in tempo reale con la mappa con il minimo supporto del server. Inoltre, possiede caratteristiche (quali un breve tempo di *download* iniziale e scalabilità costante di performance interattiva), indipendenti dalla numerosità degli oggetti geografici e dal numero di utenti che accedono simultaneamente all'applicazione web.

Questo meccanismo non è però sufficiente a rendere ottimale l'analisi esplorativa su web dei cubi geo-riferiti, perché tale interazione richiederebbe un processamento continuo delle interrogazioni sul DW. Sarebbe necessario adottare anche un server OLAP, ma nella pratica uno strumento OLAP non realizza, durante l'esplorazione multidimensionale dei dati via web, lo stesso effetto "*smooth*" (feedback immediato) ottenuto utilizzando le *basemaps*. Quindi, invece di combinare la tecnica *basemaps* con un server OLAP, si è deciso di ricercare meccanismi d'indicizzazione dei dati sommari che non compromettessero l'accesso e la manipolazione remota dei cubi geo-riferiti. Tale tecnica è qui di seguito descritta.

4.2. Struttura di Cubi Sommari Spazio-Temporali

Invece d'utilizzare uno specifico server OLAP, si è adottato l'approccio CubiST++ (*Cubing with Statistics Tree*) [12] poiché semplice e flessibile da adattare alle nostre specifiche esigenze. CubiST++ è stato sviluppato per rispondere in modo efficiente ad una classe di interrogazioni sommarie, che restituiscono soltanto dati aggregati. In particolare, utilizziamo una classe specifica di interrogazioni sommarie che coinvolgono una combinazione arbitraria delle dimensioni tematiche (ognuna ad uno specifico livello gerarchico) e a tutti i livelli gerarchici spaziali e temporali.

In CubiST++, un cubo multidimensionale è strutturato internamente in una ST (*Statistics Tree*), partendo da un'unica scansione dei dati di massimo dettaglio contenuti nel DW. Una ST è una struttura in "*albero a livelli*" dove i nodi interni rappresentano le dimensioni e ogni elemento del nodo (corrispondente a un specifico valore del dominio di una dimensione) è un *pointer* a un nodo del prossimo livello. Il grado di un nodo interno corrisponde alla cardinalità della dimensione più un *pointer* speciale che indica tutti i valori del dominio della dimensione corrispondente. I nodi-foglia invece contengono informazioni aggregate di una specifica misura.

CubiST++ non usa una unica ST, ma una famiglia di STs derivate da una *ST di base* (contenente le dimensioni al più basso livello gerarchico), in modo da velocizzare le interrogazioni con restrizione rispetto ai livelli gerarchici delle dimensioni coinvolte. CubiST++ usa un meccanismo che seleziona la struttura ST dalla famiglia che è più ottimizzata per rispondere a una interrogazione analitica.

Il vantaggio di questo approccio è che esso non considera il problema della selezione di quali viste sommarie devono essere materializzate perché manipola tutte le possibili viste (cubo multidimensionale completo). Ha inoltre una scalabilità superiore (comprovata dai test di valutazione descritti in [12]) rispetto ai meccanismi d'indicizzazione usati nelle strategie ROLAP/MOLAP, se si considerano soprattutto le performance in tempo di esecuzione. La struttura ST, sebbene ottimizzata per rispondere ad una interrogazione analitica, è comunque trasmessa al *web client* come un container compresso. E poiché una singola ST è riferita ad una granularità spaziale e temporale per volta, il processo interattivo *drill-down/roll-up* sulle dimensioni spaziale e temporale può causare un trasferimento continuo di STs sulla rete e compromettere l'alta interattività, già raggiunta grazie al meccanismo *basemaps*.

In base a questa ulteriore esigenza, abbiamo implementato una struttura ibrida che estende la struttura ST con un meccanismo di indicizzazione spazio-temporale (la descrizione dettagliata di questa estensione è riassunta in [19]). L'accesso e la manipolazione dinamica di questa nuova struttura rendono fortemente interattiva l'esplorazione visuale spazio-temporale, a qualunque livello gerarchico.

4.3 Architettura Prototipale

L'implementazione di queste due tecniche fa parte di un'architettura prototipale di applicativi *web* personalizzati per esplorazione e l'analisi visuale di dati sommarie geografici. La Figura 8 illustra i moduli che compongono l'architettura.

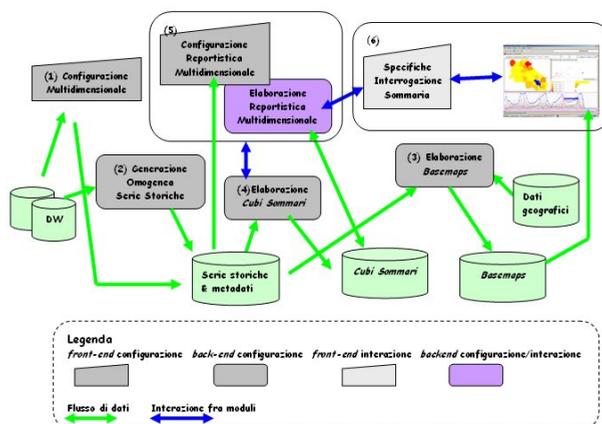


Fig. 8. Architettura Prototipale

Il modulo *Elaborazione di Basemaps* rappresenta la creazione delle immagini *raster* delle mappe necessarie per l'applicazione *web*, dove si utilizza un *map engine*

tool che individuano gli oggetti geografici di una mappa a uno specifico livello di aggregazione, definiti nei metadati multidimensionali del DW (generata nel modulo 1). Tutte le immagini sono archiviate in un *repository* di *Basemaps*.

Il modulo *Elaborazione dei Cubi Sommari* rappresenta la codifica dei *cubi di base* nelle strutture *ST*. Vengono generati periodicamente attraverso la scansione delle tuple corrispondenti alla serie storica omogenea del DW (generata nel modulo 2) e corrispondenti metadati multidimensionali. I cubi di base sono archiviati come *cointaner* compressi in un *repository* di *Cubi Sommari*. Partendo dai cubi di base, nuovi cubi sommari sono *derivati* come superviste dei diversi livelli di aggregazione delle dimensioni tematiche relative ai cubi di base (modulo 5). Ci sono due strategie di derivazione: nella prima si scelgono i livelli di aggregazione delle dimensioni tematiche attraverso un'interfaccia *wizard*, mentre nella seconda strategia si utilizza l'algoritmo *greedy* di derivazione automatica descritto in [12].

5 Conclusione

Questo articolo ha presentato uno strumento web per la visualizzazione e l'analisi di dati sommari geo-riferiti, dove viene utilizzato il DW turistico che sta dietro all'osservatorio economico della provincia di Siena, contenente serie storiche relative alle presenze turistiche nelle strutture ricettive del territorio. Lo strumento consente agli utenti di compiere navigazioni multidimensionali con un altissimo livello di interattività, integrando e rendendo accessibili da un'unica interfaccia sia la mappa geografica che le tecniche di visualizzazione grafica.

Grazie alla codifica della conoscenza geografica nelle *basemaps* ed agli algoritmi d'accesso e manipolazione dei cubi sommari spazio-temporali, non è necessario un supporto del server quando si interagisce con la mappa geografica ed i diagrammi grafici. In questo modo si ha un accesso immediato all'informazione di sintesi utilizzando meccanismi interattivi grafici che facilitano l'analisi del *trend* evolutivo, degli aspetti di similarità, delle anomalie sui dati geo-riferiti. Tali meccanismi di integrazione ed analisi delle informazioni costituiscono un valido strumento di supporto per processi decisionali distribuiti.

Lavori futuri riguardano l'estensione dell'interfaccia web al fine di arricchire ulteriormente l'analisi visuale/esplorativa multidimensionale dei dati geo-riferiti, in particolare: a) approfondire l'analisi multidimensionale attraverso l'inclusione della componente grafica *coordinata parallela* [15], b) approfondire l'analisi tematica attraverso l'aggiunta dei nuovi metodi di classificazione statistica (*quantile*, *deviazione standard*, ecc) che si riflettono nella mappa *choropleth* e diagrammi grafici, c) Aggiungere nuove tecniche di visualizzazione nell'analisi interattiva spazio-temporale, ad esempio, la tecnica *map series* [1], [18].

Ringraziamenti. Si ringraziano l'Amministrazione Provinciale di Siena e la Fondazione Monte dei Paschi di Siena come enti patrocinanti di questa ricerca specificata nel contratto di convenzione per l'attuazione del progetto "Attivazione di figure professionali specialistiche quali integratori di ricerca e sviluppo territoriale". Si ringraziano anche i revisori anonimi per i loro utili suggerimenti.

Riferimenti Bibliografici

1. Andrienko, N., Andrienko, G.: Interactive Visual Tools to Explore Spatio-Temporal Variation. Proc. of ACM Symposium on Advanced Visual Interfaces (2004) 417-420
2. Becker, R.A., Cleveland, W.S.: Brushing Scatterplots. Dynamic Graphics for Statistics. McGraw-Hill Editor (1988)
3. Bimonte, S., Tchounikine, A., Miquel, M.: Towards a Spatial Multidimensional Model. Proceedings of DOLAP'05, Bremen. Germany (2005) 39-46.
4. Brodbeck, D., Girardin, L.: Design Study: Using Multiple Coordinated Views to Analyse Geo-Referenced High-dimensional Datasets. Proc. of IEEE Int. Conference on Coordinated & Multiple Views in Exploratory Visualization (2003) 104-111
5. Campbell, J.: Map Use and Analysis. McGraw-Hill Boston (2001)
6. Codd, E.F., Codd, S.B., Salley, C.T.: Providing OLAP (on-line analytical processing) to user-analysts: An IT mandate. Technical Report (1993)
7. Dang, G., North, C., Shneiderman, B.: Dynamic Queries and Brushing on Choropleth Maps. IEEE Proc. of the Fifth Int. Conference on Information Visualization (2001) 757-764
8. Desmarais, B.: Using the Microsoft Excel Pivot Table for Reliability Applications. Reliability – Investigating in the Future. IEEE 34th Annual Spring (1996) 79-91
9. Edsall, R.M., MacEachren, A.M., Pickle, L.: Case Study: Design and Assessment of An Enhanced Geographic Information System for Exploration of Multivariate Health Statistics. IEEE Symposium on Information Visualization (2001)
10. Eick, S.G.: Visualizing multi-dimensional data. ACM SIGGRAPH Computer Graphics. **34** (2000) 61-67
11. Franklin, C.: An Introduction to Geographic Information Systems: Linking Maps to databases. Database (1992) 13-21
12. Fu, L., Hammer, J.: CubiST++: Evaluating Ad-Hoc CUBE Queries Using Statistics Trees". Journal of Distributed and Parallel Databases, 14. Kluwer Academic Publishers Manufactured in The Netherlands (2003) 221-254.
13. Gray, J. et al.: Data cube: A relational aggregation operator generalizing group-by, cross-tab, and sub-totals. Data Mining and Knowledge Discovery, Vol. 1 (1997) 29-53
14. Harinarayan, V., Rajaraman, A., Ullman, J.D.: Implementing Data Cubes Efficiently. ACM SIGMOD (1996) 205-216
15. Inselberg, A.: Multidimensional Detective. IEEE Symposium on Information Visualization (1997) 100-107
16. Kumar, N.: Mapping Spatial and Statistical Distributions in a Choropleth Map. ArcUser 6 4 (2003) 48-49.
17. North, C., Shneiderman, B.: Snap-Together Visualization: Can Users Construct and Operate Coordinated Views? Int. Journal of Human-Computer Studies. **5** 53 (2000) 715-739
18. Rivest, S. et al.: SOLAP technology : Merging business intelligence with geospatial technology for interactive spatio-temporal exploration and analysis of data. ISPRS Journal of Photogrammetry & Remote Sensing **60** (2005) 17-33
19. Silva, S.F., Schiel, U.: Case-Study: Towards a Highly Interactive Web Exploration of Spatial and Historical Aggregations from a Tourist DataWarehouse. Paper Submitted to 9th Int. Conf. on Data Warehousing and Knowledge Discovery (DaWaK 2007).
20. Stolte, C., Tang, D., Hanrahan, P.: Multiscale Visualization using Data Cubes. IEEE Symposium on Information Visualization (2002) 7-14
21. Zhao, H., Shneiderman, B.: Colour-coded pixel-based highly interactive Web mapping for geo-referenced data exploration. International Journal of Geographical Information Science. **19** 4 (2005) 413-428

Tecnologie Cognitive nei Sistemi per la Modellistica Geo-spaziale

Alberto Gemelli, Claudia Diamantini, Domenico Potena

Dipartimento di Ingegneria Informatica, Gestionale e dell'Automazione,
Università Politecnica delle Marche - via Brecce Bianche, 60131 Ancona, Italy
{gemelli,diamantini,potena}@diiga.univpm.it

Sommario Le tecnologie informatiche per la modellistica dei sistemi complessi convergono verso il paradigma cognitivo. Si tratta dello sviluppo di quelle tecniche che realizzano schemi o proprietà simili a quelle dell'intelligenza umana, fra cui la capacità di adattarsi a situazioni impreviste ovvero di prestarsi a rappresentare una realtà complessa e dinamica. Questa prospettiva porta a progettare sistemi informatici di elevata robustezza ed usabilità, dotati di proprietà decisionali. Le tecnologie in questione sono: Reti Neurali, in particolare le Self-Organizing Maps, Sistemi Multiagente, Automi Cellulari. Un banco di prova ottimale per queste tecniche sono i Sistemi Informativi Geografici (GIS) per la complessità della conoscenza che essi sono chiamati a rappresentare: quella geografica, con tutte le fenomenologie associate. Un survey sulla applicazione ai GIS delle tecniche cognitive mette in risalto le proprietà di adattatività e capacità di modellare una realtà complessa attraverso comportamenti emergenti. Si evidenzia un paradigma di valutazione dei modelli di conoscenza, non soltanto basato sull'accuratezza in un dominio specifico, ma anche sulla adeguatezza dei modelli nei confronti di aspetti qualitativi della conoscenza.

1 Introduzione

Nell'ambito dei sistemi informativi geografici, l'obiettivo di questo lavoro è quello di caratterizzare un gruppo di tecniche di modellazione dei dati secondo la prospettiva delle scienze cognitive ovvero dei processi riguardanti la mente, la memoria e l'intelligenza.

Un *sistema informativo geografico* è un sistema basato su computer che consente il modellamento, immagazzinamento, interrogazione, condivisione, manipolazione, analisi e presentazione di dati geograficamente riferiti (o dati geospaziali) [32]. Per il GIS si va configurando un ruolo sempre più importante nella gestione del territorio. Inglobando funzionalità di ragionamento automatico, il GIS svolgerà di routine ed autonomamente le operazioni di estrazione della conoscenza profonda nei dati, riservando all'operatore umano un ruolo di supervisore. Le applicazioni sono diverse: come strumento di supporto alla decisione, regolamentazione dell'uso del territorio, valutazione del rischio ambientale, monitoraggio ambientale, ricerca scientifica.

Oltre alle problematiche relative al ragionamento automatico, occorre considerare anche i temi inerenti al dominio della conoscenza geografica. Questo lavoro pone l'attenzione sui modelli geospaziali, ovvero simulazioni dello spazio geografico con le sue fenomenologie, e sui metodi impiegati per realizzare tali modelli. L'ambito è quello della *geocomputation*, come definita da [20], cioè l'elaborazione finalizzata alla soluzione di problemi di grande complessità in materia di studi geografici. La geocomputation include metodologie di data mining e tecniche di intelligenza artificiale. Nell'ottica della geocomputation si ravvisa la necessità di realizzare uno schema generale in cui tutte le tecnologie di modellazione siano inquadrare e distinguibili per proprietà e ambito di applicazione. Occorre soprattutto valutare la capacità dei modelli di rappresentare specificamente i sistemi geografici. Non è negli scopi di questo lavoro descrivere i dettagli dei sistemi per l'estrazione della conoscenza, bensì individuare dei parametri per qualificare la conoscenza che le tecnologie impiegate nella geocomputation sono in grado di gestire e produrre. Nel seguito si esaminano le proprietà dell'informazione geospaziale, dalle primitive di rappresentazione ai modelli più complessi, attraverso la sperimentazione che vari autori hanno condotto.

2 La Rappresentazione Computazionale del Dato Geografico

Un cenno sulla rappresentazione computazionale dell'informazione geografica è necessario prima di procedere negli aspetti avanzati della modellistica, per fissare le proprietà dell'elemento di riferimento: il dato geospaziale. Una definizione di informazione geografica tratta da [13] è la seguente: *l'informazione geografica* è l'informazione sulle caratteristiche ed i fenomeni localizzati in prossimità della superficie della Terra. Ciò che distingue questo particolare tipo di informazione è la presenza di un riferimento ad una posizione geografica. Tutte le informazioni geografiche possono essere ridotte alla semplice definizione che in qualche punto esiste una istanza di una qualche generalmente riconoscibile cosa, che può essere una classe, una proprietà, la misura di una variabile, un'attività, un organismo, o più generalmente un concetto qualsiasi tra miriadi di possibilità. Formalmente l'elemento primitivo fondamentale di una informazione geografica è una tupla di valori, detta *dato geospaziale*, che possiamo porre alla base della geocomputation:

$$\{x, y, z, t, U\} \quad (1)$$

Dove, dato un riferimento spaziale: x, y sono le coordinate, z è l'elevazione, t , il tempo, e U l'elemento geografico associato. Al livello di strutture di dati, i dati geospaziali possono essere aggregati in raster e strutture vettoriali.

2.1 Modelli Geospaziali Dinamici: Cognitività nella Rappresentazione della Complessità

Nei modelli dinamici altri aspetti profondi dell'informazione geospaziale, oltre a quelli delle primitive, sono tanto noti quanto basilari. Quello geografico è un

sistema complesso, lo si può vedere come composto da un gran numero di entità che mostrano un alto livello di interattività. La natura di questa interattività è tipicamente non lineare e comprende molteplici cicli di retroazione. Il risultato è un sistema in cui può essere molto difficile associare gli effetti con le cause [26]. Una volta scomposto nelle parti più semplici, la conoscenza emerge nello studio delle relazioni tra queste parti.

Nel complesso, il sistema geografico mostra le seguenti proprietà [26] [22] [28]:

- Memoria: il sistema ha una memoria della sua storia passata registrata nel suo stato attuale.
- Molteplicità di comportamenti: come effetti delle relazioni non lineari che esso include.
- Caoticità: l'evoluzione del sistema è incredibilmente sensibile a perturbazioni anche minime, e talvolta praticamente insensibile a forti sollecitazioni.
- Emergenza: può essere definita come il processo di formazione di schemi complessi a partire da regole più semplici, ovvero quando un numero di entità semplici operano in un ambiente, dando origine a comportamenti più complessi come collettività.
- Incomprimibilità: è impossibile rappresentare completamente, senza approssimazioni, un sistema complesso con un altro sistema più semplice.

Per dominare questa complessità è stata proposta la soluzione cognitiva, ovvero quell'insieme di discipline, definite *scienze cognitive* [16], che studiano il rapporto tra la mente umana e l'ambiente circostante, in termini di rappresentazione, ragionamento, percezione, con vari apporti disciplinari dalla linguistica alla filosofia. I *modelli cognitivi*, nella fattispecie, dello spazio geografico [18] legano la cognizione geografica ai sistemi informativi geografici allorché nella estrazione e rappresentazione computazionale della conoscenza si adoperano le strategie della cognitivtà. Anche nelle conclusioni degli autori [11] [18], la base di conoscenza dei GIS deve essere costituita da modelli cognitivi. La ragione dell'interesse verso la cognitivtà sta nel fatto che l'informazione geografica è raccolta ed interpretata primariamente dalla mente umana, quindi per immagazzinare e rappresentare tale informazione nelle macchine, in modo efficiente ed umanamente fruibile, ci si deve confrontare con le modalità con cui la geografia e lo spazio sono percepite ed elaborate dalla mente umana. Concludendo, l'individuazione di un modello geografico coincide con le problematiche della rappresentazione di un sistema complesso, mentre l'implementazione deve seguire le strategie della cognitivtà.

3 Modelli Geospaziali: Casi Applicativi

In questa sezione viene proposta una casistica di modelli geospaziali dinamici e delle loro affinità con i concetti cognitivi e di complessità. Una prima differenziazione dei modelli è desumibile dalla letteratura. Storicamente, nell'ambito del data mining, i modelli si distinguono sulla base della teoricità del contenuto [27], ovvero a seconda che la conoscenza espressa dal modello sia o non sia

esplicitamente posta in una forma teoretico-simbolica. Nel seguito cominceremo con l'espone una casistica distinta per ognuno di questi due approcci, successivamente concluderemo che tale distinzione nella rappresentazione della conoscenza non è determinante nelle applicazioni geospaziali, lo è invece la posizione del modello in una scala di *profondità* di conoscenza, ovvero la sua capacità di esprimere relazioni complesse tra i dati. Questa capacità si vuol mettere in evidenza come la qualità determinante ai fini della classificazione del modello. In questo quadro, cognitività e complessità sono visti come indici della capacità dei modelli di esprimere la profondità della conoscenza.

3.1 Modelli Basati su Conoscenza A-Teoretica

In quest'ambito, il metodo classico è quello delle Reti Neurali Artificiali (ANN). Le reti neurali hanno un potenziale applicativo nella modellazione spaziale ed analisi di dati geografici, riconosciuto già da tempo [21]. Ciò per le loro proprietà di adattatività, non-linearità, robustezza, multidimensionalità. Le reti neurali sono state applicate soprattutto come metodo di classificazione, ruolo che assolvono attraverso una strategia di pattern recognition [20]. La strategia è tipicamente bottom-up, ovvero estrazione di conoscenza dal basso del dato geospaziale primitivo. L'attinenza con la cognitività deriva dalle note affinità che le reti neurali hanno con la fisiologia del cervello biologico e, per alcuni algoritmi, della loro capacità di realizzare delle mappe con significati topologici e semantici, sfruttando schemi insiti nell'informazione. Le reti *feed-forward* nella previsione territoriale servono a prevedere cambiamenti nell'uso del suolo, ad esempio la transizione da area coltivata a zona urbana. Questa previsione si fa su ampie regioni, sulla base di fattori guida quali la densità agricola, la distanza dalla rete autostradale o da specchi d'acqua, o da aree residenziali. Casi di studio si hanno in [24] e [9] dove il neuro-classificatore viene addestrato con i dati relativi alle trasformazioni dell'uso del terreno occorse su un periodo abbastanza lungo, nell'ordine di alcuni decenni, ed acquista la capacità di prevedere le trasformazioni nell'uso del terreno negli anni a venire. In [15], l'algoritmo SVM (Support Vector Machine) è applicato per prevedere se un territorio è soggetto ad espansione urbana in un prossimo futuro. Questo è fatto sulla base degli stessi fattori guida considerati da [24]. I risultati confrontati con quelli della rete neurale feed-forward, evidenziano la complementarità dei risultati: laddove SVM svolge una classificazione più accurata la rete neurale incorre invece in problemi di overfitting. In [12] viene applicato l'algoritmo SVM ad un problema di classificazione a due classi, in cui si vuol prevedere per una data concentrazione di una sostanza inquinante, quali punti della superficie di un lago risultano al di sopra o al di sotto di quel valore. Si tratta di un problema con caratteri di non linearità. Peraltro i risultati della SVM mostrano una performance comparabile con quella del kriging indicator, metodo geostatistico che produce mappe bivariate minimizzando la probabilità di errore.

Ma nell'ambito delle reti neurali la maggiore profondità di conoscenza è probabilmente espressa dalla tecnica *Self-Organizing Map* (SOM). Questa è stata applicata al clustering dei dati geospaziali da [20] e [2], evidenziando il potenziale

delle reti neurali come algoritmi non-supervisionati. In ambito GIS, [2] rileva la superiorità della SOM rispetto all'algoritmo k-means. Nella rete *GeoSOM*, di [1] e [3], tipologia di SOM specificamente adattata all'informazione geospaziale, si introduce sul clustering un parametro restrittivo: la tolleranza geografica. Questa rappresenta il raggio spaziale entro cui limitare la ricerca di oggetti simili. L'effetto della tolleranza geografica è quello di forzare unità che sono vicine nello spazio geometrico di input a restare tali in quello di output. La tolleranza geografica è fissata dall'operatore in funzione del grado di dipendenza spaziale che egli vuole attribuire al clustering. Questo algoritmo è stato utilizzato nella individuazione di zone omogenee sul territorio [3], nel tracciamento di confini geomorfologici e nella costruzione di regioni di Voronoi. Una versione tridimensionale, *Geo3DSOM*, è stata recentemente impiegata per la modellistica della distribuzione di elementi chimici nelle falde acquifere [23]. L'inserimento della terza dimensione riduce ulteriormente l'errore geografico ed aumenta quello di quantizzazione, il che peraltro si manifesta nello spazio di output con un maggior numero di clusters.

Nelle reti SOM la conoscenza profonda è prodotta dall'analisi che l'algoritmo fa di un contesto, che è semantico nella generalità dei SOM, o di prossimalità spaziale nel caso specifico della *GeoSOM*. Nel caso della *prossimalità spaziale* la conoscenza esprime le relazioni che sussistono sulla base dell'omonimo principio secondo cui punti spaziali adiacenti tendono a condizionarsi vicendevolmente nel valore dei loro attributi. Inoltre c'è da osservare che il concetto di prossimalità è multidimensionale, quindi non relegato soltanto alla dimensione spaziale. Ad esempio anche la capacità di esplorare il contesto temporale contribuisce ad accrescere le potenzialità del metodo in termini di profondità di conoscenza.

3.2 Modelli Basati su Conoscenza Teoretica

Si tratta dei modelli dotati di una base di conoscenza a-priori espressa in modo proposizionale, costituita da un insieme di dati geospaziali e relazioni esplicite tra questi. Tali modelli che, nell'ambito GIS, individuiamo negli Automi Cellulari (Cellular Automata - CA) ed Agenti Intelligenti (Intelligent Agents), sono basati sul concetto di *automa a stati finiti*. L'input per il passaggio di stato dell'automa è dato dal contesto ambientale, che può essere virtualmente rappresentato da altri automi. L'impiego degli *automi cellulari* nel GIS è descritto in sintesi da [4]. Tra i casi applicativi: in [30] il metodo degli automi cellulari è impiegato per la rappresentazione del sistema urbano e per predirne l'evoluzione dinamica a partire da uno stato arbitrariamente scelto. Lo spazio urbano è trasposto in un reticolo omogeneo e continuo di celle bidimensionali, quadrate, di diversi tipi in relazione all'uso, così abbiamo celle adibite a strade, industrie, verde pubblico, abitativi. Ogni cella è un automa, soggetta quindi a cambiamenti di stato nel tempo, sulla base di semplici regole che controllano l'influenza delle celle adiacenti. Viene rispettato strettamente il principio di prossimalità spaziale. Altri esempi si hanno in [33] e in [19] con lo studio della dinamica di una foresta equatoriale tramite gli automi cellulari, finalizzato ad evidenziare il condizionamento di fattori quali: la posizione dei villaggi, l'uso del suolo, lo sviluppo stradale,

la pianificazione territoriale. L'affidabilità di queste simulazioni dipende dalla qualità della base di conoscenza già a priori fornita al sistema. È maggiore il grado di difficoltà incontrato nella modellistica esplanatoria, dove l'obiettivo è la costruzione di un modello teorico.

Anche gli *agenti* [31], sono degli automi, ma a differenza degli automi cellulari, essi possono cambiare la posizione nello spazio, ed interagire tra loro anche a distanza [28] infrangendo così il principio di prossimalità, o meglio introducendo un altro tipo di prossimalità che considera ambiti dimensionali diversi da quello spaziale. Inoltre gli agenti possono incorporare funzionalità più o meno avanzate di ragionamento, ed essere capaci di pianificazione o di mentalismo, cioè belief, desire, intention, nonché di interagire dando luogo a dinamiche sociali [8] ed ai cosiddetti *Sistemi Multiagente* (Multi-Agent Systems MAS) [14]. La prima applicazione dei sistemi multiagente nel GIS è stata il progetto SIMPOP [7]. L'obiettivo del progetto è quello di definire il peso di diversi fattori nella crescita del centro urbano. La piattaforma SIMPOP è progettata per simulare l'insorgenza, la struttura e l'evoluzione di un sistema di città, partendo da una distribuzione iniziale di insediamenti in un'ampia regione, nazione o insieme di nazioni, e da una ripartizione delle risorse che può essere scelta in modo casuale, e di regole che definiscono come le città interagiscono, crescono e si specializzano. Il Progetto SIMPOP2 [6], è una ulteriore evoluzione in cui sono stati aggiunti nuovi fattori di competizione quali innovazione tecnologica e governativa. Il modello include attualmente diverse migliaia di agenti. La simulazione di ambienti facilmente discretizzabili ha portato ad una convergenza verso i paradigmi della programmazione ad oggetti. OBEUS è un sistema basato sulla teoria dei GAS (Geographic Automata Systems) proposta da [5], secondo cui i sistemi urbani consistono esclusivamente di oggetti, fissi o mobili: individui, proprietà, veicoli, edifici, appezzamenti di terreno. La ragione per cui i Sistemi Multiagente trovano applicazione prevalente nella simulazione di sistemi urbani è che questi sono caratterizzati da relazioni ben definite tra una quantità di oggetti reali, numerabili e dalle proprietà perfettamente note, come si verifica generalmente negli ambienti ad elevata antropizzazione.

Da quanto detto si può concludere che la scelta di una tecnologia dipende dalla qualità dell'informazione geospaziale a-priori disponibile, ovvero dal suo valore di profondità. Laddove gli automi cellulari esprimono relazioni puramente spaziali, mentre gli agenti, sebbene siano oggetti geo-spaziali, sono usati per rappresentare relazioni funzionali extraspaziali. Generalmente i sistemi multiagente hanno a disposizione già in partenza una base di conoscenza con forte carattere di astrazione, ovvero di profondità. Come conseguenza di ciò la realistica del comportamento emergente è diversa nei due casi, e sono considerati più attendibili i modelli ottenuti nei sistemi multiagente.

3.3 Modelli Ibridi

Le tecnologie basate su automi sono spesso applicate in modo combinato, così da poter avere i vantaggi di entrambe: gli automi cellulari sono adatti a gestire un'informazione più approssimata ma anche più vasta, invece gli agenti vanno

bene per modellare relazioni complesse tra un numero ridotto di elementi geografici. Un esempio applicativo si ha in [10], dove automi cellulari e tecnologia multiagente sono combinati per sviluppare un modello di come entità umane si muovono in un ambiente bi- o tri-dimensionale in funzione di vari fattori. Gli individui sono rappresentati da agenti, gli automi cellulari rappresentano l'ambiente. Un'altra integrazione sperimentata tra automi cellulari ed agenti intelligenti si ha in [6], dove i primi sono utilizzati per rappresentare fenomeni continui sul territorio, i secondi rappresentano fenomeni numerabili e delimitati da contorni fisici.

Altrettanto interessanti sono i casi di innesto di reti neurali negli automi. In [17] le reti neurali e gli automi cellulari sono combinati nell'ambito di un progetto di pianificazione territoriale per giungere ad una integrazione delle loro performances nel produrre conoscenza organizzata per il processo decisionale. In [33] si rilevano casi di applicazione di reti neurali per ottimizzare i processi di transizione di stato delle celle di un automa cellulare. Con il metodo del Connectionist Model Transfer (CMT)[25], è stato realizzato un sistema multiagente per l'elaborazione di fotografie aeree, in cui gli agenti si scambiano conoscenza in forma di reti neurali. Tra gli ibridi un sistema di rilievo nelle applicazioni ai fenomeni geografici è il PANN (Periodic ANN) applicato agli studi sulle piene idrografiche da [29]. Si ottiene separando i dati di addestramento secondo criteri stagionali, ovvero in subset specifici per ogni periodo dell'anno, e quindi addestrando una rete per ogni subset. Una rete al livello superiore integra i risultati delle altre. Si è realizzato così un modello composto di elevata accuratezza.

3.4 Quadro Comparativo delle Condizioni di Applicabilità

In questa sezione si mette in evidenza come ogni metodologia ha un ambito applicativo preferenziale, dettato dalla qualità della base di conoscenza.

Traendo delle conclusioni generali sulla applicabilità delle tecnologie, in riferimento alla Tabella 1, si può dire che: le reti neurali trovano applicazione nel pattern recognition, per la loro capacità di operare in condizioni di conoscenza superficiale, ciò sia in modo supervisionato, con gli algoritmi feed-forward e SVM, che in modo non supervisionato con le SOM. Gli Automi Cellulari invece trovano applicazione preferenziale nella modellistica predittiva su vasti territori, per la loro capacità di rappresentare i fenomeni areali e le relazioni di prossimalità spaziale, peraltro operano su una base di conoscenza di basso livello dove solo i rapporti spaziali tra le entità sono definiti. Infine i sistemi multiagente sono impiegati nella modellistica esplanatoria e problem solving, soprattutto nella simulazione di processi caratterizzati da entità e relazioni ben definite come si verifica negli ambienti ad elevata antropizzazione, qui la conoscenza a disposizione è del tipo nascosto.

Strategia	Problema	Applicazione	ANN		Automi		Disciplina
			Superv.	SOM	CA	MAS	
Pattern Recognition	Controllo del territorio	Riconoscim. oggetti da immagini aeree	■	■			Remote sensing
	Monitoraggio ambientale	Controllo indici di pressione ambientale		■			Ecologia
	Prospezioni minerarie	Individuazione giacimenti		■			Geologia
	Pianificazione territoriale	Previsione uso del suolo		■			Gestione territoriale
Previsione dissestabilità su vaste aree			■			Geotecnica	
Modellistica Predittiva	Valutazione del rischio ambientale	Previsione piene	■		■		Idrologia
	Protezione civile	Prev. propagazione incendi			■		Scienze forestali
	Interventi sul territorio	Evoluzione copertura vegetale			■		
Modellistica Esplanatoria	Supporto alla decisione	Analisi dei fattori di viabilità				■	Gestione urbana
		Analisi dei fattori di dinamica urbana				■	
		Analisi della dinamica migratoria				■	
		Analisi dei fattori economici				■	Micro-economia

Tabella 1. Quadro modelli/applicazioni

4 Verso una Definizione Delle Tecnologie Cognitive e di un Quadro Sistemico

La Tabella 2 sintetizza le proprietà delle tecnologie evidenziate nei casi di studio. La finalità è di individuare un comune denominatore per tutti i metodi visti.

In Tabella 2, in grassetto, sono evidenziate le proprietà comuni a tutte le tecnologie: adattatività, non-linearità, robustezza, multidimensionalità, emergenza, ed una complessiva capacità di astrazione semantica (* se sono inclusi appropriati motori inferenziali). Complessivamente queste proprietà comuni identificano le cosiddette *tecnologie cognitive*.

A questo punto si vuol avanzare una proposta per un approccio sistematico alla classificazione delle tecnologie cognitive in ambito GIS. Dalle conclusioni sin qui tratte, si ritiene che una distinzione tra le tecnologie sia possibile sulla base di aspetti qualitativi dell'informazione che esse gestiscono. Questi i parametri a nostro avviso maggiormente significativi:

Proprietà	Tecnologie Cognitive				
	ANN		Automi		
	Superv.	SOM	Automi Cellulari	Agenti Intelligenti	Sistemi Multiagente
Adattatività	■	■	■	■	■
Non-linearità	■	■	■	■	■
Robustezza	■	■	■	■	■
non-Sensibilità al Rumore	■	■			
Amm. dati non-etichettati	■	■			■
Multidimensionalità	■	■	■	■	■
Pianificazione				■	■
Mentalismo (intenzionalità)				■	■
Comportamenti Emergenti	■	■	■	■	■
Comportamenti sociali					■
Semantica/Topologia		■	*	*	*
Rappresentazione del contesto		■	■		
Include Procedure Euristiche				■	■

Tabella 2. Proprietà delle tecnologie cognitive

- Struttura dei dati in ingresso: Vettoriale, Raster, Oggetti
- Rispetto (o meno) del principio di prossimalità: ovvero la propensione (o meno) dell'informazione al rispetto del principio di prossimalità spaziale
- Mobilità delle entità geografiche: le entità possono essere relegate in una posizione o spostarsi
- Metodo di rappresentazione della conoscenza geografica: la rappresentazione della conoscenza può essere simbolica, connessionista o ibrida su distinti livelli.

La Tabella 3 riassume la distribuzione di questi parametri, desunta dai casi esaminati. Ancora una volta da un quadro complessivo si evince un profilo preciso per ogni tecnologia cognitiva, questa volta atto a differenziare una semantica del modello, che ne spieghi le proprietà ed il significato. Con riferimento alla Tabella 3, si possono definire le reti neurali come un metodo che si applica quando la conoscenza disponibile sul dato geospaziale è superficiale o bassa, mentre i sistemi basati sugli automi sono capaci di gestire una conoscenza più profonda ma di oggetti geografici discreti e senza una collocazione spaziale definitiva. Altresì è possibile scendere ad un ulteriore grado di dettaglio nei riguardi del parametro della prossimalità. Nella Tabella 4, riconsiderando il carattere multidimensionale, ovvero non solo spaziale, della prossimalità è possibile differenziare le tecnologie cognitive sulla base della dimensione su cui mostrano maggior competenza applicativa.

Lo schema della Tabella 4 concorda con quanto rilevato anche in soluzioni ibride, dove ad esempio l'associazione automi cellulari e agenti da luogo a modelli dove sono rappresentate le relazioni geospaziali più semplici e al tempo stesso re-

Parametro qualitativo	Tecnologie Cognitive			
	ANN		Automati	
	Superv.	SOM	Automati Cellulari	Sistemi Multiagente
Struttura dati	Vettoriale	Vettoriale	Reticolo spaziale omogeneo e continuo di oggetti	Distribuzione spaziale discontinua di oggetti eterogenei
Rispetto Principio di Prossimalità	no	si	si	no
Entità con mobilità spaziale	no	no	no	si
Rappresentazione della conoscenza	stato	stato	regole/stato	regole/stato
Profondità della base di conoscenza	superficiale	bassa	bassa	nascosta

Tabella 3. Caratterizzazione multi-parametrica delle tecnologie cognitive

	Tecnologie Cognitive			
	ANN		Automati	
	Superv.	SOM	Cellulari	Multiagente
Prossimalità spaziale		■	■	
Prossimalità temporale	■			
Prossimalità semantica		■		
Prossimalità funzionale				■
Pattern arbitrario	■			

Tabella 4. Caratterizzazione in base al parametro di prossimalità

lazioni funzionali che hanno luogo su un piano non spaziale, ad esempio relazioni economiche, sociali, normative tra le entità geografiche. Lo schema è utile in una fase studio di fattibilità del modello, per la scelta delle tecnologie da utilizzare.

5 Conclusioni

Dall'analisi di casi di studio si sono individuate delle tecnologie particolarmente efficaci nella rappresentazione dell'informazione geospaziale, nell'ambito dei sistemi GIS. Tali tecnologie mostrano delle proprietà comuni ed affini alle strategie cognitive, il che ha consentito di definirle tecnologie cognitive. È stato altresì proposto un quadro sistematico di queste tecnologie, basato su aspetti qualitativi dell'informazione gestita, prima fra tutte la profondità della conoscenza. Questa classificazione inquadra anche le soluzioni ibride. Il quadro complessivo fa ritenere che il riferimento a coordinate geografiche sia soltanto uno dei possibili ambiti dimensionali, il richiamo è va al tempo ed alla semantica dell'informazione geografica.

Riferimenti bibliografici

1. F. Bacao, V. Lobo, and M. Painho. Applications of Different Self-Organizing Map Variants to Geographical Information Science Problems. In P. Agarwal and A. Skupin, editor, *Self-Organising Maps: Applications in Geographic Information Science*. John Wiley and Sons, 2004.
2. F. Bacao, V. Lobo, and M. Painho. Self-organizing Maps as Substitutes for K-Means Clustering. In *Lecture Notes in Computer Science*, volume 3516, pages 476–483. Springer-Verlag, 2005.
3. F. Bacao, V. Lobo, and M. Painho. The Self-Organizing Map, the Geo-SOM, and relevant variants for geosciences. *Computers & Geosciences*, 31(2):155–163, 2005.
4. M. Batty. Geocomputation using cellular automata. In S. Openshaw and R. J. Abraham, editors, *Geocomputation*, pages 95–126. Taylor & Francis, 2000.
5. I. Benenson and P. M. Torrens. Geosimulation: object-based modeling of urban phenomena. *Computers, Environment and Urban Systems*, 28(1):1–8, 2004.
6. A. Bretagnolle, E. Daude, and D. Pumain. From theory to modelling : urban systems as complex systems. *CyberGeo*, (335), 2006.
7. S. Bura, F. Gurin-Pace, A. Mathian, D. D. Pumain, and L. Sanders. Exploring neural network rainfall-runoff modelling. *Geographical Analysis*, 2(27):161–178, 1996.
8. C. Castle and A. Crooks. Principles and concepts of agent-based modelling for developing geospatial simulations. CASA Working Paper 110, Centre for Advanced Spatial Analysis, University College London, 2006.
9. Erfu Dai, Shaohong Wu, Wenzhong Shi, Chui kwan Cheung, and Ahmed Shaker. Modeling Change-Pattern-Value Dynamics on Land Use: An Integrated GIS and Artificial Neural Networks Approach. *Journal of Environmental Management*, pages 576–591, 2005.
10. J. Dijkstra, H. J. P. Timmermans, and A. J. Jessurun. A Multi-Agent Cellular Automata System for visualizing simulated pedestrian activity. In S. Bandini and T. Worsch, editors, *Theoretical and Practical Issues on Cellular Automata*. Springer-Verlag, 2000.
11. M.J. Egenhofer and D.M. Mark. Naive Geography. In *Lecture Notes in Computer Science*, volume 988. Springer-Verlag, 1995.
12. N. Gilardi, M. Kanevski, M. Maignan, and E. Mayoraz. Exploring neural network rainfall-runoff modelling. In *Advanced Course on Artificial Intelligence*, pages 43–51, Crete, GR, 1999. European Coordinating Committee for Artificial Intelligence.
13. M. F. Goodchild, M. J. Egenhofer, K. K. Kemp, D. M. Mark, and E. S. Shepard. Introduction to the Varenius project. *International Journal of Geographical information Science*, 13(8):731–745, 1999.
14. N.R. Jennings, K. Sycara, and M. Wooldridge. A Roadmap of Agent Research and Development. *Autonomous Agents and Multi-Agent Systems*, 1(1):7–38, 1998.
15. A. Lazar and B. A. Schellito. Comparing Machine Learning Classification Schemes a GIS Approach. In *Fourth International Conference on Machine Learning and Applications*, pages 75–81, Los Angeles, USA, 2005. Institute of Electrical and Electronics Engineers.
16. E. Lepore and Z.W. Pylyshyn. *What is Cognitive Science?* Blackwell, 1999.
17. S. M. T. Lombardo and S. Pecori. Costruzione di sistemi cognitivi per la simulazione dell'evoluzione urbana: i L.U.C.A. In *Atti del Convegno INPUT 2001* , Isole Tremiti, I, 2001. Association Geographic Information Laboratories Europe.

18. D. M. Mark, C. Freksa, S. C. Hirtle, R. E. Lloyd, and B. Tversky. Cognitive models of geographical space. *International Journal of Geographical Information Science*, 13(8):747–774, 1999.
19. J. P. Messina and S. J. Walsh. Simulating land use and land cover dynamics in the Ecuadorian Amazon through cellular automata approaches and an integrated GIS. *Plant Ecology*, (156):75–88, 2001.
20. S. Openshaw and R. J. Abraham. *Geocomputation*. Taylor & Francis, 2000.
21. S. Openshaw and C. Openshaw. *Artificial Intelligence in Geography*. John Wiley and Sons, 1997.
22. M. Painho, A. Vasilakos, F. Bacao, and W. Pedrycz. Exploring spatial data through computational intelligence: a joint perspective. *Soft Computing - A Fusion of Foundations, Methodologies and Applications*, 2(5):326–331, 2005.
23. L. Peeters, F. Bacao, V. Lobo, and A. Dassargues. Exploratory data analysis and clustering of multivariate spatial hydrogeological data by means of GEO3DSOM. *Hydrology and Earth System Sciences Discussions*, 3(5):1487–1516, 2006.
24. B. C. Pijanowski, D. G. Brown, B. A. Schellito, and G. A. Manik. Using neural nets and GIS to forecast land use changes: a land transformation model. *Computers, Environment and Urban Systems*, 26(6):553–575, 2002.
25. S. Quirolgico, K. Canfield, T. Finin, and J. A. Smith. Communicating Neural Network Knowledge between Agents in a Simulated Aerial Reconnaissance System. In *First International Symposium on Agent Systems and Applications*, pages 242–254, Palm Springs, USA, 1999. Institute of Electrical and Electronics Engineers.
26. K. A. Richardson, P. Cilliers, and M. Lissack. Complexity Science: A Gray Science for the Stuff in Between. *Emergence*, 3(2):6–18, 2001.
27. R. J. Roiger and W. M. Geatz. *Data Mining: A Tutorial-based Primer*. Addison-Wesley, 2002.
28. P. M. Torrens and I. Benenson. Geographic automata systems. *International Journal of Geographical Information Science*, 19:385–412, 2005.
29. Wen Wang, P. H. A. J. M. Van Gelder, J.K. Vrijling, and Jun Ma. Forecasting daily streamflow using hybrid ANN models. *Journal of Hydrology*, 324:383–399, 2005.
30. R. White. Cities and Cellular Automata. *Discrete Dynamics in Nature and Society*, 2(2):111–125, 1998.
31. M. Wooldridge and N. R. Jennings. Intelligent Agents: Theory and Practice. *The Knowledge Engineering Review*, 19(2):1–62, 1995.
32. M. F. Worboys and M. Duckham. *GIS: A Computing Perspective*. CRC Press, 2004.
33. A. G. O. Yeh and Xia Li. Urban simulation using neural networks and cellular automata for land use planning. In *Ottawa Symposium on GeoSpatial Theory*, Ottawa, CND, 2002.

ChronoGeoGraph: an Expressive Spatio-Temporal Conceptual Model

Donatella Gubiani¹ and Angelo Montanari²

¹ Dipartimento di Scienze, Università G. D'Annunzio di Chieti-Pescara, Italia

² Dipartimento di Matematica e Informatica, Università di Udine, Italia

Abstract. The problem of dealing with spatio-temporal information is widely recognized as a relevant one in the database and information system community. We focus our attention on the conceptual design of spatio-temporal systems. We first survey existing conceptual spatio-temporal data models and then we propose a new model, called ChronoGeoGraph (CGG). CGG encompasses multiple temporal dimensions and it supports both the object-based and the field-based view of spatial information. Furthermore, it makes it possible to describe the temporal evolution of spatial information. The paper provides a detailed account of the main CGG constructs; moreover, it presents the basic features of a software tool for the visual synthesis of CGG schemas and their mapping into either relational schemas or XML Schema documents.

1 Introduction

Conceptual modeling is a crucial phase in the development of spatio-temporal database and information systems. Spatial and temporal requirements are indeed much more difficult to deal with than the usual requirements of conventional systems. In this paper we describe the main features of an original spatio-temporal conceptual model, called ChronoGeoGraph (CGG), that extends the classical Enhanced Entity-Relationship (EER) model with constructs for representing spatio-temporal information (the CGG model extensively revises and temporally extends a spatial model originally proposed in [DeF05]).

Conventional database and information systems only maintain information about the current state of the world. Temporal databases can be viewed as a systematic attempt to overcome this limitation, making it possible to keep track of the evolution of the modeled domain as well as of the database contents. A variety of temporal databases has been proposed in the literature to model one or more temporal dimensions of data. Two temporal dimensions have been commonly recognized as fundamental, namely, valid and transaction time. The valid time of a fact is the time when the fact is true in the modeled domain, while the transaction time of a fact is the time when the fact is current in the database and may be retrieved. In [Com01], two additional temporal dimensions have been proposed, namely, event time and availability time, to remedy some weaknesses of valid and transition times. The event time of a fact consists of the occurrence times of the pair of real-world events that respectively initiate and

terminate its validity interval, while the availability time of a fact is the time interval during which the fact is known and believed correct by the information system the database belongs to (in general, such an interval does not coincide with the transaction time interval of the fact). Finally, an existence time has often been associated with objects to model their life span, e.g., [Try99]. As for spatial information, the modeling language must provide the designer with suitable conceptual tools to manage complex multidimensional data with a number of spatial features, such as shape, extension, and location. In particular, it must support different types of spatial relation, including topological, metric, and direction ones. Moreover, it must cope with the incompleteness and/or inaccuracy of spatial data and it should allow multiple representations of the same spatial data, possibly at different representation scales. In addition, besides the standard object-based representation, data models should support the alternative field-based representation of spatial information. According to the object-based view, spatial information about the world is conveyed by a set of distinct entities, each one characterized by a position, a geometric shape (e.g., point, line, polygon), and other spatial features, while the field-based view describes the space as a continuous surface over which some relevant spatial features, e.g., temperature and altitude, may change in a possibly continuous way.

CGG pairs the standard constructs of the EER model with a number of additional features for spatial and temporal information management. In particular, it encompasses all the above-mentioned temporal dimensions and it supports both the object-based and the field-based view of spatial information. Furthermore, it provides suitable constructs to constrain the temporal evolution of the geometrical properties of the modeled entities/phenomena. A set of tools for the management of CGG models is available. It includes a Java module for the visual synthesis of CGG conceptual schemas as well as two translation modules that map the generated schemas into spatio-temporal relational schemas, based on a temporal extension of the Oracle Spatial model, and XML Schema documents, respectively. The rest of the paper is organized as follows. Section 2 surveys existing conceptual spatio and/or temporal models. Section 3 describes CGG constructs. Section 4 briefly presents the set of available tools for CGG. Conclusions provide an assessment of the work and outline future research directions.

2 Related work

There exist many conceptual models for the management of temporal information or spatial information, but only a few of them deal with spatio-temporal information in an integrated way. Since there is no room for a detailed analysis of purely temporal or spatial models, we simply list the most important ones, pointing out their distinctive features, and we focus our attention on the few existing spatio-temporal models.

Various conceptual models for spatial information only support the object-based view (e.g., CONGOO, GeoER, and GeoUML). Both views are supported by GISER, OMT-G, GeoOMT, and GeoFrame. Spatial conceptual models also

differ in the way in which they represent spatial information. Some models (CONGOO) associate spatial information with relations; other models (GeoER and OMT-G) add a spatial dimension to both objects/entities and relations; other ones (GeoUML) associate spatial information with both attributes and relations. Temporal conceptual models mainly differ in the temporal dimensions they support. Most of them incorporate valid time and/or life span (TERM, RAKE, MOTAR, TEER, ITDM, STEER, ERT, TER, TUER, TERC+, and Chrono). Other models (TempEER, TimeER, TUML, and TEERM) also support transaction time. With a very few exceptions (TFBM features a decision time dimension, RAKE and XCM model events, and TERC+ includes synchronization/dynamic relations), no additional temporal dimensions are incorporated. Temporal information can be dealt with explicitly, as in MOTAR, ERT, TERC+, TimeER, Chrono, TUML, and TFBM, or implicitly, as in RAKE, TEER, ITDM, STEER, TER, TUER, TempEER, and TEERM. Moreover, as in the case of spatial models, temporal models differ in the way in which they represent temporal features. Some models (TEER) associate temporal information with objects/entities, attributes, and relations; other models (TimeER) add a temporal dimension to objects/entities and attributes; other ones (TER and TFBM) associate temporal information with relations only. Finally, while all spatial models are provided with an icon-based graphical counterpart, temporal models differ in the way in which they graphically depict temporal information. Some models (TERC+ and TempEER) are provided with a graphical annotation, while others (RAKE and ERT) are provided with a textual annotation only (TERM is devoid of a graphical notation).

Compared with the large number of existing spatial and temporal conceptual models, there exists a relatively little number of spatio-temporal models. Some of them are temporal extensions of spatial models, e.g., GeoFrame-T [VEI01], which extends the GeoFrame model [LI99], and the Perceptory model [Bed04], which is a spatio-temporal extension of UML derived from the MODUL-R model [Car93]; others are simple generalizations of existing development methods, e.g., GeoOOA [Kos97]. The most significant models can be partitioned on the basis of the formalism they originated from: STER and ST USM are based on EER, MADS is an object-oriented model, and STUML is based on UML. STER adds a spatial dimension to entities and relations (the set of spatial relations includes topological, direction and metric relations) [Try99]. Moreover, it includes attributes whose value may vary over space (spatial attributes) and it supports valid time (on attributes and relations), existence time (on entities), and transaction time. It also allows one to combine spatial and temporal features. As for STER diagrams, spatial and temporal information is incorporated into them by a suitable annotation of entity, attribute, and relation graphical representation. ST USM extends the EER-based USM model with spatio-temporal information [Kha01]. Spatial, temporal and spatio-temporal features can be added to entities, attributes, and relations. Spatial and temporal features respectively define their geometry (point, line, or polygon) and their temporal dimensions (valid time, existence time, and transaction time). Moreover, a granularity can be associated with both of them. Spatio-temporal features identify the spatial features that

may change over time, e.g., the position of an entity. MADS introduces spatial abstract data types that can be associated with objects and attributes [PSZ06]. The geometry of an object admits multiple representations; moreover there exist attributes that take their value on a spatial domain (spatial attributes) and attributes that change their value over space (space-varying attributes). In addition, MADS allows one to represent topological relations between objects and spatial aggregations of objects. Temporality can be associated with attributes (temporal and time-varying attributes) as well as with objects and relations (by specifying their life cycle). Moreover, MADS allows one to define synchronization, transition, and generation relations. Finally, MADS provides spatial and temporal features with an iconic representations. STUML is an UML-based model for requirement analysis and conceptual design of spatio-temporal applications [Pri01]. Spatio-temporal information is modeled by means of five distinct qualifiers, namely, spatial, temporal, thematic, group (to group attributes with common spatio-temporal properties), and existence-dependent (to identify attributes and associations that depend on object existence). These qualifiers can be combined and applied to object classes, attributes, and associations to model spatial extents, to represented existence time, valid time and/or transaction time, and to encode three different types of spatio-temporal data: temporal changes in spatial extents, changes in the values of thematic data over time or space, and composite data whose components may vary over time or space. In addition, a specification box can be used to describe the semantics of spatio-temporal data. Other spatio-temporal models have been proposed to cope with the requirements of specific applications. This is the case, for instance, with the Spatio-Temporal Environmental Data (STED) model [Ras03].

Conceptual models may differ in their expressive power, simplicity, orthogonality (the ability of expressing basic, spatial, and temporal features in an independent way), upward-compatibility (the ability of rendering conventional conceptual schemas spatio-temporal without affecting legacy schemas), snapshot reducibility (the ability of producing a conventional schema, with the traditional snapshot semantics, by removing spatio-temporal features), and granularity (the ability of representing data at different levels of spatial, temporal, or semantic detail). MADS, STUML, and ST USM stand out from the above set of spatio-temporal models in various respects. These three models are more expressive and orthogonal than the other two models (e.g., they provide advanced constructs for aggregation). Simplicity is achieved by all models. However, while STER, ST USM, and STED use a textual notation for spatio-temporal information (annotations in STER and ST USM, types in STED), MADS and STUML feature a graphical one. STER, ST USM, and STUML are upward-compatible (MADS is not an extension of a conventional model). All models but STED guarantee snapshot reducibility and all models but STER provide a granularity support (as a matter of fact, ST USM provides the richest support, while STUML does not manage all granularity dimensions). As for spatial information, all models directly support the object-based view, while the field-based view is indirectly managed by space-varying attributes in MADS, by spatial attributes in STER and ST USM, and by space-dependent thematic attributes in STUML. The

way in which spatial information is associated with the various constructs (object/entity, attribute, relation/association) differs from one model to the other. MADS and STUML support both spatial and space-varying attributes, while STER and ST USM only support space-varying attributes. MADS and STUML do not impose spatial dependencies between spatial entities and their spatial attributes, but allow one to specify them. All models support topological relations. MADS, STUML, and ST USM take advantage of a set of pre-defined relation types, while topological relations must be explicitly specified in STER. In addition, STUML and ST USM allow one to define space-varying relations. These relations are not necessarily spatial relations, but at least one of the participating entities must be a spatial entity. Spatial aggregation is available in MADS, STUML, and ST USM, but not in STER. As for temporal information, MADS only supports valid time (life span of entities and relations), while the other models support two temporal dimensions: STUML and ST USM support valid/existence time and transaction time, STED supports valid/existence time and event time. The models differ in the way they constrain temporal elements. STUML does not explicitly introduce temporal constraints, but it allows one to associate temporal conditions on object existence with attributes and relations. STER and ST USM impose existence dependencies. In particular, an attribute/relation cannot exist if its owner/participant entities do not exist. Finally, synchronization relations are supported by MADS and ST USM.

3 The ChronoGeoGraph model

ChronoGeoGraph (CGG) is a spatio-temporal model that pairs the classical features of the EER model with a large set of spatial and temporal constructs. On the one hand it retains the simplicity of the EER; on the other hand, it allows one to express relevant properties of spatial objects over the bidimensional space and to model their behavior with respect to four distinct temporal dimensions. CGG is upward compatible and snapshot reducible. Furthermore, its temporal and spatial components are fully orthogonal. Finally, it models spatial, temporal, and semantic granularity features. As for spatial information, CGG explicitly supports both the object-based and the field-based view. Moreover, it includes spatial entities, spatial attributes, various types of spatial relation, and cartographic specialization. It also provides a notion of schema territory, that identifies the spatial domain all spatial elements of the schema belong to. As in MADS and STUML, all spatial constructs have a graphical counterpart. As for temporal information, CGG supports existence/valid time, transaction time, event time, and availability time. Existence time can be paired with a state diagram. The last three dimensions (syntax T/E/A) can be associated with every constructs; moreover, existence time (syntax LS) can be associated with entities, while valid time (syntax V) can be associated with all the other constructs. Existence/valid, transaction, and availability times are represented as intervals, while event time is represented as a pair of points. Besides temporal dimensions, CGG introduces events (to model domain changes), temporal col-

lections (to model synchronization of attributes), and synchronization relations (to model temporal constraints on entities). As in STER and ST USM, temporal dimensions are encoded by means of textual annotations. The indication of the granularity of a time point or of an interval follows the temporal dimension it refers to and it is included within brackets.

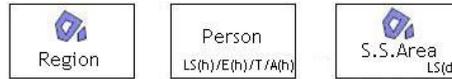


Fig. 1. Spatial (left), temporal (middle) and spatio-temporal (right) entities.

Entities. Besides basic entities, with internal or external identifiers, CGG supports spatial, temporal, and spatio-temporal entities (their graphical counterparts are given in Figure 1). A *spatial entity* is characterized by a set of descriptive attributes plus a geometry of a given spatial data type. The geometry defines the shape and location of the spatial entity and it may be part of the entity identifier. A spatial entity devoid of descriptive attributes is called a *purely spatial entity* and it is uniquely identified by its geometry. CGG supports 8 different spatial data types, which are graphically depicted in Figure 2. Besides the point, line, and polygon spatial types, it introduces the multipoint, multiline, multipolygon, and collection (multipoint, multiline, and/or multipolygon) types. Moreover, it adds an “unknown geometry” type to capture spatial entities with an unknown/different geometry. A *temporal entity* is an entity provided with one or more temporal dimensions, or with one or more time-varying attributes, or that participates in a time-varying relation. As in STER, STUML, and ST USM, existence and transaction time dimensions can be associated with a temporal entity. As in MADS, a state diagram can be associated with the existence time of an entity to define the set of its possible states and the rules that determine state changes. Moreover, CGG supports event time and availability time dimensions. *Spatio-temporal entities* are obtained by merging spatial and temporal features.



Fig. 2. The spatial data types supported by the CGG model.

Attributes and temporal collections. CGG borrows from EER the basic set of attribute types (simple, composite, optional, multivalued, and derived) and the possibility of constraining their cardinality. In addition, as in ST USM and MADS, it introduces *spatial attributes* that take their value over a spatial data type and can be associated with spatial and non-spatial entities (as an example, the non-spatial entity **person** may have a spatial attribute **residence**, with a point spatial data type). They do not model the spatial extension and location of spatial entities, which are encoded by the geometry of the entity. CGG does not impose any spatial constraint between the geometry of a spatial entity and the values of its spatial attributes (if any). Such constraints can possibly be modeled by replacing the spatial attribute with a purely spatial entity related to the given

entity by a suitable spatial relation. Besides spatial attributes, CGG introduces *temporal (non-key) attributes*. They are characterized by one or more temporal dimensions among valid, transaction, event, and availability times. Temporal dimensions can also be associated with the geometry of a spatial entity to model its changes over time (time-varying spatial entities). CGG distinguishes between snapshot and lifespan *cardinality constraints* on temporal attributes. *Snapshot* cardinality constraints specify the minimum and maximum number of values that the temporal attribute can take at *a given time*, while *lifespan* cardinality constraints specify these bounds with respect to the whole existence of the entity instance³. CGG does not impose temporal constraints between the valid time of a temporal attribute and the existence time of the owner entity. In particular, unlike other models, it does not constrain the validity interval of an attribute to be included in the existence interval of the owner entity. As an example, we may associate an attribute `numberOfSons` with an entity `person` whose value still holds, and may possibly change, after the death of the person. Temporal constraints can be explicitly added, whenever necessary. Finally, CGG allows one to collect sets of attributes that change in a synchronous way in a temporal collection, analogously to the group construct of STUML. A *temporal collection* is a set of attributes with a common temporal annotation. As a matter of fact, such a collection may include the geometry and/or the state of the entity as well some relations it participates in. Figure 3 represents the spatio-temporal entity ASL with a time-varying geometry and a temporal attribute `manager`.

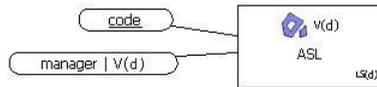


Fig. 3. An example of a spatio-temporal entity.

Relations. Besides basic relations, CGG features spatial, temporal, and synchronization relations. *Spatial relations* are relations among spatial entities. As MADS, CGG models topological, metric, and direction relations; moreover, it features a relation of spatial aggregation as MADS, STUML, and ST USM. *Topological relations* constrain the relationship between the geometries of pairs of spatial entities. They are preserved under translation, rotation, and scaling. They are graphically depicted as shown in Figure 4. A set of integrity constraints, which imposes suitable restrictions to the geometric types of the participating entities, is associated with every topological relation. As an example, they constrain the relation of inclusion to possibly hold between a point and a polygon, but not vice versa (the relation edges are labeled with the roles of participating entities). *Metric relations* specify distance constraints between pairs of spatial entities, e.g., “there must be an hospital less than 20 km far from every school”, while *direction relations* constrain the relative position of pairs of spatial entities, e.g., “there must be a control point North of every contaminated site”).

³ Apart from MADS, existing spatio-temporal models do not support such a distinction (it is present in some purely temporal models like TERC+, TER, and TimeER).

Spatial relations can be combined to deal with complex situations, e.g., “there must be a control point North of every contaminated site at a distance less than 5 Km”.



Fig. 4. Topological relations supported by CGG.

The relation of *spatial aggregation* is the composition relation over spatial entities. As in the case of topological relations, suitable restrictions are imposed to the geometric types of the participating entities. As an example, the types of the component entities of a compound entity of type multipoint can be of type point or multipoint only. In addition, CGG supports a weakened form of spatial aggregation (called semantic aggregation in [DeF05]) which allows one to relax spatial constraints. An example of a topological relation is given in Figure 5, while a spatial aggregation is reported in Figure 7.



Fig. 5. An example of a topological relation.

A *temporal relation* is a relation provided with one or more temporal dimensions or that has one or more time-varying attributes (since key attributes cannot be temporal, identifying relations cannot be temporal). Temporal relations are characterized by one or more temporal dimensions among valid time, transaction time, event time, and availability time. As in the case of temporal attributes, CGG distinguishes between snapshot and lifespan *cardinality constraints* on temporal relations. *Snapshot* cardinality constraints specify the minimum and maximum number of instances of a given entity that may participate in a relation at a *given time*, while *lifespan* cardinality constraints specify such numbers with respect to the whole validity interval of the relation. Again, CGG does not impose temporal constraints between the valid time of a temporal relation and the existence times of the participating entities. In Figure 6, we provide an example of a temporal relation.

Synchronization relations are binary relations between temporal entities, provided with one or more temporal dimensions, that impose temporal constraints on their existence intervals (as in MADS). CGG provides a set of synchronization relations that capture the standard interval/interval and point/interval relationships (MADS confines itself to interval/interval relationships). As spatial relations, synchronization relations have an intuitive graphical counterpart.

Specialization and cartographic specialization. CGG borrows from EER the relation of specialization. As usual, it can be disjoint or overlapping as well as total or partial. In addition, CGG distinguishes between static (the specialization of the entity person into man and woman) and dynamic (the specialization of person into person under age and person who has come of age) specializations.

Child entities inherit attributes and relations associated with the parent entity as well as its geometry (if any). Temporal dimensions are inherited in static specializations, but not in dynamic ones (CGG only constrains the existence time of child entities to be included in the existence time of the parent entity).



Fig. 6. An example of a temporal relation.

The relation of *cartographic specialization* allows one to model different representations of the same spatial entity (as in OMT-G). CGG distinguishes two different types of cartographic specialization: (i) with shape variation, when different geometries (either of the same or of different spatial data type) are associated with the same entity at a given scale, and (ii) with scale variation, when different geometries are associated with the same entity at different scales. The two kinds of specialization can obviously be combined: one may first introduce the scales of interest and then associate different geometries with them.

Territory schema. CGG has a notion of territory schema (MADS has a similar notion, but devoid of a graphical counterpart). A territory schema can be viewed as a degenerate purely spatial entity which admits one instance only. As a general rule, all spatial elements of a schema instance must be included in the instance of the territory schema. More restrictive conditions can be explicitly forced by using CGG spatial relations. As an example, in Figure 7 the instance of the territory schema *Italy* is defined as a spatial aggregation of instances of the entity *Region*. In addition, the territory schema can be temporal. CGG allows one to associate with the territory schema the same temporal dimensions one can associate with temporal attributes.

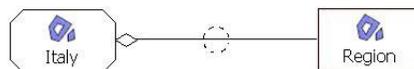


Fig. 7. An example of a territory schema involved in a spatial aggregation.

Fields. CGG models the field-based view of spatial information by the notion of (spatial) field. A field is a feature that varies over space. It can be associated with either the whole schema territory or a single spatial entity, and it is characterized by a specific sampling type (regular or irregular points, isolines, regular or irregular cells, TIN). Spatial interpolation functions over fields can be defined by means of derived attributes of the associated entity/schema territory. The temporal dimensions that can be associated with temporal attributes can also be associated with fields to model the history of sampling values.

Events. CGG explicitly keeps track of the events that change the state of a relevant element, e.g., the geometry of an entity, the validity of a relation, or the value of an attribute. Additional information about the nature (an initiating and/or a terminating event) and the effects (e.g., the event changes the state of

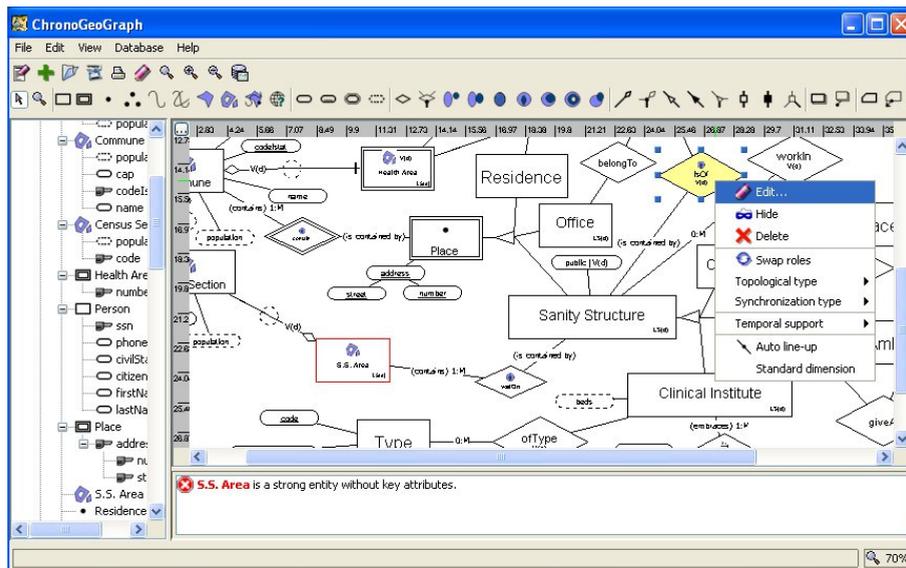


Fig. 8. The interface of the module for the visual synthesis.

the entity from active to dead) of an event can be associated with the edge that connects the event to the element on which it acts. There is not a corresponding notion in existing spatio-temporal models.

4 The CGG development framework

We conclude the paper with a short description of the available tools for the development, transformation, and validation of CGG schemas [Gub07a,Gub07b].

A module for the visual synthesis of CGG schemas. The first tool is a JAVA module for the visual synthesis of CGG schemas. Its interface, depicted in Figure 8, consists of the following elements: (i) a *menu bar* that provides the standard primitives for file manipulation and visualization as well as for the management of the interface, plus some functionalities to link the phase of conceptual design to the subsequent phase of logical design; (ii) *tool bars* that allow the designer to easily access the main primitives of the menu and to select the appropriate CGG construct to insert into the schema under development; (iii) a *central panel* which consists of three distinct parts, namely, an area where the *conceptual schema* can be drawn (up right), a *tree structure* that collects the elements of the current schema in a homogeneous way (left), and an area where the system records the *set of constraints* that are violated by the current version of the schema (down right); (iv) an *on-line assistant*, at the bottom of the interface, that helps the designer in the choice of the next element to insert. CGG constructs can be added to the current schema by selecting them

from the appropriate tool bar and following the instructions provided by the on-line assistant. Once inserted, a construct can be possibly modified by taking advantage of contextual menus (easy case) or by using specific editors (one for every construct). A distinctive feature of the module, that differentiates it from many existing tools for the synthesis of conceptual schemas, is its ability to automatically deal with integrity constraints. On the one hand, some integrity constraints are imposed by construction; on the other hand, the module allows the user to (temporarily) violate some integrity constraints. When a violation occurs, however, the module graphically identifies the involved elements of the schema, it gives a textual account of the violation, and it provides the designer with some hints on how to possibly solve the problem.

Translation and validation modules. The development framework also includes two translation modules that map the generated CGG schemas into two alternative spatio-temporal logical schemas. The first one is based on a temporal extension of the Oracle Spatial model. The resulting relational schema supports the object-based and the field-based views as well as the timestamping for all CGG temporal dimensions. In addition, it explicitly distinguishes between the current and the historical components of the schema. All the semantic constraints of the original CGG schema are encoded in terms of relational constraints and triggers. The second module generates XML Schema documents that exploit the main features of XML Schema, including elements, attributes, types, and constraints. Moreover, it takes advantage of the standard GML for the management of spatial data. Unfortunately, XML Schema cannot directly encode and check all spatio-temporal constraints of the original CGG schema (the elements that it cannot directly deal with are simply annotated with the tag `appinfo`). To overcome this limitation, we developed a spatio-temporal validation library that allows one to check whether a specific instance conforms a given schema as well as to check structural spatio-temporal properties of CGG schemas, e.g., the existence of cycles with some specific characteristics or the validity of some spatial and/or temporal relations between certain pairs of entities.

5 Conclusions and future work

In this paper we presented an original spatio-temporal conceptual model, called ChronoGeoGraph (CGG), which improves existing ones in several respects (expressiveness, orthogonality, simplicity, richness of the computational support). We also briefly described a set of accompanying software tools for the development, transformation, and validation of CGG schemas. The model has been experimented on a number of case studies (the examples we used to describe CGG constructs are taken from a medical case study). We are currently working at a logical formalization of CGG semantics based on a spatio-temporal extension of Description Logic. Moreover, we are systematically evaluating relative advantages and disadvantages of the relational and XML Schema translations.

Acknowledgements

The work has been funded by the Italian PRIN project: “Constraints and preferences as a unifying formalism for system analysis and solution of real-life problems”.

References

- [Bed04] Y. Bedard, S. Larrivee, M.J. Proulx, and M. Nadeau. Modeling Geospatial Databases with Plug-Ins for Visual Languages: A Pragmatic Approach and the Impacts of 16 Years of Research and Experimentations on Perceptory. In S. Wang et al. (Eds.), *Conceptual Modeling for Geographic Information Systems Workshop ER 2004*, pp. 17–30, 2004.
- [Car93] C. Caron and Y. Bedard. Extending the individual formalism for a more complete modeling of urban spatially referenced data. *Computers, Environment and Urban Systems*, vol. 17(4): 337–346, 1993.
- [Com01] C. Combi and A. Montanari. Data Models with Multiple Temporal Dimensions: Completing the Picture. In *Proc. of the 13th Conference on Advanced Information Systems Engineering*, LNCS 2068, Springer, pp. 187–202, 2001.
- [DeF05] I. De Fent, D. Gubiani and A. Montanari. Granular GeoGraph: a Multi-Granular Conceptual Model for Spatial Data. In *Proc. of the 13th Italian Symposium on Advanced Database Systems*, pp. 368–379, 2005.
- [Gub07a] D. Gubiani and A. Montanari. ChronoGeoGraph: a Development Framework for Spatio-Temporal Databases. *Research Report UDMI/01/07/RR*, University of Udine, 2007.
- [Gub07b] D. Gubiani and A. Montanari. A Tool for the Visual Synthesis and the Logical Translation of Spatio-Temporal Conceptual Schemas. In *Proc. of the 15th Italian Symposium on Advanced Database Systems*, Torre Canne (Fasano, BR), Italy, 2007.
- [Had96] T. Hadzilacos and N. Tryfona. Logical data modeling for geographic applications. *Int. Journal of Geographical Information Systems*, vol. 10(2): 179–203, 1996.
- [Kha01] V. Khatri, S. Ram and T. Snodgrass. ST USM: Bridging the Semantic Gap with a Spatio-Temporal Conceptual Model, *TIMECENTER Technical Report TR-64*, 2001.
- [Kos97] G. Kösters, B. U. Pagel and H. W. Six. GIS-application development with GeoOOA. *Int. Journal of Geographical Information Science*, 11(4): 307–335, 1997.
- [LI99] F. J. Lisboa and C. Iochpe. Specifying analysis patterns for geographic databases on the basis of a conceptual framework. In *Proc. of the 7th ACM Symposium on Advances in Geographic Information Systems*, pp. 7–13, 1999.
- [PSZ06] C. Parent, S. Spaccapietra and E. Zimányi. *Conceptual modeling for Traditional and Spatio-Temporal Applications: The MADs Approach*, Springer, 2006.
- [Pri01] R. Price. Conceptual Modeling Techniques for Spatiotemporal Applications. *PhD Thesis*, Monash University, Australia, 2001.
- [Ras03] J. Rasinmäki. Modelling Spatio-Temporal Environmental Data. *Environmental Modelling and Software*, 18(10): 877–886, 2003.
- [Try99] N. Tryfona and C. Jensen. Conceptual Data Modeling for Spatiotemporal Applications. *Geoinformatica*, 3(3): 245–268, 1999.
- [VEI01] L. Vargas da Rocha, N. Edelweiss and C. Iochpe. GeoFrame-T: A Temporal Conceptual Framework for Data Modeling. In *Proc. of the 9th ACM Symposium on Advances in Geographic Information Systems*, pp. 124–129, 2001.

Accurate and Fast Similarity Detection in Time Series

Francesco Gullo, Giovanni Ponti, Andrea Tagarelli, Sergio Greco

DEIS, University of Calabria

e-mail: {fgullo,gponti,tagarelli,greco}@deis.unical.it

Abstract. This paper addresses the problem of similarity detection in time series from both an effectiveness and efficiency viewpoint. A main motivation underlying our work is that, as we experimentally proved, no state-of-the-art method has capabilities for both fast and accurate similarity detection in time series. Viewed in this respect, we propose *DSA (Derivative time series Segment Approximation)*, a representation model for time series that suitably combines the notions of derivative estimation, segmentation and segment approximation to support effective and efficient similarity detection. Experiments conducted in a hierarchical clustering framework show that DSA based similarity detection is as good or better than both the most accurate and the fastest methods among the competing ones.

1 Introduction

A time series, or time sequence, T is a list of (real) numeric values upon which a total order based on timestamps is defined. The traditional form $T = [(x_1, t_1), \dots, (x_n, t_n)]$ can be rewritten as $T = [x_1, \dots, x_n]$ when, as usual, a fixed sampling period is assumed. Significant amounts of time series data are naturally available on several sources of different domains, such as speech recognition, biomedical measurement, financial and market data analysis, telecommunication and telemetry, sensor networking, motion tracking, and meteorology.

In recent years, all of these application domains have raised the demand for suitable solutions to the problem of identifying similarities among time series data. Addressing such a problem is significant for different tasks, such as indexing and query processing, change detection, rule discovery, and classification/clustering. In this context, the basic approach is dynamic time warping, and the most relevant methods are its extensions, possibly including techniques borrowed from string matching based on edit distance. Also, high dimensionality of time series has raised the demand for dimensionality reduction techniques to improve the efficiency of similarity searches.

The main contribution of this paper is twofold. At a first stage, we reviewed the state-of-the-art in time series data management focusing on existing solutions for two major issues, namely *similarity detection* and *dimensionality reduction*. Our empirical study pointed out that no existing technique can be used to perform high accuracy similarity detection while maintaining low the computational

effort. This finding led us to devise a representation scheme for time series that is able to support both effective and efficient similarity detection.

Within this view, we propose *DSA – Derivative time series Segment Approximation*, a time series representation model based on an original combination of the notions of derivative estimation, segmentation and segment approximation. DSA allows for modelling time series into a concise yet feature-rich representation, thus enabling fast and accurate time series matching and similarity detection.

Experimental results conducted in a hierarchical clustering framework show that DSA-based time series similarity detection is as good or better than both the most accurate and the fastest methods among the competing ones, thus guaranteeing the best trade-off between effectiveness and efficiency.

2 Related Work

2.1 Similarity detection

Similarity detection in time series in principle should meet the following requirements: handling local time shifting, high efficiency in computation, low sensitivity to noise, and support for indexing. The Euclidean distance (L_2), initially used in [1], is fast to compute and is a metric, but it is unable to deal with noisy sequences and sequences with different lengths or shifted in the time axis. Superior approaches are based on warping the time axis and on string matching measures.

Warping the time axis allows to achieve the best alignment between data points of two time series. The Dynamic Time Warping (DTW) algorithm has long been known in speech recognition [2], then was introduced to the data mining community as an effective solution to the sensitivity of the Euclidean distance to small distortions (i.e. fluctuations or phase shifts) in the time axis [3]. Given two sequences T_1 and T_2 , DTW performs a non-linear mapping of one sequence to another by minimizing the total distance between them. Initially, a $(|T_1| \times |T_2|)$ -matrix is built to contain the squared Euclidean distances between T_1 's points and T_2 's points. To find the best alignment between the two sequences, a warping path (i.e. a sequence of matrix elements) is computed in such a way that: it starts and ends in diagonally opposite corner cells of the matrix, all elements in the path are contiguous and monotonically spaced in time, and the total cumulative distance is minimized. The optimal path is retrieved by using a dynamic programming algorithm, whose complexity is $\mathcal{O}(|T_1| \times |T_2|)$.

DTW can handle time series with local time shifting and different lengths, although it is not a metric, unlike the Euclidean distance. Pruning techniques proposing computationally cheap lower bounds (e.g. [4–7]) have been defined to make DTW able to support indexing. In particular, the lower bounding measure proposed in [4] has been extensively used in several application contexts (e.g. [8–12]). This measure uses a bounding envelope that encloses one of two series being compared and is defined by an upper bound sequence U and a lower bound sequence L . In general, such bound sequences are defined depending on the specific

domain. A similar lower bounding measure, named FTW (Fast search method for dynamic Time Warping), has been recently proposed in [7]. FTW approximates DTW for purposes of query processing (i.e. efficient k-nearest neighbor and range queries) and leverages the inability of exact DTW for long sequences, due to its quadratic complexity. For this purpose, FTW proposes to estimate the time warping distance by using a lower bounding distance measure on a coarse and compact version of the original sequences.

A major disadvantage of DTW is that it tends to produce “singularities”, that are alignments of a single point in a sequence with multiple points of another sequence. This phenomenon becomes undesirable when unexpected singularities are produced. An effective variant of DTW able to reduce the phenomenon of singularities is Derivative Dynamic Time Warping (DDTW) [13]. The novelty of DDTW is that local derivatives of data points are estimated to capture information on slopes and trends in the sequences and find the correct warping.

An alternative approach to time series similarity detection is based on string matching measures. LCSS (*Longest Common SubSequence*) [14] is a variant of the edit distance that uses the length of the longest common subsequence of two sequences to define the distance between them. LCSS can handle time series with noise, but suffers from large-grained similarity.

A more refined measure based on edit distance is EDR (*Edit Distance with Real sequences*) [15], which performs the same distance quantization of LCSS (parametric with respect to a certain tolerance threshold) to remove noisy effects. In contrast to LCSS, EDR penalises the gaps between two matched subsequences according to the lengths of gaps. Unlike LCSS and EDR, ERP (*Edit distance with Real Penalty*) [16] is a metric and still supports local time shifting. ERP can be seen as a variant of EDR and DTW, although it does not require a noise-tolerance threshold like EDR, and does not replicate previous data points to add a gap like DTW. However, ERP shares with EDR and DTW the computational upper bound.

2.2 Dimensionality reduction

In order to improve the efficiency in time series data management, many research works have focused on *dimensionality reduction* to obtain a higher-level data representation. Typically, the goal is to approximate a (continuous) time series either with a piecewise discontinuous function or a low-order continuous function.

The first category includes Discrete Wavelet Transform (DWT) [17, 18], Piecewise Aggregate Approximation (PAA) [19, 20], and Adaptive Piecewise Constant Approximation (APCA) [21]. Using DWT, a time series is represented as a finite length, fast decaying oscillating waveform (mother wavelet), which is scaled and translated to match the original series. Unlike the continuous version of wavelet transform, the mother wavelet in DWT is discretely sampled.

PAA transforms a time series of n points in a new one composed by p segments (with $p \ll n$). All these segments have size equal to n/p . A segment is represented by a coefficient, which is the mean value of the data points falling

within the segment. Like PAA, APCA approximates a time series with a sequence of segments, each represented by the mean value of data points falling within it. A major difference is that APCA identifies segments of variable length. The $\mathcal{O}(n \log n)$ APCA algorithm is able to produce high-quality approximation by resorting to well-known solutions from the wavelet domain.

Dimensionality reduction techniques can be combined with existing similarity measures, in order to improve the computational cost in similarity searches. In particular, the use of DTW on the coefficients obtained by segmentation of time series has been investigated [22, 20].

Approaches that approximate a time series with a continuous polynomial include Discrete Fourier Transforms (DFT) [23, 24], splines, non-linear regression, and Chebyshev polynomials [25]. A very desirable requirement is the minimax approximation, i.e. an approximation that minimises the maximum deviation from the original data points. Although the optimal minimax polynomial is difficult to compute, it has been demonstrated that the Chebyshev approximation is very close to this polynomial and can be easily computed [26].

3 Derivative Time Series Segment Approximation

In this section we present a method for modelling time series into a compact representation, which suitably synthesizes the significant variations in the time series profile. The method is called *DSA (Derivative time series Segment Approximation)*, as it intuitively segments the derivative version of a time series before of approximating it into a high level representation.

Let $T = [x_1, \dots, x_n]$ be a time series, where the timestamp associated to x_1 is assumed to be zero. DSA computes in $\mathcal{O}(n)$ a new sequence τ of p values, with $p \ll n$, in three main steps: (i) derivative estimation, (ii) segmentation, and (iii) segment approximation.

3.1 Derivative estimation

Given a time series $T = [x_1, \dots, x_n]$, the derivative estimation step yields a sequence $\hat{T} = [\hat{x}_1, \dots, \hat{x}_n]$, whose elements \hat{x}_i are first derivative estimates. A simple derivative estimation model, exploited in [13] and hereinafter referred to as DDTW model, is the following:

$$\hat{x}_i = \begin{cases} \hat{x}_{i+1} & \text{if } i = 1 \\ \frac{1}{2}[(x_i - x_{i-1}) + \frac{1}{2}(x_{i+1} - x_{i1})] & \text{if } i \in [2..n-1] \\ x_{i1} & \text{if } i = n. \end{cases}$$

This estimation model computes for each point (except the first and the last one in the series) the mean value between the slope of the line from the left neighbor to the point and the slope of the line from the left neighbor to the right neighbor.

We slightly modify the above model by considering also the slope of the line from the point to the right neighbor; this modification leads to an algebraic simplification producing an expression which is equivalent to consider only the

slope of the line from the left neighbor to the right neighbor. Neighbors are also considered when computing the derivatives of the first and last points:

$$\dot{x}_i = \begin{cases} x_{i+1} - x_i & \text{if } i = 1 \\ \frac{1}{2}(x_{i+1} - x_{i1}) & \text{if } i \in [2..n-1] \\ x_i - x_{i1} & \text{if } i = n. \end{cases}$$

As experimentally founded, our derivative estimation model surprisingly gives approximation errors lower than the DDTW model.

3.2 Segmentation

The segmentation of a time series of length n consists in identifying $p - 1$ points ($p \ll n$) to partition it into p contiguous subsequences of points, i.e. segments, having similar features.

In our approach, the novel idea is that the segmentation is computed on derivative versions of time series. In particular, the derivative time series $\dot{T} = [\dot{x}_1, \dots, \dot{x}_n]$ is transformed into a sequence $S_{\dot{T}} = [s_1, \dots, s_p]$ of variable-length segments $s_i = [s_{i,1}, \dots, s_{i,k_i}] = [\dot{x}_{i_1}, \dots, \dot{x}_{i_{k_i}}]$, such that:

- $s_{1,1} = \dot{x}_1$,
- $s_{p,k_p} = \dot{x}_n$,
- for each $i \in [1..p-1]$, s_{i,k_i} immediately precedes $s_{i+1,1}$ in the time axis.

The critical aspect in segmentation is to determine the segment delimiters. Our approach falls into the sliding windows category: a segment is grown until it exceeds an error threshold, and the process repeats starting from the next point not yet considered. The key idea in our method is simply to break a series according to the first point such that the absolute difference between it and the mean of the previous points is above a certain threshold; this point becomes the anchor for the next segment to be identified in the rest of the series.

Formally, let $\mu(s_i)$ denote the average of the points in a potential segment s_i , defined as $\mu(s_i) = \frac{1}{k_i} \sum_{j=1}^{k_i} \dot{x}_{i_j}$. The sequence s_i , for each $i \in [1..p-1]$, is identified as a segment if and only if $|\mu([s_{i,1}, \dots, s_{i,j}]) - s_{i,j+1}| \leq \epsilon$, for all $j \in [1..k_i-1]$, and $|\mu([s_{i,1}, \dots, s_{i,k_i}]) - s_{i+1,1}| > \epsilon$.

Intuitively, this condition allows for aggregating subsequent data points having very close derivatives. In such a way, the growth segment represents a subsequence of points with a specific trend.

Parameter ϵ can be estimated in principle by considering an index of dispersion of the (derivative) data points within the same sequence around the respective mean value. Estimating ϵ might be accomplished based on one of three different contexts: globally to a given collection of time series, globally to a given time series, or locally to a given time series.

Given a dataset of N time series, the first way of computing ϵ may lead to the following definition: $\epsilon = \frac{1}{N} \sum_{k=1}^N \frac{|\dot{T}_k|}{\max_{i=1}^N \{|\dot{T}_i|\}} \sigma^2(\dot{T}_k)$ where $\sigma^2(\dot{T}_k)$ denotes the variance over the points in the k -th derivative series. The normalisation of the lengths is significant if the assumption of equally-sized series does not hold. The above definition can be adequate for the purpose provided that the time

series of a collection are quite equally sized. This cannot necessarily hold in some real domains (e.g. sensor network measurements) in which the time series generated may have varying lengths.

An opposite solution consists in estimating threshold ϵ locally to each time series, and in particular as function of the segment s_i which is currently being identified, that is, e.g. $\epsilon(s_i) = \sigma^2(s_i)$. However, although intuitively more accurate, this way of computing ϵ might be expensive.

A good trade-off between a collection-global and a series-local computation of ϵ is represented by the definition of a series-global computation: $\epsilon(\hat{T}_i) = \sigma^2(\hat{T}_i)$. We hereinafter refer to that as the definition of ϵ adopted in the DSA model.

While the main dimensionality reduction methods (Chebyshev polynomials, PAA and APCA) require in input the number of segments or coefficients which have to be identified, DSA does not need any input parameter. This is an important advantage of our method.

3.3 Segment approximation

All individual segments of a derivative time series are approximated with a synthetic information capturing their respective main features. More precisely, each segment s_i is mapped to a pair formed by the timestamp t_i of the last point ($\hat{x}_{i_{k_i}}$) of s_i and an angle that explains the average slope of the portion of time series bounded by s_i . This is mathematically expressed by the notion of arctangent applied to the mean of the (derivative) points in each segment.

Given a segmented derivative time series $S_{\hat{T}} = [s_1, \dots, s_p]$, a sequence $\tau = [(\alpha_1, t_1), \dots, (\alpha_p, t_p)]$ is computed, where

$$\alpha_i = \arctan(\mu(s_i)), \text{ for } i \in [1..p],$$

$$t_i = \begin{cases} k_i & i = 1 \\ t_{i-1} + k_i & i \in [2..p]. \end{cases}$$

4 Experiments

Experiment devised to assess both effectiveness and efficiency of the previously presented methods, including our DSA, in a clustering framework. Specifically, we tested LCSS, EDR, ERP DTW, DDTW and FTW directly as distance measures. Moreover, in order to include Chebyshev, PAA, APCA and our DSA in the comparative evaluation of distance measures, we chose to apply DTW over the sequences computed by each approximation scheme. The goal of the evaluation was to demonstrate the superiority of DSA to achieve fast and accurate similarity detection of time series in comparison with competing methods.

Input parameters required by LCSS, EDR, ERP, FTW and Chebyshev have been chosen as suggested in their respective works: constant gap for ERP has been set to 0, time interval for FTW has been set to 4, number of coefficients for Chebyshev has been set to 20 and the matching thresholds for LCSS and EDR have been assumed to be equal to $\frac{1}{4} \max\{\sigma(T_i)\}$ and $\min\{\sigma(T_i)\}$, for all T_i in the target dataset, respectively. As concerns PAA and APCA, since no indication about how to set the number of segments (p) is provided in their respective

works, we chose p as follows: for effectiveness evaluation, p was set as the number of segments produced by DSA, in order to evaluate accuracy reached at the same compression level; instead, for efficiency evaluation, we firstly measured effectiveness for different compression levels by varying p and, finally, chose p according to the best trade-off between accuracy and computational time.

4.1 Data description

We selected well-known datasets in the time series domain according to two main aspects: heterogeneity of the series profiles and significance of the series lengths. Table 1 summarises the main characteristics of the datasets, which are mostly available at http://www.cis.temple.edu/~latecki/TestData/TS_Koegh/.

dataset	size	classes	time steps
GunX	200	2	150
Trace	200	4	275
ControlChart	600	6	60
CBF	300	3	128
Twopat	800	4	128

Table 1. Main characteristics of the test datasets

GunX comes from the video surveillance domain, whereas Trace simulates signals representing instrumentation failures. In CBF, each class is characterized by a specific pattern, namely a plateau (C), an increasing ramp followed by a sharp decrease (B), a sharp increase followed by a decreasing ramp (F). ControlChart contains synthetically generated control charts which are classified into 6 classes. In Twopat, two different patterns (upward step and downward step) are used to define the classes.

4.2 Clustering method and quality measures

In our work the choice of a clustering scheme is functional to a comparative evaluation of methods for similarity detection. In this paper we employ a traditional agglomerative hierarchical clustering (AHC) algorithm [27] into our clustering framework.

Since the availability of reference classifications for the test datasets, the desired number of clusters at a hierarchy level is used as AHC termination criterion. Also, evaluating clustering effectiveness can be accomplished by adopting an external validity criterion; we use the well-known F-measure (ranging within [0..1]), which is defined in terms of classic Information Retrieval notions' precision and recall [28]. The objective is to assess how well a clustering fits a predefined scheme of known classes (natural clusters).

4.3 Preprocessing time series

A preliminary step may consists in preprocessing raw time series in order to dampen the effect due to noise in data and to improve both effectiveness and efficiency in the clustering task. This is usually accomplished by using some

smoothing models. Moving average represents the simplest family of smoothing models, as it is a compromise between the mean model and the random walk model. Given an original series $T = [x_1, \dots, x_n]$, a centered q -point moving average recomputes the data points as follows:

$$x_i^{smoothed} = \begin{cases} \mu([x_1, \dots, x_{i+r}]) & \text{if } i-r \leq 0 \\ \mu([x_{i-r}, \dots, x_{i+r}]) & \text{if } i-r > 0 \text{ and } i+r \leq n \\ \mu([x_{i-r}, \dots, x_n]) & \text{if } i+r > n \end{cases}$$

where q is the smoothing degree (i.e. the maximum width of the moving average) and $r = (q-1)/2$ denotes the maximum number of back and forward points considered for smoothing the i -th point. The centered q -point version considers both previous and next observations around a center, unlike simple moving average, although it still treats these points equally.

More refined models, such as exponential smoothing models, compute the weighted average of past observations with more weight given to the more recent values and decreasing weights for earlier values. The formula for simple exponential smoothing is as follows:

$$x_i^{smoothed} = \begin{cases} x_i & \text{if } i = 1 \\ wx_i + (1-w)x_{i-1}^{smoothed} & \text{if } i > 1 \end{cases}$$

where w is a smoothing weight with values in $[0..1]$.

4.4 Effectiveness evaluation

In this section we aimed at checking the ability of DSA and the competing methods in producing high-quality clustering. To accomplish this, we explored how clustering results can be influenced by choosing different alternatives for data preprocessing and setting the parameters therein involved. Then, we compared DSA with the competing methods according to their respective best settings.

Tuning preprocessing parameters. We employed two schemes of preprocessing, based on centered moving average and simple exponential smoothing. In the first case, we had to set the smoothing degree $q (= 2r + 1)$, whereas in case of exponential smoothing we tried different values for the smoothing weight w . In particular, q was set to typical values, i.e. 5 and 9, and w was varied within $[0..1]$ by a 0.1 step. We performed multiple iterations of smoothing (up to 5) in order to handle possibly excessive noise. On the other hand, we also tried to not perform any smoothing operation in order to prevent unnecessary loss of information for series with very low noise.

We performed tuning tests on DSA as well as on the competing methods. For the sake of brevity, Table 2 summarises the best preprocessing setups for DSA and the other methods on the selected datasets. Term MA (resp. EXP) stands for moving average (resp. exponential smoothing) and is followed by the value set for q (resp. w) and the number of iterations.

DSA vs. competing methods. Table 3 gives a summary of results obtained by DSA and the competing methods from an accuracy viewpoint. The reported results correspond to F-measure values obtained by algorithm AHC, and they

	GunX	Trace	ControlChart	CBF	Twopat
DTW	MA $q=9$ it=1	No preproc.	EXP $w=0.8$ it=1	EXP $w=0.5$ it=1	EXP $w=0.8$ it=3
LCSS	MA $q=5$ it=1	No preproc.	MA $q=9$ it=4	MA $q=9$ it=4	MA $q=5$ it=1
EDR	MA $q=9$ it=4	MA $q=5$ it=3	EXP $w=0.5$ it=2	EXP $w=0.8$ it=2	MA $q=5$ it=2
ERP	EXP $w=0.6$ it=1	No preproc.	MA $q=5$ it=1	EXP $w=0.15$ it=5	EXP $w=0.5$ it=3
DDTW	EXP $w=0.1$ it=1	MA $q=9$ it=1	EXP $w=0.2$ it=3	MA $q=9$ it=3	MA $q=9$ it=1
FTW	EXP $w=0.1$ it=1	No preproc.	EXP $w=0.8$ it=1	EXP $w=0.1$ it=2	EXP $w=0.3$ it=3
DTW on Chebyshev	EXP $w=0.2$ it=1	EXP $w=0.9$ it=1	MA $q=5$ it=4	EXP $w=0.5$ it=2	MA $q=9$ it=5
DTW on PAA	EXP $w=0.5$ it=1	EXP $w=0.1$ it=1	EXP $w=0.6$ it=4	EXP $w=0.6$ it=2	EXP $w=0.8$ it=5
DTW on APCA	MA $q=9$ it=3	MA $q=9$ it=1	MA $q=9$ it=2	MA $q=5$ it=3	MA $q=9$ it=4
DTW on DSA	EXP $w=0.3$ it=3	MA $q=9$ it=4	EXP $w=0.5$ it=5	EXP $w=0.3$ it=3	EXP $w=0.3$ it=3

Table 2. Summary of best preprocessing setups

	LCSS	EDR	ERP	DTW	DDTW	FTW	DTW on Chebyshev	DTW on PAA	DTW on APCA	DTW on DSA
GunX	0.60	0.68	0.67	0.67	0.64	0.67	0.67	0.66	0.64	0.68
Trace	0.36	0.58	0.77	0.75	1	0.77	0.65	0.77	0.74	1
ControlChart	0.66	0.83	0.81	0.86	0.86	0.83	0.80	0.86	0.71	0.86
CBF	0.66	0.84	0.65	0.72	0.99	0.67	0.69	0.72	0.82	0.99
Twopat	0.39	0.41	0.39	0.81	1	0.49	0.45	0.66	0.72	1

Table 3. Summary of best clustering quality results

refer to the best preprocessing setups respectively chosen for each similarity method on each dataset (see Table 2).

Table 3 shows that DSA fares as good as DDTW, which turns out to be mostly the best performing method among the remaining ones. Notice also the relative better performance of DDTW against FTW, Chebyshev, PAA and APCA. EDR and ERP perform similarly and both outperform LCSS. Yet, DDTW behaves as good or far better than DTW on all datasets but GunX, whose 2-class series have trends very similar but shifted in the time axis. The 2-class composition of this dataset is probably a reason why clustering results are relatively poor for all methods, suggesting that AHC fails in distinctly separating the two classes of series. We can also observe that DSA drastically outperforms the other dimensionality reduction techniques (Chebyshev, PAA and APCA).

4.5 Efficiency evaluation

We present here the time performances of DSA based clustering, and a comparison with respect to DTW, DDTW, FTW, Chebyshev, PAA and APCA based clustering. We left string matching based approaches out of the presentation since they turned out to be significantly slower than the other methods.¹

¹ Experiments were conducted on a platform Intel Pentium IV 3GHz with 2GB memory and running on Microsoft Windows XP Pro.

	DTW	DDTW	FTW	DTW on Chebyshev	DTW on PAA	DTW on APCA	DTW on DSA
GunX	280,078	330,281	64,500	12,328	56,124	117,125	62,281
Trace	947,516	1,420,969	163,347	42,860	200,684	181,750	87,719
ControlChart	559,063	478,235	82,656	147,984	163,249	226,016	104,875
CBF	643,797	573,547	56,766	36,203	152,462	199,265	155,968
Twopat	5,068,859	5,959,391	946,484	315,906	1,260,537	1,630,344	356,609

Table 4. Summary of best time performances

Table 4 reports the time performances (in milliseconds) on each dataset, using setups as in Table 2. As we expected, DTW and DDTW are drastically outperformed by the other measures, while APCA is slower than FTW, Chebyshev, PAA and DSA. DSA is comparable to FTW and PAA on all datasets but in Trace and Twopat, in which DSA is faster: this is particularly significant since Twopat and Trace are the largest datasets among the competing ones regards to the number of series and the mean series length, respectively. Chebyshev is found as the faster method; this is mainly due to the choice done for the number of coefficients (20). Nevertheless, DSA remains comparable while achieving drastically higher accuracy.

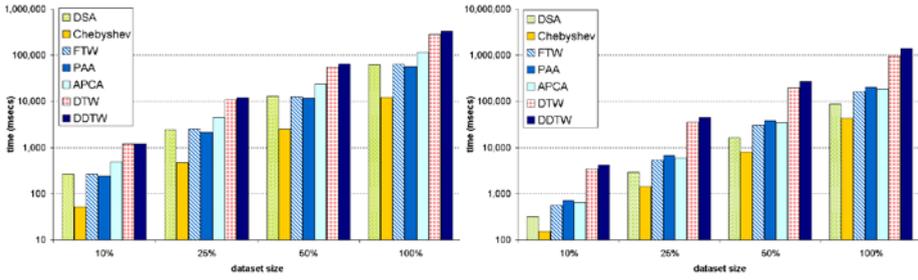
We also carried out experiments to test the time performances by varying the dataset size. Each dataset was sampled in portions with size 10%, 25%, 50%, and 100%; then, for each dataset and for each method, we carried out one run of AHC over each of these portions. Figure 1 shows that the advantage of DSA over DTW and DDTW absolutely increases with the dataset size.

Finally, we evaluated the effect of dimensionality reduction due to the segmentation performed by DSA. Experiments showed that DSA reduces the series lengths from 54% up to 77%, with noteworthy advantages in time performance.

5 Conclusion

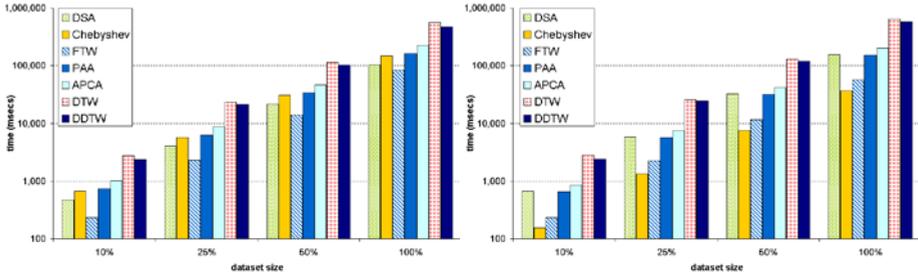
The presented work originally arises from our review of the state-of-the-art of similarity detection in time series, and from the finding that no existing method is able to provide both high effectiveness and efficiency. This prompted us to devise a solution to the identified challenge by combining the notions of derivative estimation and segmentation into a concise yet feature-rich representation model, called *DSA (Derivative time series Segment Approximation)*. Experiments highlight that DSA-based time series similarity detection is as good or better than both the most accurate and the fastest methods among the competing ones, thus guaranteeing the best trade-off between effectiveness and efficiency.

We are currently working on making DSA able to deal with amplitude translation and scaling, which may be significant in some application domains. Moreover, we shall provide the beneficial of DSA in summarising sets of time series to compute cluster prototypes that enable high cluster compactness and separation. Finally, we would like to extend DSA for multidimensional time series (i.e. moving object trajectories), to which many research work has recently paid attention.



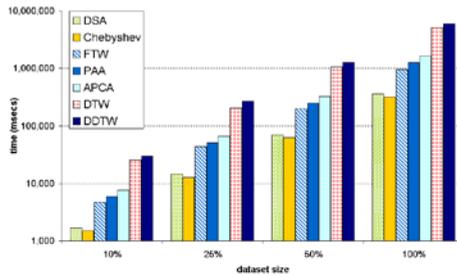
(a) GunX

(b) Trace



(c) ControlChart

(d) CBF



(e) Twopat

Fig. 1. Time performances

References

1. Agrawal, R., Faloutsos, C., Swami, A.N.: Efficient Similarity Search in Sequence Databases. In: Proc. FODO Conf. (1993) 69–84
2. Rabiner, L., Juang, B.H.: Fundamentals of Speech Recognition. Englewood Cliffs, N. J. (1993)
3. Berndt, D.J., Clifford, J.: Using Dynamic Time Warping To Find Patterns in Time Series. In: Proc. AAAI Workshop on Knowledge Discovery in Databases. (1994) 359–370
4. Keogh, E.: Exact Indexing of Dynamic Time Warping. In: Proc. VLDB Conf. (2002) 406–417

5. Kim, S.W., Park, S., Chu, W.W.: An Indexed-Based Approach for Similarity Search Supporting Time Warping in Large Sequence Databases. In: Proc. ICDE Conf. (2001) 607–614
6. Yi, B.K., Jagadish, H.V., Faloutsos, C.: Efficient Retrieval of Similar Time Sequences Under Time Warping. In: Proc. ICDE Conf. (1998) 201–208
7. Sakurai, Y., Yoshikawa, M., Faloutsos, C.: FTW: Fast Similarity Search under the Time Warping Distance. In: Proc. ACM PODS. (2005) 326–337
8. Keogh, E., Palpanas, T., Zordan, V., Gunopulos, D., Cardle, M.: Indexing Large Human-Motion Databases. In: Proc. VLDB Conf. (2004) 780–791
9. Vlachos, M., Hadjieleftheriou, M., Gunopulos, D., Keogh, E.: Indexing Multi-Dimensional Time-Series with Support for Multiple Distance Measures. In: Proc. ACM KDD Conf. (2003) 215–225
10. Wei, L., Keogh, E., Herle, H.V., Mafra-Neto, A.: Atomic wedge: Efficient query filtering for streaming times series. In: Proc. IEEE ICDM Conf. (2005) 490–497
11. Fung, W., Wong, M.: Efficient Subsequence Matching for Sequences Databases under Time Warping. In: Proc. IDEAS Conf. (2003) 139–148
12. Keogh, E., Wei, L., Xi, X., Lee, S., Vlachos, M.: LB_Keogh Supports Exact Indexing of Shapes under Rotation Invariance with Arbitrary Representations and Distance Measures. In: Proc. VLDB Conf. (2006) 882–893
13. Keogh, E., Pazzani, M.: Dynamic Time Warping with Higher Order Features. In: Proc. SIAM Int. Conf. on Data Mining. (2001)
14. Vlachos, M., Gunopulos, D., Kollios, G.: Discovering Similar Multidimensional Trajectories. In: Proc. ICDE Conf. (2002) 673–684
15. Chen, L., Özsu, M.T., Oria, V.: Robust and Fast Similarity Search for Moving Object Trajectories. In: Proc. ACM SIGMOD Conf. (2005) 491–502
16. Chen, L., Ng, R.: On The Marriage of Lp-norms and Edit Distance. In: Proc. VLDB Conf. (2004) 792–803
17. Chan, K., Fu, A.: Efficient Time Series Matching by Wavelets. In: Proc. ICDE Conf. (1999) 126–133
18. Wu, Y., Agrawal, D., Abbadi, A.: A Comparison of DFT and DWT Based Similarity Search in Time-Series Databases. In: Proc. ACM CIKM Conf. (2000) 488–495
19. Keogh, E., Chakrabarti, K., Pazzani, M., Mehrotra, S.: Dimensionality Reduction for Fast Similarity Search in Large Time Series Databases. *Knowledge and Information Systems* **3** (2000) 263–286
20. Keogh, E., Pazzani, M.: Scaling up Dynamic Time Warping for Datamining Applications. In: Proc. ACM KDD Conf. (2000) 285–289
21. Keogh, E., Chakrabarti, K., Mehrotra, S., Pazzani, M.: Locally Adaptive Dimensionality Reduction for Indexing Large Time Series Databases. In: Proc. ACM SIGMOD Conf. (2001) 151–162
22. Keogh, E., Pazzani, M.: Scaling up Dynamic Time Warping to Massive Datasets. In: Proc. PKDD Conf. (1999) 1–11
23. Rafiei, D., Mendelzon, A.O.: Efficient Retrieval of Similar Time Sequences Using DFT. In: Proc. FODO Conf. (1998)
24. Rafiei, D., Mendelzon, A.: Similarity-based queries for time series data. In: Proc. ACM SIGMOD Conf. (1997) 13–25
25. Mason, J.C., Handscomb, D.: *Chebyshev Polynomials*. Chapman & Hall (2003)
26. Cai, Y., Ng, R.: Indexing Spatio-Temporal Trajectories with Chebyshev Polynomials. In: Proc. ACM SIGMOD Conf. (2004) 599–610
27. Jain, A., Dubes, R.: *Algorithms for Clustering Data*. Prentice-Hall (1988)
28. van Rijsbergen, C.J.: *Information Retrieval*. Butterworths (1979)

Finding Generalized Closed Frequent Itemsets for Mining Non Redundant Association Rules

Corrado Loglisci¹, Margherita Berardi¹, Saverio D'Alessandro³, and Pietro Leo²

¹Dipartimento di Informatica, Università degli Studi di Bari

via Orabona, 4 - 70126 Bari - Italy

²IBM GBS - Innovation Center - Bari

Via Tridente, 42/14 - 70125 Bari

¹{loglisci,berardi}@di.uniba.it

²pietro_leo@it.ibm.com

³dalessandro_saverio@yahoo.it

Abstract. Generalized association rules are a very important extension of traditional association rules which allows to exploit taxonomical knowledge defined over items to be mined. However, by using a taxonomy several thousands of rules are discovered and the most of them can be redundant. In this paper, we propose a solution to the problem of mining non redundant generalized association rules by resorting to the closed itemset framework and to the concept of minimal non-redundant rules. We define a formal framework and design an algorithm which solves the problem of mining *generalized closed frequent itemsets*. Generation of non-redundant generalized rules from the set of generalized closed frequent itemsets is considered as well. The proposed framework has been applied to biomedical textual data analysis. Experimental results are reported and conclusions are drawn.

Key words: generalized closed itemsets, association rule mining, non-redundancy, taxonomies

1 Introduction

The discovery of association rules has attracted a great deal of attention in data mining research. Association rules are a class of regularities introduced by [1] which discover co-occurrence relations among pieces of data (i.e., items). An association rule is expressed in form of $X \Rightarrow Y$, where both the antecedent X and the consequent Y are sets of items (i.e., *itemsets*) such that $X \cap Y = \emptyset$. The meaning of such rules is quite intuitive: given a database D of transactions, where each transaction $T \in D$ is a set of items, $X \Rightarrow Y$ expresses that whenever a transaction T contains X than T probably contains Y also. Two parameters are reported for association rules, namely the *support*, $s(X \Rightarrow Y) = s(X \cup Y)$, namely the percentage of transactions in D containing the whole itemset $X \cup Y$, and the *confidence*, $c(X \Rightarrow Y) = s(X \cup Y)/s(X)$, estimating the probability $P(X \cup Y)/$

$P(X)$. The goal of association rule mining is to find all the rules with support and confidence exceeding user specified thresholds, henceforth called *minsup* and *minconf*, respectively. The problem of mining association rules is usually solved by a two-step procedure:

Given, a database D of transactions and threshold values for *minsup* and *minconf*, (1) generate all itemsets with support greater than or equal to *minsup* (i.e., frequent itemsets); (2) for each frequent itemset X_i , find all association rules $X_j \Rightarrow X_i - X_j$, $X_j \subset X_i$, with confidence greater than or equal to *minconf* (i.e., valid association rules).

Effectiveness and efficiency of association rule mining algorithms strongly depend on the strategy adopted to generate frequent itemsets. In the literature, several works have been proposed to keep under control the explosion of the search space in the candidate itemset lattice [4, 8, 11, 13, 18], especially when low levels of minimum support bring to large amount of itemsets. Some works have also presented solutions to obtain non redundant sets of rules that are in some way representative of the whole space of valid rules. To the best of our knowledge, most of related works do not consider strategies to prune redundant knowledge during the generation phase, that may also allow to reduce the number of expensive tests. They usually operate in a post-processing mode by evaluating redundancy of discovered association rules and selecting compact and representative subsets [2, 10]. The problem of redundancy in association rule mining is even more emphasized in the case of generalized association rules [16, 9], where rules are discovered at different levels of the taxonomy defined on database items. Algorithms proposed for generalized association rule mining generally differ in the strategy to traverse the item taxonomy and in the generation of *generalized itemsets* (GI).

In this paper, the problem of mining non redundant generalized association rules is tackled by resorting to the concept of closed itemsets [6]. Closed itemsets are maximal sets among any other itemsets occurring in the same transactions. Formally, the itemset X is said to be a *closed itemset* when there is no other itemset Y such that $X \subset Y$ and $s(X)=s(Y)$. For instance, let $\{A,B,C\}$ be a set of items of a database transaction and $\langle A,B,C \rangle$, $\langle A,B \rangle$ two itemsets with the same support; it is quite obvious that information captured by $\langle A,B \rangle$ is “subsumed” in $\langle A,B,C \rangle$ which expresses additional information too. Intuitively, selecting itemsets like the first one preserves information and allows to lower the computational cost as well. Considering the notion of *Galois connection* [5, 7] and *closed itemset lattice* [14, 15], we provide a suitable definition of *generalized closed itemsets* (GCI). This is exploited to design a novel algorithm for *generalized closed frequent itemset* discovery (GCFI), named DELIS (DEscending Levels Increasing Size). DELIS allows to generate compact sets of itemsets by preventing information loss. The paper provides as further contribution some operative definitions useful for non redundant rule generation which fully exploit GCFIs. At this aim, we resort to the idea of *minimal non-redundant rules* given in [3] that allows to discard rules with the same support and confidence at the aim to save the most informative ones. For instance, when the following set of

rules with identical support and confidence values is generated: $R_1: A \Rightarrow B, C$; $R_2: A \Rightarrow B, C, D$; $R_3: A \Rightarrow B, C, D, E$, the last one should be considered as the minimal non-redundant rule since it captures information expressed by R_1 and R_2 as well.

The paper is organized as follows. In the next section, a formalization of the problem of mining generalized closed itemsets and an high level description of the DELIS algorithm are reported. Section 3 reports an operative characterization to mine non-redundant generalized rules. In Section 4, a text mining application to the biomedical literature field is discussed. Generalized association rules are discovered from Medline abstracts to detect associations between biomedical concepts belonging to the MeSH (Medical Subject Headings) taxonomy ¹. Finally, some conclusions are drawn in Section 5.

2 The Problem of Finding Generalized Closed Itemsets

In order to formalize the problem we intend to solve, some useful definitions are necessary.

2.1 Formal Statement

A generalized itemset can be formally stated as follows. Let $I: \{x_1, \dots, x_n, y_1, \dots, y_m, z\}$ be a set of distinct items, D be a set of transactions T , each $T \subseteq I$. An itemset X that is composed by k items is denoted as $|X|=k$. A transaction T supports an itemset X when $X \subseteq T$. Let G be a taxonomy represented as an acyclic directed graph over the items in I . An edge $x_i \leftarrow y_j$ in G denotes a is-a relationship: the item x_i is called *child* of y_j (the item y_j is called *parent* of x_i). An item z is called *ancestor* of x_i if there is a path from x_i to z (x_i is called *descendant* of z). A transaction T supports an item $y_j \in I$ when $y_j \in T$, or $\exists x_i \in T$ such that y_j is *ancestor* of x_i . An itemset Y is called *generalized itemset* if no any items in the set is an ancestor item of the others. An itemset Y is called *ancestor* of an itemset X , if Y can be obtained by replacing one or more items in X with one of their ancestors and $|X| = |Y|$ (conversely X is called *descendant* of Y). A generalized itemset Y is frequent (GFI) if the number of transactions supporting Y is greater than or equal to the minimum support.

Definition 1. Given an item x_i and a taxonomy G , we define the *item depth* function $d_{X_i}: I \rightarrow \mathbb{N}^*$ in the following way:

- (i) $d_{X_i}(x_i)=1$, iff x_i is the first non-leaf item in G ,
- (ii) $d_{X_i}(x_i)=d_{X_i}(\text{parent}(x_i))+1$, iff x_i is not the first non-leaf item in G , and $\text{parent}(x_i)$ is a parent of x_i .

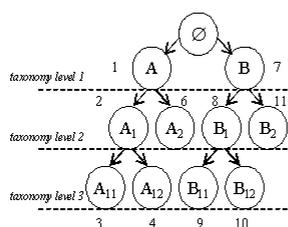
Definition 2. Given an itemset X and a function d_{X_i} , the function $d_X: \wp(I) \rightarrow \mathbb{N}^*$ such that $d_X(X)=\max(d_{X_i}(x_i))$, $x_i \in X$, is called *itemset depth*.

¹ <http://www.nlm.nih.gov/mesh/>

Let f be the function which returns the set of itemsets that can be generated from a given set of transactions, g the function which returns the set of transactions supporting a given itemset, the composition function $h=f \circ g$ be the *Galois closure operator* as defined in [3] that meets the properties described in [5, 7], the following definitions are given:

Definition 3. Let $Y=\langle y_1, y_2, \dots, y_j, \dots, y_h \rangle$ be a generalized itemset, Y is a *generalized closed itemset* iff:

- (i) \nexists item $w, w \in I, w \notin Y, w$ is not ancestor of $y_j \in Y,$
- (ii) $d_X(Y) \geq d_{X_i}(w),$
- (iii) $g(Y) \subseteq g(\langle w \rangle).$



Transaction Identifier	A ₁₁	A ₁₂	A ₂	B ₁₁	B ₁₂	B ₂	A	B	A ₁	B ₁
1	1	0	1	1	1	1	1	1	1	1
2	0	1	0	0	0	1	1	1	1	0
3	1	1	0	1	1	1	1	1	1	1
4	1	0	1	0	0	0	1	0	1	0
5	1	1	1	1	1	1	1	1	1	1
6	0	0	1	0	0	0	1	0	0	0

Fig. 1. A taxonomy over a set of the items and a transaction table for both leaf items and parent items.

Let us consider the taxonomy and the set of transactions reported in Fig. 1 and the item $w=B,$ it can be straightforward derived that the itemset $Y=\langle A \rangle$ is a GCI while the itemset $Y=\langle A_{11}, A_{12} \rangle$ is not. In fact, $g(\langle A \rangle)=\{1,2,3,4,5,6\}$ and $g(\langle B \rangle)=\{1,2,3,5\}.$ Hence, $g(\langle A \rangle) \subseteq g(\langle B \rangle)$ does not hold $\Rightarrow Y$ is a GCI. On the contrary, $g(\langle B \rangle)=\{1,2,3,5\}.$ Hence, $g(\langle A_{11}, A_{12} \rangle) \subseteq g(\langle B \rangle)$ does hold $\Rightarrow Y$ is not a GCI. Trivially, if we consider child items of $w=B.$

Considering the above definition of GCI, the problem we intend to solve can be formulated as follows:

Given a set of transactions $D,$ a set of items $I,$ a taxonomy G over the set $I,$ a threshold value for minsup;

Find the set of generalized closed frequent itemsets (GCFI) such that $\forall Y \in \text{GCFI} : s(Y) \geq \text{minsup}.$

2.2 The DELIS (DEscending Levels Increasing Size) algorithm

The problem presented in the previous subsection is solved by the DELIS (DEscending Levels Increasing Size) algorithm. A suitable order relation “ \prec ” among

items and itemsets has been defined in DELIS in order to traverse the space of items. This allows us to make some assumptions.

Given a set of items ² I , a taxonomy G over I , a function $O: I \rightarrow \{1, 2, \dots, |I|\}$, we assume that:

(a) $\forall x_i, x_j \in I, d_{X_i}(x_i) = d_{X_i}(x_j), O(x_i) < O(x_j) : x_i \prec x_j$, (e.g., for A_1, A_2 in Fig. 1, $O(A_1)=2, O(A_2)=6: A_1 \prec A_2$);

(b) $\forall x_i, x_j \in I, \forall x_{ik}$ child of $x_i, d_{X_i}(x_i) = d_{X_i}(x_j), d_{X_i}(x_{ik}) = d_{X_i}(x_j) + 1, O(x_i) < O(x_{ik}) < O(x_j) : x_i \prec x_{ik} \prec x_j$, (e.g., for A_1, A_2, A_{11} in Fig. 1, $O(A_1)=2, O(A_2)=6, O(A_{11})=3: A_1 \prec A_{11} \prec A_2$);

(c) given two itemsets $X_1, X_2, |X_1| = |X_2| = n, d_X(X_1) \leq d_X(X_2), \exists x_i^m, x_j^m, m \leq n$, the m -th items of X_1, X_2 respectively, $x_i^m \prec x_j^m : X_1 \prec X_2$.

These assumptions allow us to formulate main ideas that are implemented in DELIS. First, for each level of the taxonomy, frequent k -itemsets can be generated by joining frequent $(k-1)$ -itemsets found in the previous iteration. Second, by downward traversing the taxonomy, it is guaranteed that all descendants of infrequent itemsets are infrequent too. Third, since it can be proved that descendants of not closed itemsets are not closed too, the algorithm prevents the generation of itemsets whose ancestors are not frequent closed. Finally, closed itemsets can be generated by means of ancestor frequent closed itemsets that have been found at the upper level of the taxonomy.

An high-level description of the DELIS generalized closed frequent itemset discovery is reported in alg. 1, where:

- k denotes the current level of the taxonomy and *leafItemLevel* represents the lowest level of the taxonomy;
- $GFI_{h,k}$ is the list of GFI with size h generated at the k level;
- $GCFI$ is the list of discovered GCFI;
- *lowerLevel*(G, k) returns the lower level of the taxonomy G w.r.t. the level k ;
- *frequentItemsGeneration*(k) returns frequent items at level k by exploiting the set of frequent items in $GFI_{1,k-1}$ if $k \neq \text{lowerLevel}(G, \emptyset)$;
- *join*(X_i, X_j) joins X_i and X_j , with $X_i \prec X_j$;
- *frequentItemsetsGeneration*($GFI_{h,k-1}, GFI_{h-1,k-1}$) returns $GFI_{h,k}$ by joining itemsets with size h that are descendants of frequent itemsets;
- *getClosed*($GFI_{h,k}$) returns the GCFI from the set GFI;
- *getGenerator*($GCFI_{h,k}$) returns the generator frequent itemsets from the set $GFI_{h,k}$. We provide the definition of generator itemset in the next section, where the set CG of all generator itemsets is used for the rule generation step.
- *leafItemLevel* represents the lowest level of the taxonomy.

A trace of alg. 1 by considering example reported in Fig. 1 and minsup=0.4 follows. Given $h=1$ and $k=\text{taxonomy level } 1$, $GFI_{h,k} = \{\langle A \rangle, \langle B \rangle\}$ is determined (line 4). Frequent itemsets with larger size are determined $GFI_{h,k} = \{\langle A, B \rangle\}$ and

² We assume that the set of transactions D contains only leaf-items of the taxonomy without generality loss.

closed itemsets $\{\langle A \rangle, \langle A, B \rangle\}$ are saved (lines 5-12). Then, the algorithm iteratively goes down towards lower levels of the taxonomy by scanning only the child items of closed frequent parent items: given $h=1$ and $k=\text{taxonomy level } 2$, $GFI_{h,k}=\{\langle A_1 \rangle, \langle A_2 \rangle, \langle B_2 \rangle\}$ (line 15) where $\langle A_2 \rangle$ and $\langle B_2 \rangle$ are not closed (Def. 3 by considering $w=A_1$). Then, (line 21) given $h=2$, $GFI_{h,k}=\{\langle A_1, A_2 \rangle, \langle A_1, B_2 \rangle, \langle A_2, B_2 \rangle\}$, and (line 23) the algorithm updates $GFI_{h,k}$ with $\{\langle A, B_2 \rangle, \langle A_1, B \rangle, \langle A_2, B \rangle\}$. By exploiting Def. 3 (line 24), only the frequent itemsets descendants from ancestor closed itemsets are proven to be closed. Finally, at $h=3$, $GFI_{h,k}=\{\langle A_1, A_2, B_2 \rangle, \langle A_1, A_2, B \rangle\}$.

Algorithm 1 Top-level description of DELIS algorithm.

```

1: input:  $D, G, \text{minsup}$ ;
2: output:  $GCFI, CG$ ;
3:  $GCFI \leftarrow \emptyset$ ;  $CG \leftarrow \emptyset$ ;  $GFI_{h,k} \leftarrow \emptyset$ ;
4:  $k \leftarrow \text{lowerLevel}(G, \emptyset)$ ;  $h \leftarrow 1$ ;  $GFI_{h,k} \leftarrow \text{frequentItemsGeneration}(k)$ ;
5: while  $GFI_{h,k} \neq \emptyset$  do
6:    $h \leftarrow h + 1$ ;  $GFI_{h,k} \leftarrow \emptyset$ ;
7:   for each  $X_i, X_j \in GFI_{h-1,k}$  and  $X_i \neq X_j$  do
8:      $GFI_{h,k} \leftarrow GFI_{h,k} \cup \text{join}(X_i, X_j)$ ;
9:   end for
10:   $GCFI \leftarrow GCFI \cup \text{getClosed}(GFI_{h,k})$ ;
11:   $CG \leftarrow CG \cup \text{getGenerator}(GFI_{h,k})$ ;
12: end while
13: while  $k \leq \text{leafItemLevel}$  do
14:   $h \leftarrow 1$ ;  $k \leftarrow \text{lowerLevel}(G, k)$ ;
15:   $GFI_{h,k} \leftarrow \text{frequentItemsGeneration}(k)$ ;
16:   $GCFI \leftarrow GCFI \cup \text{getClosed}(GFI_{h,k})$ ;
17:   $CG \leftarrow CG \cup \text{getGenerator}(GFI_{h,k})$ ;
18:  while  $GFI_{h,k} \neq \emptyset$  do
19:     $h \leftarrow h + 1$ ;  $GFI_{h,k} \leftarrow \emptyset$ ;
20:    for each  $X_i, X_j \in GFI_{h-1,k}$  and  $X_i \neq X_j$  do
21:       $GFI_{h,k} \leftarrow GFI_{h,k} \cup \text{join}(X_i, X_j)$ ;
22:    end for
23:     $GFI_{h,k} \leftarrow GFI_{h,k} \cup \text{frequentItemsetsGeneration}(GFI_{h,k-1})$ ;
24:     $GCFI \leftarrow GCFI \cup \text{getClosed}(GFI_{h,k})$ ;
25:     $CG \leftarrow CG \cup \text{getGenerator}(GFI_{h,k})$ ;
26:  end while
27: end while

```

3 Mining Non Redundant Generalized Association Rules

DELIS generates the set GCFI that represents a set of non redundant itemsets which can generate (trivially by [3]) the complete set of generalized frequent itemsets and their supports. This means that the rule set that is discovered by

using GCFI, which should be smaller in size, constitutes a minimal set of non redundant rules from which all valid generalized association rules can be inferred. By resorting to the approach reported in [3], we propose a suitable extension to generalized association rules:

Definition 4. Let X be a generalized closed frequent itemset: the rule $Y \Rightarrow X - Y$ is called *exact generalized association rule* iff $c(Y \Rightarrow X - Y) = 1$, $Y \subset X$.

Moreover, $s(X \cup Y) / s(Y) = 1 \wedge Y \subset X: s(X) = s(Y) \wedge h(X) = h(Y)$. $\forall Z, Y \subset Z \subset X: s(Z) = s(X) = s(Y) \wedge Z \Rightarrow X - Z$ is exact. This means that given Y itemset with smaller size, $Y \subset X$, it is possible to generate every Z . Hence, the following definitions can be provided.

Definition 5. Let X be a closed itemset, and h the Galois closure operator. An itemset Y , $|Y| > 1$, $Y \subseteq X$, is called *generator* of closed itemset X iff $h(Y) = X \wedge \nexists Y' \subseteq X$ with $Y' \subset Y \ni h(Y') = X$, $s(Y') = s(Y)$.

Definition 6. Let GCFI be the set of generalized closed frequent itemsets and CG be the set of generators: the set of rules discovered by using GCFI with confidence = 1 is *ExactRuleSet*: $\{Y \Rightarrow X - Y \mid X \in \text{GCFI} \wedge Y \in \text{CG} \wedge Y \text{ generator of } X \wedge s(X) = s(Y)\}$.

Definition 7. Let X be a generalized closed frequent itemset: the rule $Y \Rightarrow X - Y$ is called *approximate generalized association rule* iff $c(Y \Rightarrow X - Y) < 1$, $Y \subset X$.

Moreover, $s(X \cup Y) / s(Y) < 1 \wedge Y \subset X: s(X) < s(Y) \wedge h(Y) \subseteq h(X) = X$. $\forall Z, Z \subset X \wedge h(Z) \subseteq h(X): Z \Rightarrow X - Z$ is approximate.

Definition 8. Let GCFI be the set of generalized closed frequent itemsets and CG be the set of generators: the set of rules discovered by using GCFI with confidence < 1 is *ApproximateRuleSet*: $\{Z \Rightarrow X - Z \mid X \in \text{GCFI} \wedge Z \in \text{CG} \wedge h(Z) \subseteq h(X)\}$.

All valid generalized association rules can be deduced by the union of *ExactRuleSet* and *ApproximateRuleSet* which represents the minimal set of non redundant rules, that is the most informative rule set to be presented to the user.

4 Experiments

In this section, we intend to compare results obtained by running the standard Apriori algorithm and the DELIS algorithm. Generalized association rules are discovered on biomedical literature datasets obtained by querying PubMed³. An example of PubMed query formulated by biomedical researchers may ask for discovering the factors related to the reactions to Diabetes treatments (i.e., "Diabetes Drugs Response"). By using generalized association rules collections of abstracts can be compactly described and analysed in terms of strong co-occurrences discovered at multiple levels of abstraction. Submitting the query to PubMed, a set of relevant abstracts is returned and initially annotated by the BioTeKS Text Analysis Engine (TAE) provided within the IBM UIM Architecture [12], by using a local MeSH terms dictionary. The MeSH taxonomy is a

³ <http://www.ncbi.nlm.nih.gov/entrez/query.fcgi>

set of hierarchically related biomedical terms which permits to mine generalized association rules. The MeSH taxonomy is organized in 15 distinct hierarchies structured in a tree form that is about 11 levels deep. For each query, a single table of a relational database is created and fed with MeSHs occurring in the corresponding set of retrieved abstracts. In particular, each transaction of a single table is associated to an individual abstract and it is described in terms of items that correspond to MeSHs. The simplest representation, namely the boolean representation, is adopted in order to represent the occurrence of a MeSH term in an abstract and we use the “canonical” form of each MeSH term, which is available in the MeSH dictionary. In this study, two segments of Medline have been considered, that is the sets of abstracts related to two queries, namely *Neuropathy Ataxia Retinitis Pigmentosa* (NARP) and *Alzheimer Drug Interactions* (ADI). The first query returns 455 abstracts while the second 135 abstracts. Both have been described on the basis of about 50 MeSH terms. The first query generates a more sparse data set than the second query.

For each data set, generalized closed frequent itemsets and valid generalized association rules are generated by varying the minimum support value.

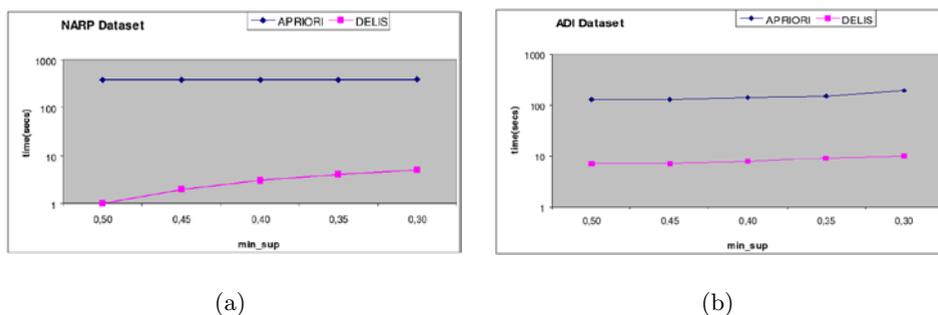


Fig. 2. Running times on the varying of the minimum support.

The DELIS algorithm produces the set of generalized closed frequent itemsets and the set of corresponding generators. Rules are subsequently generated as in the case of Apriori but exploiting the definition of exact rule set and approximate rule set. Experiments have been conducted in order to compare performances of the two algorithms along three different criteria, namely, running times of both frequent itemset and rule generation, complexity in terms of the number of itemsets as well as the number of association rules. Execution time of both the algorithms for the two data sets is reported in Fig. 2. Running times refer to executions performed on a 1.4 Ghz IBM Centrino notebook equipped with 512 Mb of RAM. We can observe that the Apriori algorithm requires more execution time on the sparse data sets while running times of DELIS seem to be not affected

by the data set. Data sparseness usually implies a low correlation in data and however, the correlation is more evident on the lowest level of the taxonomies. Apriori does not take into account this information since it basically combines frequent itemsets. On the contrary, DELIS generates itemsets at low levels of the taxonomies and it proceeds by generating descendant itemsets from their ancestors by descending them at more and more deep levels. This allows DELIS to gain time when the mining process is performed on sparse data sets.

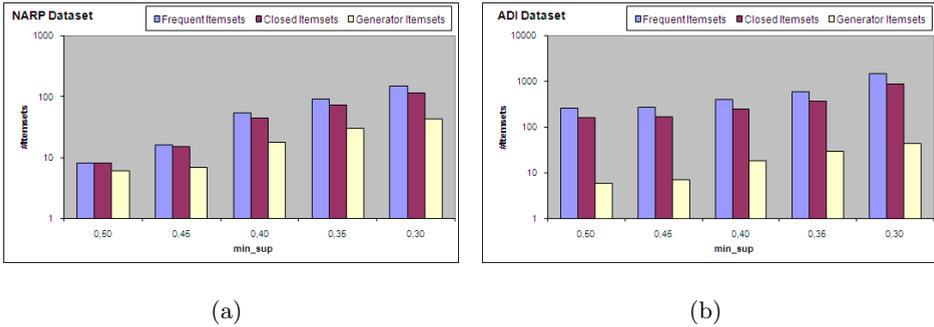


Fig. 3. Number of generated itemsets on the varying of the minimum support.

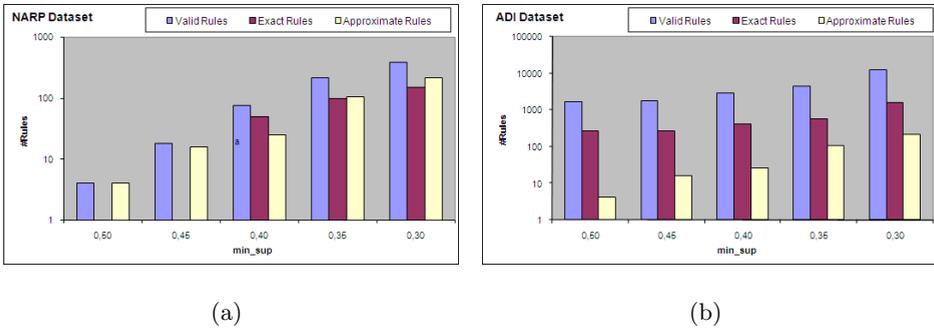


Fig. 4. Number of the discovered rules on the varying of the minimum support.

Considering the number of frequent itemsets that have been generated (see Fig. 3), we find that with the decreasing of the value of the minimum support, in both data sets, DELIS generates a lower number of frequent itemsets than in the

case of Apriori. Moreover, in the case of the *ADI* data set, DELIS generates less itemsets than in the case of the *NARP* data set. This is due to the fact that the *ADI* data set is more dense than the *NARP* data set, hence the positive effect of the closed itemsets generation is more evident, since data are more correlated. Actually, the mechanism subsumed by the closed itemset discovery allows to generate compact subsets of itemsets by preserving information on correlation among items. Consequently, the number of association rules that have been generated is also limited (see Fig.4). By scanning these sets of rules benefits coming from the use of the DELIS algorithm can be observed. For instance, we have considered rules generated for the *ADI* data set, where the effect of redundancy elimination is more evident, and we consider the experiment run with $minsup=0.3$ and $minconf=0.5$.

In the following, rules extracted by means of Apriori are showed:

- “R1” Alzheimer Disease \Rightarrow Chemical and drugs;
- “R2” Alzheimer Disease \Rightarrow Biological sciences,Chemical and drugs;
- “R3” Alzheimer Disease,Mental disorders \Rightarrow Biological sciences,Chemical and drugs;
- “R4” Alzheimer Disease \Rightarrow Biological sciences,Chemical and drugs,Mental disorder;
- “R5” Alzheimer Disease,Tauopathies \Rightarrow Biological sciences,Chemical and drugs;
- “R6” Alzheimer Disease \Rightarrow Biological sciences,Chemical and drugs,Tauopathies;
- “R7” Alzheimer Disease,Mental disorders,Tauopathies \Rightarrow Biological sciences, Chemical and drugs;
- “R8” Alzheimer Disease,Tauopathies \Rightarrow Biological sciences,Chemical and drugs, Mental disorders;
- “R9” Alzheimer Disease,Mental disorders \Rightarrow Biological sciences,Chemical and drugs,Tauopathies;
- “R10” Alzheimer Disease \Rightarrow Biological sciences,Chemical and drugs,Mental disorders,Tauopathies.

The first rule has 0.53 support and 0.79 confidence, while the others have 0.35 support and 0.52 confidence.

DELIS discovers the following rules:

- “R11” Alzheimer Disease \Rightarrow Chemical and drugs,Mental disorders,Tauopathies;
- “R12” Alzheimer Disease \Rightarrow Biological sciences,Chemical and drugs,Mental disorders,Tauopathies.

The first rule has 0.53 support and 0.79 confidence, while the other has 0.35 support and 0.52 confidence.

Reminding that association rules are monotone with respect to pruning on the consequent side (i.e., if $X \Rightarrow Y \cup \{A\}$ is valid, then the rule $X \Rightarrow Y$ is valid too), some considerations can be drawn. Rules extracted by Apriori with index “R2”, “R3”, “R4”, “R5”, “R6”, “R7”, “R8”, “R9” can be obtained by the “R10” rule, since they have the same support and confidence and this rule results to have the

shortest antecedent and the longest consequent. This means that moving an item from the consequent side to the antecedent side does not affect the meaning of the rule when the resulting rule has the same support and confidence. Therefore, DELIS is able to discover a single rule, the “R12”. On the contrary, in the case of “R1” rule, although the consequent of the rule is contained in the consequent of the “R10” rule, both of them should be saved since the confidence and support values are not preserved. For this reason, DELIS discovers another rule, rule “R11”, that has the same support and confidence of “R1” extracted by Apriori, and as consequent a superset of the “R1” rule. By compacting the consequent of the “R11” rule, the rule “R1” is obtained.

5 Conclusions

In this paper, we presented an approach for non redundant generalized association rule discovery by introducing the concept of generalized closed itemsets. The approach involves both the step of frequent itemsets discovery and the step of valid rules generation. Initial results show benefits of our proposal. Differently to Apriori algorithm for generalized rules, our approach discovers more compact sets of generalized rules containing the most informative rules. In [19] the author proposed a framework to discover a set of non redundant rules in terms of the most general ones: the most informative rules generally differ from the most general ones. Since this proposal concerned the discovery of non generalized rules, a comparison has not been performed. A work that is similar to our approach is described in [17] where the problem of mining generalized closed itemsets is described by introducing the concept of generalized Galois closure. Our contribute proposes an alternative framework for mining generalized closed frequent itemsets and, additionally, it introduces an operative definition for non redundant generalized rules. As future work, we plan to perform further experiments on other real-world datasets.

References

1. R. Agrawal and R. Srikant. Fast algorithms for mining association rules in large databases. In *VLDB '94: Proceedings of the 20th International Conference on Very Large Data Bases*, pages 487–499, San Francisco, CA, USA, 1994. Morgan Kaufmann Publishers Inc.
2. M. Z. Ashrafi, D. Taniar, and K. A. Smith. A new approach of eliminating redundant association rules. In *DEXA*, pages 465–474, 2004.
3. Y. Bastide, N. Pasquier, R. Taouil, G. Stumme, and L. Lakhal. Mining minimal non-redundant association rules using frequent closed itemsets. In *CL '00: Proceedings of the First International Conference on Computational Logic*, pages 972–986, London, UK, 2000. Springer-Verlag.
4. R. J. Bayardo. Efficiently mining long pattern from databases. In *Proc. of ACM SIGMOD International Conference on Management Data*, pages 85–93, 1998.
5. G. Birkhoff. Lattice theory. *Am. Math. Soc. Colloquium Publications. AMS, Providence, 2 edition*, 25, 1948.

6. C. Carpineto and G. Romano. *Concept Data Analysis: Theory and Applications*. John Wiley and Sons, 2004.
7. B. A. Davey and H. A. Priestley. *Lattices Theory. Introduction to Lattices and Order*. Cambridge University Press. Fourth Edition, 1994.
8. J. Han, J. Pei, and Y. Yin. Mining frequent patterns without candidate generation. In *SIGMOD '00: Proceedings of the 2000 ACM SIGMOD international conference on Management of data*, pages 1–12, New York, NY, USA, 2000. ACM Press.
9. J. Hipp, A. Myka, R. Wirth, and U. Güntzer. A new algorithm for faster mining of generalized association rules. In *PKDD '98: Proceedings of the Second European Symposium on Principles of Data Mining and Knowledge Discovery*, pages 74–82, London, UK, 1998. Springer-Verlag.
10. S. Jaroszewicz and D. A. Simovici. Pruning redundant association rules using maximum entropy principle. In *PAKDD '02: Proceedings of the 6th Pacific-Asia Conference on Advances in Knowledge Discovery and Data Mining*, pages 135–147, London, UK, 2002. Springer-Verlag.
11. C. Lucchese, S. Orlando, and R. Perego. Fast and memory efficient mining of frequent closed itemsets. *IEEE Transactions on Knowledge and Data Engineering*, 18(1):21–36, 2006.
12. R. Mack, S. Mukherjea, A. Soffer, N. Uramoto, E. Brown, A. Coden, J. Cooper, A. Inokuchi, B. Iyer, Y. Mass, H. Matsuzawa, and L. Subramaniam. Text analytics for life science using the unstructured information management architecture. *IBM Systems Journal*, 43(3):490–515, 2004.
13. H. Mannila, H. Toivonen, and A. I. Verkamo. Efficient algorithms for discovering association rules. In U. M. Fayyad and R. Uthurusamy, editors, *AAAI Workshop on Knowledge Discovery in Databases (KDD-94)*, pages 181–192, Seattle, Washington, 1994. AAAI Press.
14. N. Pasquier, Y. Bastide, R. Taouil, and L. Lakhal. Pruning closed itemset lattices for association rules. In *Proc. of Journées Bases de Données Avancées*, pages 177–196, 1998.
15. N. Pasquier, Y. Bastide, R. Taouil, and L. Lakhal. Discovering frequent closed itemsets for association rules. In *ICDT '99: Proceeding of the 7th International Conference on Database Theory*, pages 398–416, London, UK, 1999. Springer-Verlag.
16. R. Srikant and R. Agrawal. Mining generalized association rules. In U. Dayal, P. M. D. Gray, and S. Nishio, editors, *VLDB'95, Proceedings of 21th International Conference on Very Large Data Bases*, pages 407–419. Morgan Kaufmann, 1995.
17. K. Sriphaew and T. Theeramunkong. Mining generalized closed frequent itemsets of generalized association rules. In *Proc. of Knowledge-Based Intelligent Information and Engineering Systems, 7th International Conference*, pages 476–484, 2003.
18. M. J. Zaki. New algorithms for fast discovery of association rules. In *Proc. of 3th International Conference on Knowledge Discovery in Databases*, pages 283–286, 1997.
19. M. J. Zaki. Generating non-redundant association rules. In *KDD '00: Proceedings of the sixth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 34–43, New York, NY, USA, 2000. ACM Press.

Musifind: A System for the Automatic Identification of Music Works

Nicola Orio

Department of Information Engineering – University of Padova
Via Gradenigo, 6/a, 35131, Padova, Italy
orio@dei.unipd.it

Abstract. This paper describes a methodology for the recognition of audio recordings. The recognition is based on an application of hidden Markov models (HMMs), which are automatically built from digital music scores. States of the HMMs are labeled by score events, and the transition and observation probabilities are directly computed from the information on the score. Six approaches to the recognition task have been tested on a set of audio excerpts. Tests showed that the methodology can achieve satisfactory results, at least for a supervised labeling of audio recordings.

1 Introduction and Related Work

Automatic music identification has a number of applications, that range from digital rights management (DRM), to automatic metadata extraction and music retrieval. A common approach to music identification is to extract, directly from a recording in digital format, its *audio fingerprint*, which is a unique set of features that allows for the identification of digital copies even in presence of noise, distortion, and compression. A fingerprint can be seen as a content-based signature that summarizes an audio recording. A comprehensive tutorial about audio fingerprinting techniques and applications can be found in [?]. Audio fingerprinting systems normally identify each particular recording of a given song, because – apart from obvious DRM requirements – it is assumed that users are normally interested to a specific recording of a given work.

There are some cases where the identification should be carried out without linking the process to a particular performance. For instance, identification of works performed in live concerts may not benefit from the fingerprints of other performances, because most of the acoustic parameters may be completely different. Moreover, users may be interested in discovering if there are alternative recordings of a music work they like.

This paper reports a methodology for automatic recognition of unknown recordings in audio format. The approach is based on Hidden Markov Models (HMMs), which model the unobservable process underlying the production of a music performance given a score. The approach can be applied to recognize tonal Western music and all the music genres where it can be assumed that a score is available and that the performance strictly adheres to it.

The recognition of tonal Western music can be carried out also with a direct match between two audio recordings [?], in particular using Dynamic Time Warping as an alternative to HMMs [?]. Yet, these approaches require that at least a recording is available for all the scores to be recognized, which may not be realistic assumption. On the other hand, there is a wide availability of files of tonal Western music in Midi format, which can be downloaded both by registered users from specialized Web sites and by casual users from personal homepages. The use of scores, rather than digital recordings of performances, reduces also the possibility of copyright issues in creating the collection of models for the recognition.

The automatic recognition of unknown audio recordings through statistical models shares some features with another relevant task, which consists in recognizing the positions along the score that correspond to the audio frames of a performance of that same score. In case of real time constraints, the task is usually defined *score following*. An example of the applications of statistical models to score following is presented in [?], while the application of HMMs to the task can be found in [?] and in [?]. The present research work builds upon previous research on score following, applied to a recognition task [?].

2 Score and Performance Modeling

The idea underlying the proposed approach is that a performance can be considered as the realization of a process that converts a music score into a sequence of audio features. The process is non deterministic because different performances correspond to the same score, depending on a number of parameters which are usually not known: instrumental acoustics, playing techniques, room reverberation, and so on. Nevertheless, given a score, it is possible to make some assumptions about the sequence of acoustic parameters corresponding to its performance and model them as a stochastic process. HMMs proved to be a powerful tool to model sequences in different application domains. Detailed introductions to the theory of HMMs can be found in [?] for applications to speech recognition and in [?] for applications to biological sequence analysis.

HMMs are stochastic finite-state automata, where transitions between states are ruled by probability functions. At each transition, the new state emits a random vector with a given probability density function. A HMM λ is completely defined by:

- a set of N states $Q = \{q_1, \dots, q_N\}$, in which the initial and final states are identified;
- a probability distribution for state transitions, that is the probability to go from state q_i to state q_j in a single step;
- a probability distribution for observations, that is the probability to observe the features r when in state q_j .

The process that generates the audio parameters extracted from a performance can be modeled with a HMM providing that: states are labeled with

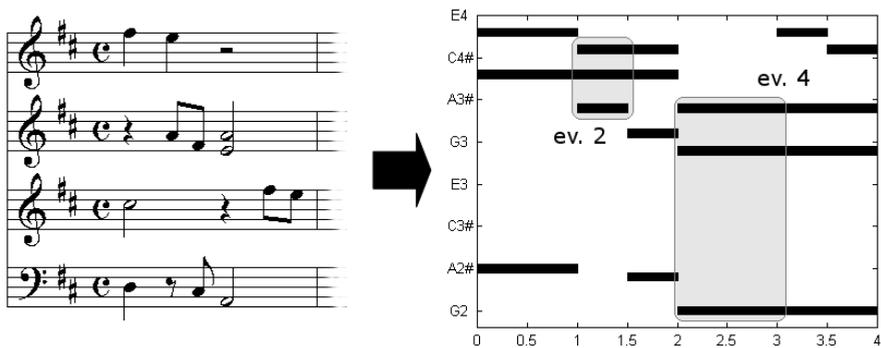


Fig. 1. A polyphonic score, in music notation (left) and in pianoroll (right); two events are highlighted on the pianoroll.

events in the score, transitions model the temporal evolution of the score, and observations are related to audio features that help distinguishing different events. The model is hidden because only the audio features can be observed and it is Markovian because transitions and observations are assumed to depend only on the actual state.

2.1 Modeling Score Events

The first step regards the creation of a set of HMM states which are labeled by events of a polyphonic score – a score is defined as polyphonic when multiple notes can be active at the same time, with attacks and decays at different positions on the time axis. A simple approach to state labeling consists in parsing the score to identify the times at which there is a change in polyphony. This may happen when one or more notes either start or stop playing, thus it is considered that a new event appears on the time axis when one or more notes are added to or deleted from the list of active notes. Events are used to label states of the HMM, as described in [?].

An example of score parsing is shown in Figure ??, where the same polyphonic score is represented in the usual music notation (on the left) and in a visual notation known as pianoroll (on the right), which represents each note as a line starting and ending at its onset and offset time, respectively. A new event is added to the list of score events every time there is a new onset or a new offset. Figure ??, which has 6 events, highlights event 2, which is due to a change in the number of the active notes, and event 4, which is due to a simultaneous attack of different notes.

Score events may have different durations, which can be inferred by the nominal note onsets, durations, and offsets given by the score. Yet the score gives only a rough estimate of the real duration, which may vary dramatically depending on the tempo chosen by the performer and by the presence of *rubato*. For this reason, event durations are modeled stochastically by the probabilities

of self and forward transitions of a chain of states with the same label, as shown in Figure ?? for events 3 and 4 of the score of Figure ?. If all the n states labeled by the same event have the same self-transition probability p , the probability of duration d is given by a negative binomial, according to equation

$$P(d) = \binom{d-1}{n-1} p^{d-n} (1-p)^n$$

from which $P(d)$ can be maximized on the expected event duration by choosing a value of n and computing p accordingly.

The modeling of the complete score can be carried out by simply connecting two subsequent events with a transition, as shown in Figure ?. In this case, the only assumption that is made on the performance is that the musicians will play the events in the same order written in the score, which is a reasonable assumption at least for tonal Western music. The overall topology of the HMM is strictly left-to-right, with self-transitions used to model events duration and forward transitions used to model the evolution in time of the score. From Figure ? it can be noted that every state has usually two transitions with non zero probability – a self and a forward transition – with the exception of the last state of a chain that may have three transitions – a self transition, a transition to the state modeling an articulatory rest, and a transition to the first state of the next event.

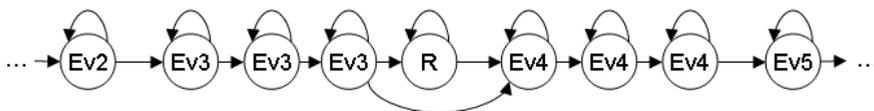


Fig. 2. Excerpt of the HMM built from a score; the state labeled with letter **R** refers to an articulatory rest.

2.2 Observations

The choice of features to be used as observations is crucial, because they have to be robust to distortions of the recording, timbre variations of musical instruments, room acoustics, and possible digital compression. Since states are labeled with score events, it is possible to compute a probability density function of the features that are relevant for each event. A simple and general assumption is that the Fourier transform of observations should have a number of peaks corresponding to the expected harmonics of the signal [?].

For example, the two events highlighted in Figure ? are both made of three notes with a known fundamental frequency. Thus, it is expected that the frequency bands around the three fundamental frequencies and their first integer

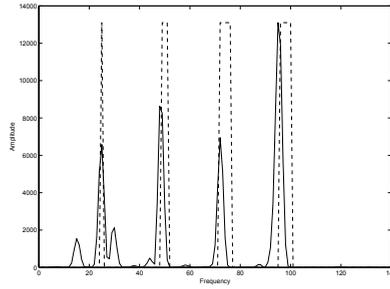


Fig. 3. Example of a bank of bandpass filters.

multiples will carry most of the overall energy of the signal. This computation has to be made for each event in the score, and it can be efficiently carried out with a single Fourier transform and by summing the energy of frequency bins around the expected peaks. This approach can be seen as filtering of the signal with a bank of bandpass filters, where each filter is centered on the expected harmonics. Figure ?? exemplifies this approach.

The energy of the signal after filtering is then normalized through division by the overall energy of the signal. This way, for each audio frame and for each event we obtain a value between 0 when there is no energy in the expected bands, and 1 when there is no energy outside the expected bands. Thus, the observations of state n for frame r are then computed according to equation

$$o(r, n) = 1 - \frac{\sum_i F_n(i) S_r(i)}{\sum_i S_r(i)}$$

where F_n are the spectral bands for state n and S_r is the Fourier transform of the signal frame r . The probabilities for state n to emit observation o are modeled with an exponential probability density function [?], computed through the equation

$$P(o|n) = \frac{e^{-o/\mu}}{\mu}$$

where μ has been experimentally determined.

Another relevant feature is the overall energy of each frame, which allows us to cope with silence where the harmonic content is almost random. The presence of silence can be due either to explicit rests in the score, or to articulatory rests, which can be found only when all the notes of an event have an attack at the same time. Figure ?? shows an excerpt of the HMM built from the score reported in Figure ??, where the presence of an articulatory rest before event 4 is shown. The overall energy may vary dramatically depending on the audio quality, the signal to noise ratio, and the sound level of the recording. For this reason the probability density functions are computed from the audio recording using the classical E-M algorithm.

3 Audio Recognition

Recognition is probably the application of HMMs that is most often described in the literature. The recognition problem may be stated as follows:

- given an unknown *audio recording*, described by a sequence of audio features $R = \{r(1), \dots, r(T)\}$ and given a set of *competing models* λ_i
- find the model that more likely generates R

The definition does not impose a particular evolution of models λ_i , that is the path across the N states that corresponds to the generation of R . This allows us to define a number of criteria for solving the recognition problem, depending on different constraints applied to the evolutions of the states of λ_i . Three different approaches are proposed, whose names are derived from the notation proposed in a classical tutorial on HMMs [?].

3.1 Approach α

The most common approach to HMM-based recognition, is to compute the probability that λ_i generates R regardless of the state sequence. This can be expressed by equation

$$\lambda_\alpha = \operatorname{argmax}_i P(R|\lambda_i)$$

where the conditional probability is computed over all the possible state sequences of a model. The probability can be computed efficiently using the *forward probabilities*, also known as *alpha probabilities*. Even if approach α is the common practise for speech recognition, it may be argued that also paths that have no relationship with the actual performance (e.g., the first state that continuously performs self-transitions) give a positive contribution to the final probability. These consideration lead us to the next two approaches.

3.2 Approaches δ and γ

Another typical HMM problem is finding the most probable path across the states, given a model and a sequence of observations. A widely used algorithm is Viterbi decoding, which computes a path that is *globally* optimal according to equation

$$q^\delta = \operatorname{argmax}_q P(q|R, \lambda_i)$$

Alternatively, a *locally* optimal path [?] can be computed, according to equation

$$\begin{aligned} \bar{q}(t) &= \operatorname{argmax}_q P(q(t)|R, \lambda_i) \\ q^\gamma &= \{\bar{q}(1), \bar{q}(2), \dots, \bar{q}(T)\} \end{aligned}$$

Both global and local optimal paths can be used to carry out a recognition task, for finding the model that more likely generates R while state evolution is constrained. This consideration leads to equations

$$\begin{aligned}\lambda_\delta &= \operatorname{argmax}_i P(R|q^\delta, \lambda_i) \\ \lambda_\gamma &= \operatorname{argmax}_i P(R|q^\gamma, \lambda_i)\end{aligned}$$

that show how the probability of R is conditioned both by the model λ_i and by the state sequence of the global or optimal paths.

3.3 Additional Constraints

For each of the approaches, it is likely that the paths across states of the model that correctly correspond to R will end on the final state q_N of the HMM. This knowledge can be introduced in the recognition task as an additional constraint on the paths that are used to compute the conditional probabilities, obtaining three alternative approaches.

- Approach $\hat{\alpha}$: only states sequences that end on q_N are used to compute $P(R|\lambda_i)$.
- Approach $\hat{\delta}$: Viterbi decoding is forced to stop in q_N .
- Approach $\hat{\gamma}$: the local optimal path has q_N as the final state.

These additional approaches assume that the recording is a complete performance of the score used to compute the model. This assumption may not hold when recognition is based on short audio excerpts, because the number of score frames that may correspond to the performance is not known in advance. Nevertheless, these approaches have been evaluated to have a comparison with the unconstrained ones.

3.4 Complexity

All the presented approaches allows the computation of the probabilities using a dynamic programming approach. In particular, it is known in the literature that each of the approaches requires $\mathbf{O}(DTN^2)$ time, where D is the number of competing models, T is the duration of the audio sequence, and N is the average number of states of the competing HMMs. Considering that, as described in Section ??, each state may perform a maximum of three transitions, it can be shown that complexity becomes $\mathbf{O}(DTN)$.

In order to increase efficiency, the length of the unknown sequence should be small, that is the method should give good results also with short audio excerpts. Moreover, also the number of states N in each model should be limited, which can be obtained using a small number of states for each score event.

Table 1. Recognition rates for the different approaches, reporting the percentage by which the correct score was ranked first ($= 1$) or either first or second (≤ 2).

	α	δ	γ	$\dot{\alpha}$	$\dot{\delta}$	$\dot{\gamma}$
$= 1$	84	84	92	96	96	96
≤ 2	96	96	96	96	96	96

4 Experimental Evaluation

A set of experiments has been carried out to evaluate the methodology with real acoustic data from original recordings, taken from the personal collection of the author. Tonal Western music repertoire has been used as a testbed because of the wide availability of scores in digital format. Scores have been downloaded from Web sites providing files in MIDI format, with the permission of the administrators. As already mentioned, the approach can be applied to all the music genres where performers play accordingly to a predefined score.

The audio performances to be recognized were 25 short excerpts of works of well known composers, from Bach to Debussy. Excerpts were played with different instruments, including polyphonic piano, string ensembles, and complete orchestras. Recordings were the opening bars of the selected works, with an average length of 8.3 seconds, from 4.2 to 12.3 seconds. A preliminary analysis showed that recognition performances may improve with longer excerpts, yet one of the aims of this work was to test the robustness to short examples that reduce the computational cost. The audio files were all mono recordings, with a sampling rate of 44.1 kHz, and they have been divided in frames of 2048 samples, applying a hamming window, with an overlap of 1024 samples. With these parameters, a new observation is computed every 23.2 milliseconds.

The models used for recognition have been computed using 100 scores, including the ones corresponding to the performances to be recognized. These latter 25 scores have been manually edited in order to have the same opening bars of the corresponding audio excerpts, obtaining an average number of 26.3 events, with a standard deviation of 16.8. The additional scores have been chosen to have a similar content and a comparable number of events. Testing the robustness to differences between scores and performances is left to future work.

4.1 Recognition Rates

The 25 audio excerpts have been considered as unknown sequences to be recognized, using the alternative approaches presented in Section ???. Table ?? reports the recognition rates at two thresholds, that is the percentages at which the correct score was ranked the most similar one ($= 1$), and when it was ranked within the first two (≤ 2), scores. The results, presented in Table ?? have been obtained modeling each event with a chain of four states; observations were the energy in the first four harmonics of the active notes of the event.

With this experimental setup the average recognition rate was quite high for all the approaches. As expected, the best performances have been obtained with the introduction of the additional constraints on the final states of the HMM, as described in Section ???. With all these approaches the correct score either was retrieved at the first position, or it was retrieved at a quite high ranking – above the 15th position. These results are affected by the presence of a single performance that turned out to be very difficult to recognize. The performance was a polyrhythmic piano piece by Claude Debussy, *Primiere Arabesque*, where the left hand plays an even tempo and the right hand an odd tempo. This characteristic is translated in a number of short events, due to the fact that new events appear in the score each time a new note is played either by the left or by the right hand. A closer analysis showed that the pianist was not playing exactly the score, which explains the low results for this particular piece. Yet, human listeners are perfectly able to recognize the song.

Among the approaches that do not take into account additional constraints, γ is the one that gave the best results, even if by a small percentage. This result may seem surprising because, among the approaches, α is the most natural one for a recognition task. Yet, the combination of alignment and recognition, at least for this particular collection, gave better recognition rate. Moreover, results showed that local alignment exploited in γ is to be preferred to global alignment exploited in δ . An inspection of the excerpts that had a lower rank for δ than for γ showed that the former approach is more sensible to local mismatches between states of the HMM correct model and the audio features, because they affect also surrounding states.

It is worth noting that audio identification based on fingerprinting techniques is considered a resolved research problem, because of its very high identification rate and its robustness to distortion and compression. Commercial audio fingerprinting systems are already available since years. On the other hand, the more general problem of music identification based on score modeling is still a new research area. For this reason, it is difficult to compare the results with other approaches, because of the lack of a common benchmark for the evaluation of these approaches. For example, the work reported in [?], based on the chroma representation, has been tested on a collection of pop songs with an identification rate of 49%, with more than 22% of recordings that have been ranked above the tenth position. Another related approach, presented in [?], is based on harmonic descriptors and is reported to achieve an average precision of 55% on a collection of pop and jazz songs. It would have been interesting to compare these results with the work reported in [?] that, even if based on audio to audio matching, has been applied to tonal Western music. Unfortunately, [?] presents only a qualitative analysis of the results, from which it just seems that identification rate is below 70%, but the real value is not given.

4.2 Robustness to the Choice of the Parameters

Additional evaluation has been carried out to test the robustness to HMMs parameters. In particular, the number of states that model each event has been

Table 2. Recognition rate (RR) and average precision (AP) varying the number N of HMMs states modeling an event.

	$N = 3$	$N = 4$	$N = 5$
	RR-AP	RR-AP	RR-AP
α	80%-88%	84%-90%	84%-90%
δ	80%-88%	84%-90%	80%-88%
γ	84%-90%	88%-92%	80%-88%

varied to see if the computational complexity can be reduced using fewer states without affecting recognition rate. Moreover, the number of states for each event has been also augmented in order to test if recognition could be further improved. Table ?? summarizes the results when the number of states has been varied, maintaining the same parameters of the audio analysis. As representative measures, Table ?? reports the *recognition rate* (RR) – the percentage of time the excerpt has been assigned the highest rank – and *average precision* (AP) which is another common measure for this kind of task.

The choice of $N = 4$ states to model events is the one that gave the best results when compared with shorter and longer chains of states. This result may depend on the analysis parameters, considering that the length of the analysis window and amount of overlap gave a constant frame rate of about 23 milliseconds, and also on the particular collection. Shorter chains of states may not be able to precisely model different durations, while longer chains may fail in modeling very short events. The choice of 4 states seems to be a good tradeoff between precision on duration modeling and robustness to short events.

Another evaluation regarded the choice of the number of harmonics that are used to compute the observations. Extensive tests, using from one to eight harmonics, showed that the best results are obtained using four harmonics. A further decrease in the number of harmonics, which may be advisable because it may reduce the computational complexity, gave a drop in performances both in terms of recognition rate and average precision. Finally, also the effect of the analysis parameters have been tested, using different combinations of window length and overlap between windows. Results show that the choice of parameters introduced at the beginning of this section – window length of 2048 samples with an overlap of 1024 samples – is the one that gave the best results.

4.3 Computing Time

All tests have been carried out on a laptop computer, with an Intel CPU Centrino at 2.0 GHz, with 1 GB of RAM, mounting a Microsoft Windows XP operative system. All the routines have been written in ANSI C++, with no particular optimization. With this quite low level equipment, the average time needed to recognize an audio excerpt was about 21 seconds, meaning that the recognition task for each of the scores was about 0.2 seconds. These results may show that

the recognition task becomes feasible, at least off-line, also when large collections of competing scores are used.

4.4 Recognition of Monophonic Performances

Recognition of monophonic performances is an easier task: the spectral content is composed by a single harmonic sequence, and the computation of HMMs observations can be based on a single bank of equally spaced filters; the presence of a single note prevent also for the presence of interference between different sound sources; the creation of the state chain of the HMM is straightforward, because the number of events is equal to the number of notes and rests in the score. For these reasons, the experimental tests that have been presented in the previous sections do not include any monophonic score, which could have biased the final results. For the sake of completeness, a number of tests have been carried out on a collection of purely monophonic scores, recognizing 28 short excerpts out of 87 different scores. Results for monophonic performances have been really satisfactory, because all the three approaches, either with and without the additional constraints, gave a recognition rate of 100%.

5 Conclusions

A methodology for automatic audio recognition has been proposed, based on HMMs. Six approaches to compute the conditional probability of observing a performance given the model of a score have been tested on a collection of digital scores and acoustic performances. Experimental results showed that, at least for tonal Western music, it is possible to achieve a good recognition rate, which may allow for the development of a supervised system for labeling unknown recordings.

A possible application scenario of the proposed methodology may regard the automatic annotation of recordings of broadcasted concerts of tonal Western music. To this end, a graphical interface have been developed on the top of the recognition routines, in order to provide users with a tool to annotate the content of music collections with the additional information contained in the scores. For each unknown audio recording, users are presented with a list of candidate scores in MIDI format, which they may be played in order to verify if they have been correctly recognized. The task of classifying the content of a digital audio archive becomes almost trivial also for non musicologist. The tool can also be useful for organizing the content of personal music collections, which is the reason why the methodology has been tested using a simple hardware and software configuration.

One of the current limitations is the need of a collection of scores in symbolic format in order to perform recognition. It can be noted that are currently available on the Web thousands of files in MIDI format, which in the case of tonal Western music are often well organized and categorized. Most of the Web sites specialized in tonal Western music allow downloading of the complete collections,

at least for subscribers, and hence for this particular repertoire the development of a collection of scores does not represent a real problem. Yet there may be more difficulties if the approach is extended to other music genres based on a score representation, such as Opera or traditional folk music.

References

1. P. Cano, E. Batlle, T. Kalker, and J. Haitsma. A review of audio fingerprinting. *Journal of VLSI Signal Processing*, 41:271–284, 2005.
2. P. Cano, A. Loscos, and J. Bonada. Score-performance matching using hmms. In *Proceedings of the International Computer Music Conference*, pages 441–444, 1999.
3. S. Dixon and G. Widmer. MATCH: a music alignment tool chest. In *Proceedings of the International Conference of Music Information Retrieval*, pages 492–497, 2005.
4. R. Durbin, S. Eddy, A. Krogh, and G. Mitchison. *Biological Sequence Analysis*. Cambridge University Press, Cambridge, UK, 2000.
5. E. Gómez and P. Herrera. The song remains the same: Identifying versions of the same piece using tonal descriptors. In *Proceedings of the International Conference on Music Information Retrieval*, pages 180–185, 2006.
6. L. Grubb and R.B. Dannenberg. A stochastic method of tracking a vocal performer. In *Proceedings of the International Computer Music Conference*, pages 301–308, 1997.
7. N. Hu, R.B. Dannenberg, and G. Tzanetakis. Polyphonic audio matching and alignment for music retrieval. In *Proceedings of the IEEE Workshop on Applications of Signal Processing to Audio and Acoustics*, pages 185–188, 2003.
8. M. Müller, F. Kurth, and M. Clausen. Audio matching via chroma-based statistical features. In *Proceedings of the International Conference of Music Information Retrieval*, pages 288–295, 2005.
9. N. Orio. An automatic accompanist based on hidden markov models. In F. Esposito, editor, *Advances in Artificial Intelligence. Proc. of 7th Congress of the Italian Association for Artificial Intelligence*, volume LNAI 2175, Lecture Notes in Artificial Intelligence, pages 64–70, Berlin, DE, 2001. Springer.
10. N. Orio and D. Schwarz. Alignment of monophonic and polyphonic music to a score. In *Proceedings of the International Computer Music Conference*, pages 129–132, 2001.
11. N. Orio and M. Sisti Sette. A HMM-based pitch tracker for audio queries. In *Proceedings of the International Conference on Music Information Retrieval*, pages 249–250, 2003.
12. L.R. Rabiner and B.-H. Juang. *Fundamentals of Speech Recognition*. Prentice-Hall, Englewood Cliffs, NJ, 1993.
13. C. Raphael. Automatic segmentation of acoustic musical signals using hidden markov models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 21(4):360–370, 1999.
14. D. Schwarz, N. Orio, and N. Schnell. Robust polyphonic MIDI score following with hidden markov models. In *Proceedings of the International Computer Music Conference*, pages 442–445, 2004.

Trajectory Data Warehouses: Design Issues and Use Cases

S. Orlando¹, R. Orsini¹, A. Raffaetà¹, A. Roncato¹, C. Silvestri²

¹ Dipartimento di Informatica - Università Ca' Foscari Venezia

² Dipartimento di Informatica e Comunicazione - Università di Milano

Abstract. In this paper we discuss how data warehousing technology can be used to store aggregate information about trajectories and to perform OLAP operations over them. To this end, we define a data cube with spatial and temporal dimensions, discretized according to a hierarchy of regular grids. Moreover, we analyse some measures of interest related to trajectories, such as the number of trajectories starting from a cell, the distance covered by the trajectories in a cell, the average and maximum speed and the average acceleration of the trajectories in the cell. We provide some algorithms to compute and load these measures in the base cells, starting from a stream of trajectory observations. Such stored values are used, along with suitable aggregate functions, to compute the roll-up operations. Finally, we envision some use cases that would benefit from such a framework, in particular in the domain of supervision systems to monitor road traffic (or movements of individuals) in a given geographical area.

1 Introduction

The widespread diffusion of modern technologies such as low-cost sensors, wireless, ubiquitous and location-aware mobile devices, allows for collecting overwhelming amounts of data about trajectories of moving objects. Such data are usually produced at different rates, and arrive in streams in an unpredictable and unbounded way. This opens new opportunities for monitoring and decision making applications in a variety of domains, such as traffic control management and Location-Based Services. However, for these applications to become reality, new technical advances in spatial information management are needed. Typically, in fact, analytical and reasoning processes for a large set of data require, as a starting point, their organisation in repositories, or data warehouses (DWs), where they can be extracted with powerful operators and further elaborated by means of sophisticated algorithms.

The tools traditionally used to manage geographical data are spatial databases, GISs [16, 14, 20] and spatial DWs [8, 15, 11, 17]. However, spatial data warehousing is in its infancy. The pioneering work of Han et al. [8] introduces a spatial data cube model which consists of both spatial and non-spatial dimensions and measures. Additionally, it analyses how OLAP operations can be performed in such a spatial data cube. Recently, several authors have proposed

data models for spatial DWs at a conceptual level (e.g., [13, 3]). Unfortunately, none of these approaches deal with objects moving continuously in time.

A related research issue, which is gaining an increasing interest in the last years and is relevant to the development of DWs for spatio-temporal data, concerns the specification and efficient implementation of the operators for spatio-temporal aggregation. A first comprehensive classification and formalisation of spatio-temporal aggregate functions is presented in [9] whereas in [12, 19] techniques for the computation of aggregate queries are developed based on the combined use of specialised indexes and materialisation of aggregate measures.

In this paper we propose the definition of a simple DW model for storing aggregates about trajectories, implementable using off-the-shelf DW systems. We start by defining a data cube with spatial and temporal dimensions, discretized according to a hierarchy of regular grids. The model abstracts from the identifiers of the objects in favour of aggregate information concerning global properties of a set of moving objects, such as the distance travelled by these objects inside an area, or their average speed or acceleration. There are good reasons for storing only this aggregate information: in some cases personal data should not be stored due to legal or privacy issues; individual data may be irrelevant or unavailable; and individual data may be highly volatile and involve huge space requirements. We analyse the loading phase of our DW, which has to deal with the fact that trajectory observations arrive in streams. In order to avoid the use of an unbounded amount of memory buffer, where to store trajectories, we suggest some methods for feeding the DW before having received the complete trajectory. We outline two algorithms which can be used to compute a set of measures of interest for trajectories. The first one applies to measures which can be computed directly by using only the trajectory observations. The second one applies to measures whose calculation exploits additional points obtained by reconstructing the trajectory through interpolation. Finally, we discuss how the aggregate information stored in the DW could be used to discover properties of real world entities, either directly by means of standard OLAP operators, or using them as an input for subsequent analyses, e.g., for data mining purposes.

The rest of the paper is organised as follows. Section 2 introduces the issues related to the representation of the trajectories and we discuss the DW model which is later used to store measures concerning trajectories. Section 3 discusses several issues concerning the loading phase of the DW and Section 4 describes two algorithms to compute the measures stored in the base cells of the data cube. In Section 5 we discuss some possible use cases of the trajectory DW. Finally, Section 6 draws some conclusion and gives possible directions for future work.

2 The DW model for trajectories

In real-world applications the movement of a spatio-temporal object, i.e., its *trajectory*, is often given by means of a finite set of *observations*. This is a finite subset of points taken from the actual continuous trajectory. Such a finite set of observations is called a *sampling*. For example, Fig. 1(a) shows a trajectory of a

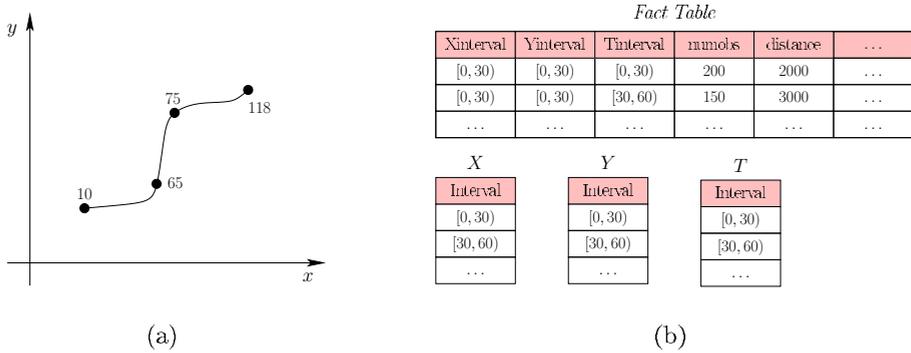


Fig. 1. (a) A 2D trajectory with a sampling and (b) an example of Trajectory DW.

moving object in a two dimensional (2D) space and a possible sampling of such a trajectory. Each point is annotated with the corresponding time-stamp. It is reasonable to expect that observations are taken at irregular rates for each object, and that there is no temporal alignment between the observations of different objects. In the following an observation will be a tuple (id, x, y, t) , meaning that the object id is at location (x, y) at time t .

In many situations, e.g., when one is interested in computing the cumulative number of trajectories in a given area, an (approximate) reconstruction of each trajectory from its sampling is needed. Among the several possible solutions, in this paper we will use *linear local interpolation*, i.e., objects are assumed to move straight between the observed points with constant speed. The linear (local) interpolation seems to be a quite standard approach to the problem and yields a good trade-off between flexibility and simplicity.

Starting from a stream of samplings, we want to investigate if DW technologies [2, 6], and thus the concept of multidimensional and multilevel data model, can be used to store and compute specific aggregate *measures* regarding trajectories. First, we define a DW model by means of a star schema, as simple and generic as possible. The *facts* of our DW are the trajectories and the dimensions of analysis are: the spatial dimensions X, Y ranging over spatial intervals, and the temporal dimension T ranging over temporal intervals. The related tables in a star schema (see Fig. 1(b)) store the details of these dimensions. The fact table stores the set of the foreign keys to the dimension tables and the set of measures.

Some examples of measures that are of interest for trajectories are listed in Table 1. The complexity, the memory space and the aggregate functions necessary to load them starting from a stream of observations, can vary from simple computation with no buffer requirements to very expensive and approximate functions, as we will discuss in the next sections.

Table 1. Some numeric measures.

$m1$	$nmobs$	Number of observations in the cell.
$m2$	$trajinit$	Number of trajectories starting in the cell.
$m3$	$presence$	Number of trajectories in the cell.
$m4$	$distance$	Total distance covered by trajectories in the cell.
$m5$	$speed$	Average speed of trajectories in the cell.
$m6$	v_{max}	Maximum speed of trajectories in the cell.
$m7$	acc	Average acceleration of trajectories in the cell.

For each dimensional attribute it is useful to introduce a *concept hierarchy*, that allows data to be handled at varying levels of abstraction by performing roll-up and drill-down OLAP operations. In particular, since we are interested in associating concept hierarchies with spatial and temporal attributes, this can be naturally carried out by discretizing the corresponding values, resulting in a set-grouping hierarchy. A partial order can thus be defined among groups of values, e.g., $[0, 30) < [0, 60)$ and $[30, 60) < [0, 60)$.

We can build the spatial data cube [5] as the lattice of cuboids, where the lowest one (*base cuboid*) references all the dimensions at the primitive abstraction level, while the others are obtained by summarising on different subsets of the dimensions, and/or at different abstraction levels along the concept hierarchy. In order to denote a component of the base cuboid we will use the term *base cell*, while we will simply use *cell* for a component of a generic cuboid.

In order to summarise information contained in the base cells, Gray et al. [5] categorise the aggregate functions into three classes based on the space complexity needed for computing a super-aggregate starting from a set of sub-aggregates already provided, e.g., the sub-aggregates associated with the base cells of the DW. The classes are the following:

1. *Distributive*. The super-aggregates can be computed from the sub-aggregates.
2. *Algebraic*. The super-aggregates can be computed from the sub-aggregates together with a *finite* set of auxiliary measures.
3. *Holistic*. The super-aggregates cannot be computed from sub-aggregates, not even using any finite number of auxiliary measures.

Consider the measures in Table 1. The super-aggregates for $m1$, $m2$, $m4$ and $m6$ are simple to compute because the corresponding aggregate functions are *distributive*. In fact once the base cells have been loaded with the exact measure, for $m1$, $m2$ and $m4$ we can accumulate such measures by using the function *sum* whereas for $m6$ we can apply the function *max*. The super-aggregate for $m5$ and $m7$ are algebraic: we need some auxiliary measures in order to compute the aggregate functions. For the average speed of trajectories in a cell, a pair $\langle distance, time \rangle$ where *distance* is the measure $m4$ and *time* is the total time spent by trajectories in the cell must be considered. For a cell C arising as the union of adjacent cells, the cumulative function performs a component-wise addition, thus producing a pair $\langle distance_f, time_f \rangle$. Then the average speed in C is given by $distance_f / time_f$. In a similar way, to compute the average acceleration, the sum of the variations of the speed in the cell and the total time spent by

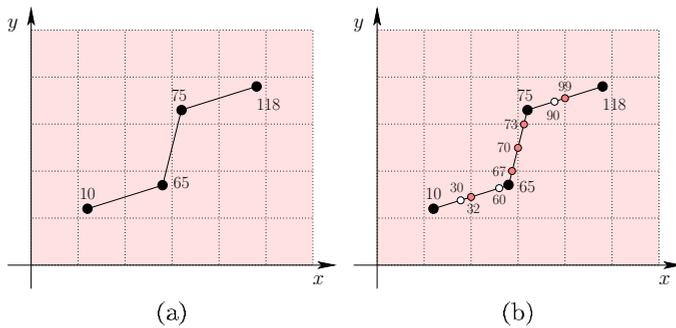


Fig. 2. (a) Linear interpolation of a 2D trajectory, and (b) the interpolated trajectory with additional points matching the spatial and temporal minimum granularity.

the trajectories in the cell are required as auxiliary measures. The aggregate function for $m3$ is holistic since it is a sort of `COUNT_DISTINCT()` aggregate. A thorough discussion on the computation of the measure *presence* can be found in [1] where we provided an algebraic function which approximates in a very accurate way the number of distinct trajectories in a cell.

3 Feeding the Data Warehouse

In this section we deal with the problem of feeding the DW, i.e., the base cells of our base cuboid, with suitable sub-aggregate measures, from which the aggregate functions compute the super-aggregates.

We recall that trajectory observations arrive in streams at different rates, in an unpredictable and unbounded way. In order to limit the amount of *buffer memory* needed, it is essential to store information only about *active*, i.e., not ended trajectories. In our simple model of trajectory sampling, since we do not have an end-mark associated with the last observation of a given trajectory, the system module that is responsible for feeding data could decide to consider a trajectory as *ended* when for a long time interval no further observation for the object has been received.

As mentioned in Section 2, almost all the measures of interest, e.g., measures $m3$ to $m7$ in Table 1, require to reconstruct the trajectory from its sampling. We recall that in this paper we use *linear local interpolation* in order to infer further spatio-temporal locations of intermediate points, occurring between two known trajectory observations. The goal is to guarantee the correct update of a given measure in our DW.

Given a set of observations for a trajectory, as depicted in Fig. 1(a), a possible reconstructed trajectory using linear interpolation is shown in Fig. 2(a), where we also illustrate the discretized 2D space. If we updated the DW on the basis of each single observation, the four observations of our example contribute to the *measures* (M_1, \dots, M_k) of the base cells illustrated in Fig. 3(a).

From this simple example two main issues arise:

Time label	X	Y	T	M_1	...	M_k
10	[30,60)	[30,60)	[0,30)
65	[60,90)	[30,60)	[60,90)
75	[90,120)	[90,120)	[60,90)
118	[120,150)	[90,120)	[60,120)

(t_i, t_{i+1})	X	Y	T	M_1	...	M_k
(10,30)	[30,60)	[30,60)	[0,30)
(30,32)	[30,60)	[30,60)	[30,60)
(32,60)	[60,90)	[30,60)	[30,60)
(60,65) (65,67)	[60,90)	[30,60)	[60,90)
(67,70)	[60,90)	[60,90)	[60,90)
(70,73)	[90,120)	[60,90)	[60,90)
(73,75) (75,90)	[90,120)	[90,120)	[60,90)
(90,99)	[90,120)	[90,120)	[90,120)
(99,118)	[120,150)	[90,120)	[90,120)

(a)
(b)

Fig. 3. A fact table for a Trajectory DW fed with (a) observations, and (b) the segments composing the interpolated trajectory.

- the identifier of the object is not stored in the base cell;
- some base cells are traversed by the trajectories but, since no observation falls in them, they do not appear in the Fact Table.

The lack of the identifier becomes a problem for the computation of the measures requiring to count the number of *distinct* elements satisfying a property, as it happens for the measure *presence*. This is a well-known problem, out of the scope of this paper, and a solution can be found in [1].

Concerning the second issue, we propose to consider additional interpolated points for each cell traversed by a trajectory. These points correspond to the intersections between the trajectory and the border of the spatio-temporal cell. Figure 2(b) shows the resulting interpolated points as white and gray circles. The white points, with temporal labels 30, 60, and 90, correspond to cross points of a temporal border of some 3D cell. The gray points, labelled with 32, 67, 70, 73, and 99, correspond to the cross points of the spatial borders of some 3D cell, or, equivalently, the cross points of the spatial 2D squares shown in Fig. 2(b). All these additional points do not cause a space explosion, since they are generated, considered on-the-fly and finally thrown away during the stream processing.

Thanks to these additional interpolated points, we have further 3D base cells in which we can now store significant measures associated with the trajectory of the given object. In particular, by construction, the new points subdivide the interpolated trajectory into small segments, each one completely included in some 3D base cell. These segments are used to update the cell measures. In Fig. 3(b) we show the sequence of edges composing the interpolated trajectory of Fig. 2(b), and the base cell which the edge belongs to. For the sake of simplicity, each edge is identified by a pair of timestamps (t_i, t_{i+1}) , associated with the starting and ending points.

4 Algorithms for loading the fact table

Following the line suggested in the previous section, it is reasonable to distinguish two kinds of algorithms to load the Fact Table: one is used to compute measures which do not require interpolation whereas the second one, which is

more complex, uses linear interpolation in order to add border points which allow a more precise computation of the measures. We assume that a cell C is associated with a tuple consisting of the following components: $C.numobs$, $C.trajinit$, $C.v_{max}$, $C.distance$, $C.time$, $C.speed$, $C.\Delta v$, $C.acc$. It is worth noticing that besides the measures already introduced, the tuple contains the total time spent by the trajectories in the cell ($C.time$) and the sum of the variations of speed of the trajectories in the cell ($C.\Delta v$), which are auxiliary measures required for the roll-up operation, as discussed in Section 2.

Schema of an algorithm for measures computed without interpolation

Input: Stream ST of observations (obs) in the form (id, x, y, t) .

Ouput: Fact Table FT

```

 $FT \leftarrow \emptyset$ 
 $buffer \leftarrow \emptyset$ 
repeat forever
   $obs \leftarrow getNext(ST)$ 
   $C_{cur} \leftarrow findCell(obs.x, obs.y, obs.t)$ 
  if ( $C_{cur} \notin FT$ )
     $insert(C_{cur}, FT)$ 
     $initMeasures(C_{cur}, obs, buffer)$ 
  else
     $updateMeasures(C_{cur}, obs, buffer)$ 

 $initMeasures(Cell, obs, buffer)$        $updateMeasures(Cell, obs, buffer)$ 
 $Cell.numobs \leftarrow 1$                $Cell.numobs \leftarrow Cell.numobs + 1$ 
if ( $buffer[obs.id] = empty$ )         if ( $buffer[obs.id] = empty$ )
   $Cell.trajinit \leftarrow 1$            $Cell.trajinit \leftarrow Cell.trajinit + 1$ 
 $buffer[obs.id] \leftarrow obs.t$        $buffer[obs.id] \leftarrow obs.t$ 

```

In order to compute measures $m1$ and $m2$ we do not need interpolation. In fact measure $m1$ requires only the observations and no auxiliary storage. Instead, in the case of measure $m2$, in order to understand whether an incoming observation starts a new trajectory, we store in the buffer, for each active trajectory, the identifier and the time-stamp of the last processed point.³ As discussed in the previous section, using the time-stamp, we can decide if a trajectory must be considered as ended. In this case the corresponding data in the buffer can be removed, thus freeing the space for storing new incoming observations. This procedure is not detailed in the pseudo-code for the sake of readability.

The algorithm consists of an infinite loop which processes the observations arriving in streaming. At each step, the function *getNext* gets the next observation obs in the stream and we determine the base cell C_{cur} which obs belongs to (function *findCell*). Then we check whether the base cell C_{cur} is in the Fact Table FT . In this case we correspondingly update the measures (function *updateMeasures*), otherwise we insert the cell in the Fact Table and we initialise the corresponding tuple (function *initMeasures*).

³ If the first observation of a trajectory were marked then the buffer would not be necessary.

Schema of an algorithm for measures requiring interpolation

Input: Stream ST of observations (obs) in the form (id, x, y, t) .

Output: Fact Table FT

```

 $FT \leftarrow \emptyset$ 
 $buffer \leftarrow \emptyset$ 
repeat forever
   $obs \leftarrow \text{getNext}(ST)$ 
   $C_{cur} \leftarrow \text{findCell}(obs.x, obs.y, obs.t)$ 
  if ( $C_{cur} \notin FT$ )
     $\text{insert}(C_{cur}, FT)$ 
     $\text{initMeasures}(C_{cur})$ 
  if ( $obs.id \notin buffer$ )
     $\text{insert}(obs.id, buffer)$ 
     $\text{initBuffer}(obs, C_{cur}, buffer)$ 
  else
     $point_{pred} \leftarrow \text{extractPoint}(buffer[obs.id])$ 
     $C_{pred} \leftarrow \text{extractCell}(buffer[obs.id])$ 
    if ( $C_{pred} \neq C_{cur}$ )
       $Q \leftarrow \text{interp.NewPoints}(point_{pred}, obs)$ 
      repeat
         $p \leftarrow \text{extract}(Q)$ 
         $C_p \leftarrow \text{findCell}(p.x, p.y, p.t)$ 
        if ( $C_p \notin FT$ )
           $\text{initMeasures}(C_p)$ 
           $\text{insert}(C_p, FT)$ 
         $\text{updateMeasures\&Buffer}(C_{pred}, point_{pred}, p, obs.id, buffer)$ 
         $C_{pred} \leftarrow C_p$ 
         $point_{pred} \leftarrow p$ 
      until  $Q = \emptyset$ 
     $\text{updateMeasures\&Buffer}(C_{pred}, point_{pred}, (obs.x, obs.y, obs.t), obs.id, buffer)$ 

 $\text{initMeasures}(Cell)$ 
   $Cell.v_{max} \leftarrow 0$ 
   $Cell.distance \leftarrow 0$ 
   $Cell.time \leftarrow 0$ 
   $Cell.speed \leftarrow 0$ 
   $Cell.\Delta v \leftarrow 0$ 
   $Cell.acc \leftarrow 0$ 

 $\text{initBuffer}(Cell, obs, buffer)$ 
   $buffer[obs.id] \leftarrow \langle (obs.x, obs.y, obs.t), Cell, 0 \rangle$ 

 $\text{updateMeasures\&Buffer}(Cell, point_{pred}, point_{cur}, id, buffer)$ 
   $Cell.distance \leftarrow Cell.distance + \text{dist}(point_{pred}, point_{cur})$ 
   $Cell.time \leftarrow Cell.time + (point_{cur}.t - point_{pred}.t)$ 

   $Cell.speed \leftarrow \frac{Cell.distance}{Cell.time}$ 

   $v_{cur} \leftarrow \frac{\text{dist}(point_{pred}, point_{cur})}{point_{cur}.t - point_{pred}.t}$ 

```

```

if ( $v_{cur} > Cell.v_{max}$ )
     $Cell.v_{max} \leftarrow v_{cur}$ 
 $v_{init} \leftarrow \text{extractSpeed}(buffer[id])$ 
 $Cell.\Delta v \leftarrow Cell.\Delta v + (v_{cur} - v_{init})$ 

 $Cell.acc \leftarrow \frac{Cell.\Delta v}{Cell.time}$ 
 $buffer[id] \leftarrow \langle point_{cur}, Cell, v_{cur} \rangle$ 

```

This second algorithm is used to compute measures m_4 to m_7 . The buffer contains the identifier of the active trajectories, their last processed point, the cell such point belongs to, and the speed in the segment ending at such a point (for the first point of a trajectory we assume that the speed is 0).

As in the previous case, the algorithm consists of an infinite loop which processes the observations arriving in streaming. For any observation obs , after initialising the measures and the buffer, we consider the cell C_{cur} which obs belongs to. If C_{cur} is different from the cell C_{pred} associated with the last stored observation $point_{pred}$ of the trajectory, we linearly interpolate the two points (obs and $point_{pred}$) and consider all the intersections with borders of spatio-temporal cells (function *interpNewPoints*) as described in Section 3. The points corresponding to such intersections are inserted in a queue (ordered with respect to the time) and then processed in order to update the measures of the cells which these points belong to. We remark that as a result of the addition of these new points, the trajectory is divided in segments, each one completely included in a base cell, a fact which allows to update the measures of base cells in a correct way.

5 Exploiting collected data

Measures extracted from our trajectory DW could be used to discover properties of the real world, either directly by means of standard OLAP operators, or using them as an input for subsequent analyses, e.g., for data mining purposes.

As a possible use case of our system, consider the data collected automatically by mobile phone companies about the movements of individuals, and stored in our trajectory DW. In this context, observations correspond to the information recorded by the base stations of the cellular network about mobile phones. The measures stored in the DW could be used to discover traffic phenomena, like traffic jams. Unfortunately, it seems difficult to detect them by only considering single measures. As an example, consider that a traffic jam in a cell implies a low average speed of trajectories crossing the cell itself. In many cases this simple argument is not reliable, when observations not only refer to car drivers but also to pedestrians, bikers, etc. Thus a cell crossed by a lot of pedestrians could be associated with a low average speed, even if no traffic jam actually occurred. In addition, a velocity of 50 km/h is normal for a urban road, but it surely corresponds to a traffic jam for a highway. These considerations seem to exclude that a single measure, along with a trigger level, can be sufficient to detect a complex event, like a traffic jam, with enough accuracy.

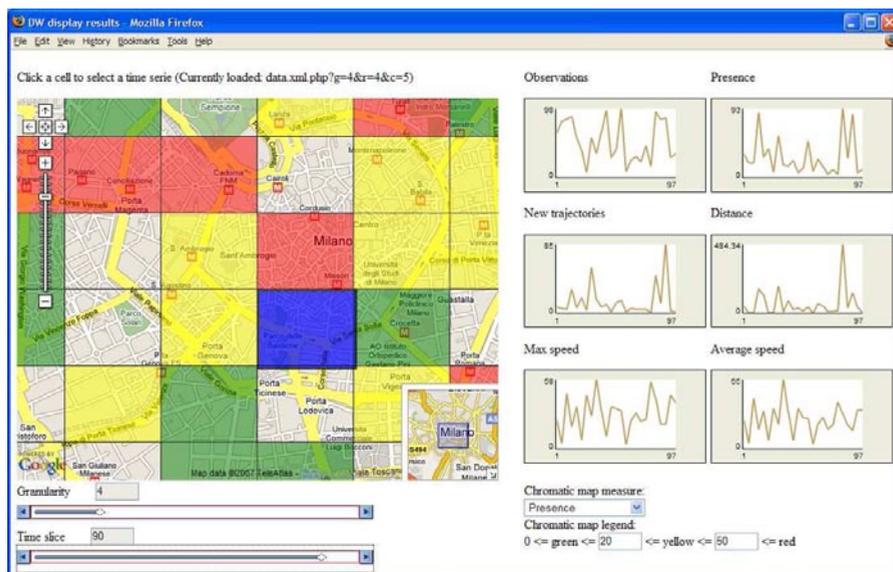


Fig. 4. Visualisation interface.

In order to discover interesting phenomena, analysts could take advantage from visual tools providing a visual representation of the extracted measures. For example, they could visualise the time series of the measures associated with a given spatial region (*zone*) in order to understand specific temporal trends in that area. Moreover, if a combination of measures is recognised as predictive for some phenomena, such as a traffic jam or a very crowded area, their values can be used to colour chromatic maps. Figure 4 shows an example of these two types of visualisations coexisting in the same user interface. The chromatic map on the left shows the discretized value of the measure *presence* in a given temporal interval (the 90th time slice) for different zones of the centre of Milan. Only zones traversed by at least one trajectory are coloured. In particular, the various colours (appearing as gray levels in this printout) indicate different concentrations of trajectories. The user can modify the map by selecting the granularity, the time slice and the measures to display, and, also, the discretization criteria to be associated with different colours. Finally, if the user selects a specific zone using the mouse, a thick border appears around the zone itself, while a set of charts is shown on the right side. Each chart represents the temporal evolution of a measure in the selected zone.

The data extracted from the DW, referenced in both space and time, can also be used as input for a subsequent *data mining* analysis (such as clustering, classification, mining of sequential and non sequential frequent pattern) [7]. Even if this can be considered a well known and common activity, in our case some additional complexity comes from the spatial and temporal dimensions of the data (see, e.g., [10, 4]) extracted from the trajectory DW, which make

it challenging to exploit mining tools that were not thought for analysing such data.

Typical data extracted from our trajectory DW can simply be the *records of measures* associated with the various cells (base or aggregate ones), possibly integrated with other information/measures related to town/road planning, demography, economics, etc., which can come from external sources like a GIS.

According to [18], in order to extract knowledge we can mine either *intra-zone* or *inter-zone* data, where a zone is identified by its spatial coordinates. An intra-zone analysis is conducted on data records referring to the same spatial zone observed at different times. An inter-zone analysis is instead conducted on records referring to various zones – e.g., zones that are geographically close or that have some common features – observed during different time intervals.

In addition, if we know that a given phenomenon (e.g, a traffic jam) actually occurred in specific spatio-temporal cells, we can exploit this supervised knowledge to label our extracted data. For example, such labelled records can then be used to build a *classification model*, or to understand which combination of inter- or intra-zone measures implies, with high probability, the occurrence of the given phenomenon.

6 Conclusions and Future Work

In this paper we have discussed some relevant issues arising in the design of a DW suited to store measures regarding trajectories. In particular, we outlined two algorithms for incrementally updating the DW on the basis of a stream of observations and we discussed how the values stored in base cells can be aggregated in order to implement the roll-up operations. We hinted at the possibility of using the aggregate information stored in the DW in order to discover properties of real world entities, either by visualisation techniques or by using such data as input for subsequent data mining analysis.

The multi-dimensional cube model adopted in the present work is simple, since the spatio-temporal dimensions are discretized according to a regular grid. In future works we aim at considering more sophisticated concept hierarchies, to be associated with spatial (i.e., coordinate roadway, district, cell, city, province, country) and temporal (i.e., second, minute, hour, day, month, year) dimensions, corresponding to the spatio-temporal framework wherein objects actually travel. We think that most of the techniques presented in this paper can be extended to this more general situation. Certainly some modifications will be required, e.g., to deal with the presence of base cells with irregular borders and possibly overlapping each other. The new setting should also influence the way in which we reconstruct trajectories from the set of samplings, since the known constraints on object movements, e.g. the existence of roads, should be taken into account in the interpolation process.

Finally, the study of more complex and structured measures, like frequent sequential patterns on semantic mapping of trajectories (e.g., *home* → *school* → *office*), appears to be a stimulating direction of future research.

References

1. F. Braz, S. Orlando, R. Orsini, A. Raffaetà, A. Roncato, and C. Silvestri. Approximate aggregations in trajectory data warehouses. In *Proc. of ICDE Workshop STDM*, pages 536–545, 2007.
2. S. Chaudhuri and U. Dayal. An overview of data warehousing and OLAP technology. In *SIGMOD Record*, volume 30, pages 65–74, 1997.
3. M. L. Damiani and S. Spaccapietra. *Spatial Data Warehouse Modelling*, pages 21–27. Processing and Managing Complex Data for Decision Support. 2006.
4. F. Giannotti, M. Nanni, D. Pedreschi, and F. Pinelli. Mining sequences with temporal annotations. In *Proc. of SAC*, pages 593–597, 2006.
5. J. Gray, S. Chaudhuri, A. Bosworth, A. Layman, D. Reichart, M. Venkatrao, F. Pellow, and H. Pirahesh. Data cube: A relational aggregation operator generalizing group-by, cross-tab and sub-totals. *Data Mining and Knowledge Discovery*, 1(1):29–54, 1997.
6. J. Han, Y. Chen, G. Dong, J. Pei, B. W. Wah, J. Wang, and Y. D. Cai. Stream cube: An architecture for multi-dimensional analysis of data streams. *Distributed and Parallel Databases*, 18(2):173–197, 2005.
7. J. Han and M. Kamber. *Data Mining: Concepts and Techniques*. Morgan Kaufmann, 2nd edition, 2006.
8. J. Han, N. Stefanovic, and K. Kopersky. Selective materialization: An efficient method for spatial data cube construction. In *Proc. of PAKDD*, pages 144–158, 1998.
9. I. Lopez, R. Snodgrass, and B. Moon. Spatiotemporal aggregate computation: A survey. *IEEE TKDE*, 17(2):271–286, 2005.
10. D. Malerba, A. Appice, and M. Ceci. A Data Mining Query Language for Knowledge Discovery in a Geographical Information System. In *Database Support for Data Mining Applications*, pages 95–116. 2004.
11. P. Marchant, A. Brisebois, Y. Bédard, and G. Edwards. Implementation and evaluation of a hypercube-based method for spatiotemporal exploration and analysis. *ISPRS Journal of Photogrammetry & Remote Sensing*, 59:6–20, 2004.
12. D. Papadias, Y. Tao, P. Kalnis, and J. Zhang. Indexing spatio-temporal data warehouses. In *Proc. of ICDE*, pages 166–175, 2002.
13. T. Pedersen and N. Tryfona. Pre-aggregation in Spatial Data Warehouses. In *Proc. of SSTD*, volume 2121 of *LNCS*, pages 460–480, 2001.
14. P. Rigaux and A. V. M. Scholl. *Spatial Database: With Application to Gis*. Morgan Kaufmann, 2001.
15. S. Rivest, Y. Bédard, and P. Marchand. Towards Better Support for Spatial Decision Making: Defining the Characteristics of Spatial On-Line Analytical processing (SOLAP). *Geomatica*, 55(4):539–555. 2001.
16. S. Shekhar and S. Chawla. *Spatial Databases: A Tour*. Prentice Hall, 2003.
17. S. Shekhar, C. Lu, X. Tan, S. Chawla, and R. Vatsavai. *Map Cube: a Visualization Tool for Spatial Data Warehouse*, chapter Geographic Data Mining and Knowledge Discovery. 2001.
18. P.-N. Tan, M. Steinbach, V. Kumar, C. Potter, S. Klooster, and A. Torregrosa. Finding Spatio-Temporal Patterns in Earth Science Data. In *KDD Workshop on Temporal Data Mining*, 2001.
19. Y. Tao and D. Papadias. Historical spatio-temporal aggregation. *ACM TOIS*, 23:61–102, 2005.
20. M. Worboys and M. Duckham. *GIS: A Computing Perspective (second edition)*. CRC Press, 2004.

Web Service Discovery at Process-level Based on Semantic Annotation

Francesco Pagliarecci¹, Marco Pistore², Luca Spalazzi¹, and Paolo Traverso²

¹Università Politecnica delle Marche, Ancona ²ITC-IRST, Trento
¹{pagliarecci, spalazzi}@diiga.univpm.it
²{pistore, traverso}@itc.it

Abstract. This paper proposes a new approach to the discovery of distributed processes described as semantic web services. In existing approaches, the discovery is performed by means of keywords that the search engine use to classify and to select web services, or on the basis of the “functional” description of a service in terms of its inputs, outputs, preconditions and effects. However, this way of discovering web services is not satisfactory. Indeed, a user may have requirements about the behavior of web services. In this paper, we define a formal framework that allows for process-level service discovery, i.e., for discovering services taking into account requirements on the behavior. The approach is based on a representation of services at the process level that is based on BPEL specifications for the behavior and that extends BPEL specification with semantic annotations that permit to perform a limited, but efficient and still useful, semantic reasoning for discovering web services.

1 Introduction

A lot of work has been done on the problem of discovering services that are described at the *functional level*, i.e., of services described as single request-response atomic components, which, given some inputs, return some outputs. In the *functional-level discovery problem*, semantic web services are described with inputs, outputs, preconditions and effects, see, e.g. [8, 5]. However, it is widely recognized that complex services should be described also at the *process level* - see e.g., [4, 12, 1], i.e., with a behavioral description of the flow that is required to interact with a service. Indeed, most often, component services are not atomic, and they cannot in general be executed in a single request-response step. They are instead stateful processes that require to follow an interaction protocol, which may involve different sequential, conditional, and iterative steps.

For these kinds of services, functional-level discovery may not be fully satisfactory, since some of the requirements that a service should satisfy are requirements about the behavior that a service should exhibit, they are requirements about the steps of interaction of the desired service. For instance, a reasonable process-level requirement for a hotel booking service is the possibility to cancel the reservation after a room has been booked. For an e-Government service in the health care domain, a requirement can be that all the interaction steps used to provide personal and sensitive data should be done after some authentication procedure is performed. A requirement for an on-line shop could be that the data for payment (e.g., credit card number) is requested only

after the interaction loop for selecting items is concluded. We call a service discovery problem that involves requirements about the behavior of the service to be discovered, a *process-level discovery problem*.

In this paper, we propose a solution to the process-level discovery problem, which is based on the following ingredients:

- We describe web services at the process level with semantically annotated abstract BPEL process. According to the line described in [10], this allows for both a behavioral and (incremental) semantic description of web services.
- We define a language that can express requirements on the behavior of the service that has to be discovered. The language is a temporal logic enriched with concept and role assertions.
- We formally define the problem of process-level service discovery. The behavior of services that are described as semantic annotated abstract BPEL processes can be described as (semantically annotated) state transition systems. Given a set of services whose behavior is formalized as annotated state transition systems, and given a discovery requirement described in temporal logic, the problem is to find a service such that its behavior satisfies the requirement.
- We provide a novel algorithm for discovering services that satisfy semantically annotated behavioral requirements. The algorithm combines both reasoning about procedural behaviors that must satisfy temporal conditions, as well ontological reasoning about the semantics of data.

The paper is structured as follows. In Section 2, we describe a reference example that is used all along the paper. In Section 3 we define semantically annotated state transition systems that describe BPEL processes. In Section 4, we report the language for describing semantically annotated discovery requirements and we formally define the process-level discovery problem. In Section 5 we describe the discovery algorithm.

2 Overview of the Approach: An Example

Our running example consists in the discovery of existing theatre ticket booking services that satisfy given user requirements. We assume that the user has already selected a set of suitable services by means of traditional functional-level discovery techniques (e.g., keyword-based matching, semantic matching based on an input-output-precondition-effect description of service, etc.). Now, the user has to select a Web service according to its process-level description.

According to our approach [10], each Web service defines: an ontology defining the relevant terminology; a process interface defining the interactions necessary to execute the service; and an annotation of the choreography that defines (partial) correspondences between the ontology and the process. In the following, we assume that the ontologies provided by the different services have been mapped into a common ontology that defines all the relevant concepts of the discovery scenario — the literature on ontologies defines many powerful techniques to achieve this mapping among the local ontologies of the processes. We will also assume that the interface processes are defined in abstract BPEL and are annotated according to the common ontology using the standard description logic notations.

$$\begin{aligned}
\text{Time} &= \forall \text{hh.Number} \sqcap \forall \text{mm.Number} , & \text{TransStatus} , \text{PayStatus} \sqsubseteq \mathbb{T} , \\
\text{Date} &= \forall \text{year.Number} \sqcap \forall \text{month.Number} \sqcap \forall \text{day.Number} , \text{Gender} \sqsubseteq \mathbb{T} , \\
\text{Client} &= \forall \text{clientName.String} \sqcap \forall \text{gender.Gender} , & \text{SessionID} \sqsubseteq \text{Number} , \\
\text{Ticket} &= \forall \text{sessionID.SessionID} \sqcap \forall \text{ticketID.String} \sqcap \forall \text{show.Show} \sqcap \forall \text{client.Client} \sqcap \\
& \quad \forall \text{sector.String} \sqcap \forall \text{seatNumber.String} \sqcap \forall \text{status.TransStatus} \sqcap \forall \text{payment.Payment} , \\
\text{Show} &= \forall \text{title.String} \sqcap \forall \text{date.Date} \sqcap \forall \text{time.Time} \sqcap \forall \text{theatre.Theatre} , \\
\text{Theatre} &= \forall \text{thName.String} \sqcap \forall \text{loc.String} , \\
\text{Payment} &= \forall \text{sessionID.SessionID} \sqcap (\leq 1\text{id}) \sqcap (\geq 1\text{id}) \sqcap \forall \text{date.Date} \sqcap \forall \text{amount.Number} \sqcap \\
& \quad \forall \text{method.PayMethod} \sqcap \forall \text{status.PayStatus} , \\
\text{PayMethod} &\sqsubseteq \mathbb{T} , & \text{Cash} \sqsubseteq \text{PayMethod} , \\
\text{CreditCard} &= \text{PayMethod} \sqcap \forall \text{client.Client} \sqcap \forall \text{cardType.String} \sqcap \forall \text{cardNumber.String} \sqcap \\
& \quad \forall \text{from.Date} \sqcap \forall \text{to.Date} , \\
\text{BankTransfer} &= \text{PayMethod} \sqcap \forall \text{client.Client} \sqcap \forall \text{bankName.String} \sqcap \forall \text{account.String} , \\
\text{PayPal} &= \text{PayMethod} \sqcap \forall \text{client.Client} \sqcap \forall \text{ppAccount.String}
\end{aligned}$$

Fig. 1. The terminology of the running example

male : Gender, female : Gender, available : TransStatus, notAvailable : TransStatus,
booked : TransStatus, cancelled : TransStatus, waiting : PayStatus,
sentData : PayStatus, authorized : PayStatus, paid : PayStatus

Fig. 2. The common part of each ABox in the running example

This ontology contains some general, domain-independent concepts (Date, Time, Client), and some concepts that are specific on the considered domain (Ticket, Show, Payment, ...). Notice that, in the definition of Ticket and Payment, we have the role status whose values are restricted to be of the concepts TransStatus and PayStatus, respectively. This role captures different states of the client request and of the payment procedure. More precisely, the status of a (requested) ticket can assume the values available or notAvailable depending on the availability; the status assumes values booked or cancelled if the ticket has been booked or cancelled. Similarly, the status of a payment is sentData after the information related to the payment (e.g., account number or credit card number) has been sent, it is authorized if the client has authorized the payment gateway to pat, and it is paid when the payment has been processed; it is waiting in all the other cases. The possible values (instances) for the concepts TransStatus and PayStatus are listed in the ABox in Figure 2.

In our approach, the interface processes defining the interaction behaviors of the component services are defined in abstract BPEL. BPEL [1] provides an operational description of the (stateful) behavior of web services on top of the service interfaces defined in their WSDL specifications. An abstract BPEL description identifies the partners of a service, its internal variables, and the operations that are triggered upon the invocation of the service by some of the partners. Operations include assigning variables, invoking other services and receiving responses, forking parallel threads of execution,

and nondeterministically picking one amongst different courses of actions. Standard imperative constructs such as if-then-else, case choices, and loops, are also supported.

```

<process abstractProcess="yes" name="TicketTwo" ... >
...
  <assign name="offerDataMapping">
    <copy>
      <!--semann="/theatreOfferMsg/t:Ticket, tid:String, s:Show, c:Client, sect:String, seat:String, p:Payment, today:Date, th:Theatre,
cc:CreditCard, /theatreOfferMsg/t.sessionID=/theatreRequestMsg/key, /theatreOfferMsg/t.ticketID=tid, /theatreOfferMsg/t.show=s,
/theatreOfferMsg/t.client=c, /theatreOfferMsg/t.sector=sect, /theatreOfferMsg/t.seatNumber=seat,
/theatreOfferMsg/t.status=available, /theatreOfferMsg/t.payment=p, s.title=/theatreRequestMsg/title,
s.date=/theatreRequestMsg/date, s.time=/theatreRequestMsg/time, s.theatre=th, c.name=/theatreRequestMsg/clientName,
p.sessionID=/theatreRequestMsg/key, p.date=today, p.amount=/theatreRequestMsg/ticketPrice, p.method=cc,
p.status=waiting, th.thName=/theatreRequestMsg/theatreName" -->
      <from opaque="yes"/>
      <to part="ticket" variable="theatreOfferMsg"/>
    </copy>
  </assign>
...
</process>

```

Fig. 3. A portion of annotated BPEL process of the *TicketTwo* service. A full file is available at <http://www.diiga.univpm.it/~spalazzi/reports/TicketTwo.bpel>.

In Figure 3, we report the abstract BPEL specification of a possible interface for a ticket service, namely service *TicketTwo*¹. After receiving a request for a theatre ticket (receive operation at the beginning of the file), the service of the process decides on the availability of the tickets (switch operation named *Availability*). We remark that the implementation of this check, which depends on the information system of the theatre, is irrelevant for the description of the interaction protocol and is hence not specified in the abstract BPEL. If the ticket is not available (case named *NotAvailableTickets*), this is notified to the client and the service finishes its execution. If the ticket is available (case named *AvailableTickets*), the service sends an offer and, in the case the client accepts it (onMessage operation *transactionAck*), it handles the payment with the credit card.

A BPEL specification such as the one in Figure 3 describes in a very detailed way the interactions that need to be carried out with a web service in order to exploit it. However, this is still not sufficient to allow for the purpose of automatically discovering such web service. Indeed, it is necessary to describe also the “semantic” aspects of such interactions. We do this by extending the BPEL specification with “semantic annotations” (the **semann** attributes in Figure 3). In our example, it is necessary first of all to associate concepts in the ontology to the (parts of the) input and output messages exchanged by the process. This is the role, for instance of the semantic annotations “/theatreRequestMsg/clientName : String, /theatreRequestMsg/theatreName : String, ...” of the receive activity for operation *theatreRequest*, at the beginning of the BPEL process. Moreover, it is necessary to express “semantic” relations among the input and output data values exchanged during the interaction with the web service, e.g., between the show title, date, and time requested by the client and the ticket returned by the reservation service. This is done in annotation “... /theatreOfferMsg/t.show=s, ... s.title=/theatreRequestMsg/title, s.date=/theatreRequestMsg/date, s.time=/theatreRequestMsg/time ...” of the opaque

¹ The specification contains some annotations in boldface: they are not part of the BPEL language, and we will explain their meaning later on in this section. Also, the BPEL specification has been slightly edited to improve the readability.

assignment. A further usage of semantic annotations is to define the outcome of an interaction with a web service. In our example it is clear that a show has been booked only if a ticket is available, the reservation service sends an offer, and the user acknowledges the acceptance of the offer. To express this in the BPEL specification, we add annotation “/theatreOfferMsg/t.status = booked” to the activity corresponding to the reception of the acknowledgment.

We remark that the process described in Figure 3 is only one of the possible interfaces for a ticket reservation service. Different sellers could adopt different approaches, e.g., requiring that the information on the client credit card is given before the acknowledgment rather than after, or providing to the user a second choice if the first offer is refused. The semantic annotations are necessary to compensate the specificities of the interface at hand, and to put it in relation with the common ontology. We remark, however, that the semantic annotations that have to be added to this purpose are very limited if compared to processes defined in languages such as OWL-S or WSMO. As we will see, they are sufficient for the automated discovery task we are interested in.

3 BPEL Processes as Annotated STSs

We encode BPEL processes (extended with semantic annotations) as *annotated state transition systems* [10]. State transition systems (STS) describe dynamic systems that can be in one of their possible *states* (some of which are marked as *initial states*) and can evolve to new states as a result of performing some *actions*.

```

PROCESS TicketTwo;
STATE pc : { START, ReceiveRequest, CheckAvailability, AvailableTickets, PrepareOffer,
            invokeOffer, requestAck, transactionAck, receiveCCData, invokeCCService,
            NotAvailableTickets, invokeNotAvailable, END_NA, END_ACK, END_NACK };
INIT pc = { START };
CONCEPT String; Number; Date; Time; SessionID; Ticket; Show; Client; Payment; Theatre; CreditCard;
INPUT request(theatreRequestMsg); transactionNAckMsg(); transactionAckMsg(); receiveCCData(CCDDataMsg);
OUTPUT ticketsNotAvailable(); theatreResponse(theatreOfferMsg); checkCreditCard(checkCCMsg);
TRANS
  pc = START -[TAU]-> pc = ReceiveRequest;
  pc = ReceiveRequest -[INPUT request(theatreRequestMsg)]-> pc = CheckAvailability;
  pc = CheckAvailability -[TAU]-> pc = NotAvailableTickets;
  pc = CheckAvailability -[TAU]-> pc = AvailableTickets;
  ...
ANNOTATION FUNCTION
...
LAMBDA(ReceiveRequest) = LAMBDA(START);
LAMBDA(CheckAvailability) = { /theatreRequestMsg/clientName:String,
                              /theatreRequestMsg/theatreName:String,
                              /theatreRequestMsg/title:String, /theatreRequestMsg/date:Date,
                              /theatreRequestMsg/time:Time, /theatreRequestMsg/ticketPrice:Number,
                              /theatreRequestMsg/sector:Number, /theatreRequestMsg/key:SessionID }
  ∪ LAMBDA(ReceiveRequest);
...

```

Fig. 4. A portion of annotated STS corresponding to the TicketTwo process. A full file is available at <http://www.diiga.univpm.it/~spalazzi/reports/TicketTwo.smv>.

We distinguish actions in *input actions*, *output actions*, and τ . *Input actions* represent the reception of messages, *output actions* represent messages sent to external

services, and τ is a special action, called *internal action*, that represents internal evolutions that are not visible to external services. In other words, τ represents the fact that the state of the system can evolve without producing any output, and without consuming any inputs. A *transition relation* describes how the state can evolve on the basis of inputs, outputs, or of the internal action τ .

In an *annotated STS*, we associate to each state a set of *concept assertions* and *role assertions*. This configures a state as the assertional component (or ABox) of a knowledge representation system based on a given description logic where the ontology plays the role of the terminological component (or TBox). Therefore, *concept assertions* are formulas of the form $a : C$ (or $C(a)$) and state that a given individual a belongs to (the interpretation) of the concept C . *Role assertions* are formulas of the form $a.R = b$ (or $R(a, b)$) and state that a given individual b is a value of the role R for a . As a consequence, each action can be viewed as a transition from a state consisting in an ABox in a different state consisting in a different ABox.

Definition 1 (Annotated state transition system). An annotated state transition system is a tuple $\langle \Sigma, \mathcal{T}, \Lambda \rangle$ where:

- $\Sigma = \langle \mathcal{S}, \mathcal{S}^0, \mathcal{I}, \mathcal{O}, \mathcal{R} \rangle$ is the state transition system;
- \mathcal{S} is the finite set of states;
- $\mathcal{S}^0 \subseteq \mathcal{S}$ is the set of initial states;
- \mathcal{I} is the finite set of input actions;
- \mathcal{O} is the finite set of output actions;
- $\mathcal{R} \subseteq \mathcal{S} \times (\mathcal{I} \cup \mathcal{O} \cup \{\tau\}) \times \mathcal{S}$ is the transition relation;
- \mathcal{T} is the terminology (TBox) of the annotation;
- $\Lambda : \mathcal{S} \rightarrow 2^{\mathcal{A}\mathcal{T}}$ is the annotation function, where $\mathcal{A}\mathcal{T}$ is the set of all the concept assertions and role assertions defined over \mathcal{T} .

Example 1. Figure 4 shows a textual description of the annotated STS corresponding to the annotated BPEL code of Figure 3. The set of states \mathcal{S} (the section STATE in Figure 4) models the steps of the process and the evolution of the concept and the role assertions. `pc` is a variable that ranges over the set of states \mathcal{S} and thus holds the current execution step of the service (e.g., `pc = CheckAvailability` when it is ready to check whether the ticket for the required show is available). The set of initial states \mathcal{S}^0 is represented by the section INIT in Figure 4. The concepts used in the annotated STS are listed in the section CONCEPT of Figure 4. They must be defined in the terminology \mathcal{T} . According to the formal model, we distinguish among three different kinds of actions. The input actions \mathcal{I} model all the incoming requests to the process and the information they bring (e.g., `request` is used for the receiving of the client request). The output actions \mathcal{O} model all the actions that send outgoing messages (e.g., `theatreResponse` is used to bid a ticket). The action τ is used to model internal evolutions of the process, such as assignments and decision making. The set of input actions \mathcal{I} (output actions \mathcal{O}) are represented by the section INPUT (OUTPUT). The evolution of the process is modelled by the section TRANS in Figure 4.

The annotation function Λ (see the section ANNOTATION FUNCTION in Figure 4) models how the assertions vary depending on the states. For instance, $\Lambda(\text{LAMBDA}(\text{PrepareOffer})) = \{/\text{theatreOfferMsg}/t:\text{Ticket},$

..., /theatreRequestMsg/title:String, ..., /theatreOfferMsg/t.show=s, ..., s.title=/theatreRequestMsg/title, ...} represents the fact that state PrepareOffer contains, for instance, the concept assertions /theatreOfferMsg/t:Ticket, s:Show, and /theatreRequestMsg/title:String (i.e., /theatreOfferMsg/t, s, and /theatreRequestMsg/title are individuals that belong to concepts Ticket, Show, and String, respectively) and the role assertions /theatreOfferMsg/t.show=s, (the role show of the individual /theatreOfferMsg/t is filled with the individual s) and s.title=/theatreRequestMsg/title (the role title of the individual s is filled with the individuals /theatreRequestMsg/title).

We remark that each TRANS clause and each LAMBDA clause of Figure 4 corresponds to different elements in the transition relation \mathcal{R} and in the annotation function Λ , respectively. For example, the transition and the LAMBDA clause described above generate different elements of \mathcal{R} and Λ depending on which individuals /theatreRequestMsg/title we have in the destination state. Concerning cancel, it has been defined in Figure 2 and, thus, it denotes the same individual in all the states (i.e., all the ABoxes). For this reason, we have included in LAMBDA(START) and propagated in all the other states the assertions defined in Figure 2. The definition of the state transition system provided in Figure 4 is therefore parametric w.r.t. the individuals that can be associated to concepts declared in the section CONCEPT. In order to obtain a concrete state transition system (a set of concrete ABoxes) and to apply the techniques described in this paper, a finite set of individuals have to be assigned to those concepts. A possible approach to assign these individuals consists in defining appropriate concept assertions in the common part of the ABoxes (e.g., the part of ABoxes depicted in Figure 2). Another, better technique is to use knowledge level techniques such as the ones in [9] to avoid an explicit enumeration of the individuals.

As already remarked in [10], when we have to check if a given assertion p is true in a given state we have to apply instance checking denoted as $\langle \mathcal{T}, \Lambda(s) \rangle \models p$. In the specific case, when we check subsumption, ABoxes play no active role [11], therefore subsumption can be checked without considering what is the current state (i.e., the current ABox). For example, when we have to check $\langle \mathcal{T}, \Lambda(s) \rangle \models C \sqsubseteq D$, we only need to check $\langle \mathcal{T}, \emptyset \rangle \models C \sqsubseteq D$. Furthermore, let us assume to use \mathcal{ALN} as description logic and a generalized acyclic TBox as \mathcal{T} [2]. This language is expressive enough to describe non-trivial examples as the ticket seller example. Nevertheless, the computational complexity of subsumption w.r.t. an acyclic terminology is NP-complete while the computational complexity of instance checking is P [6], which makes the reasoning services tractable and, for example, less complex than those of OWL-S.

4 Web Service Discovery Problem

According to our approach, the inputs of the discovery problem are (1) a global ontology $\langle \mathcal{T}, \mathcal{A} \rangle$, (2) a set of annotated BPEL processes, or, equivalently, of annotated STSs $\langle \Sigma_i, \mathcal{T}, \Lambda_i \rangle$, defining the available services, and (3) a discovery requirement ρ , that formalizes the desired properties of the web service to be selected. Inputs (1) and (2) have been already described. We now focus on the definition of the discovery requirement.

Example 2. We would like to buy a ticket for the show “Cats” in a given theatre. Therefore, first of all we need to discover web services for booking and buying such a kind of ticket. Furthermore, we do not want to provide our credit card data before the confirmation of the ticket. Therefore, we want to restrict to services that respect this constraint on the interaction protocol.

After a search at the functional level, we can retrieve several services able to satisfy our first requirement, but we are not able to distinguish what are the services that satisfy our second requirement. Indeed, we need to check the behavior of the services in order to know what are the services satisfying that requirement.

In order to express process-level discovery requirements such as the one in the previous example, we introduce a variant of CTL [7], a well-know temporal logic that is widely used to express requirements on the behavior of dynamic systems. More precisely, once a process-level requirement is modeled as a CTL formulas, the formula can be evaluated on a State Transition System, checking in this way of the STS satisfies the requirement. The key feature of our extension of CTL is that it allows for expressing conditions on *annotated* STS, i.e., it can express conditions on concept and role assertions.

Definition 2 (Goal Condition). Let $x : C$ be a concept assertion and $x.R = y$ be a role assertion defined w.r.t. \mathcal{T} , then a goal condition is defined as follows:

$$p = x : C \mid x.R = y \mid p \text{OR } p \mid p \& p \mid \text{NOT } p \mid \text{AF } p \mid \text{AG } p \mid \text{EF } p \mid \text{EG } p \mid \text{AX } p \mid \text{EX } p \mid \\ A(p \mathcal{U} p) \mid E(p \mathcal{U} p) \mid A(p \mathcal{B} p) \mid E(p \mathcal{B} p)$$

CTL is a propositional, branching-time, temporal logic. Intuitively, according to our extension, a goal condition must be verified along all possible computation paths (state sequences) starting from the current state. Concerning the propositional connectives NOT and &, they have their usual meanings of negation and conjunction. The other propositional operators can be defined in terms of these. Concerning the temporal operators (i.e., AF, EF, AX, and so on), they maintain the same intuitive meaning that they have in standard CTL. In other words, $\text{EX}p$ is true in a state s if and only if s has at least a successor t such that p is true at t . $\text{E}[p\mathcal{U}q]$ is true in a state s if and only if there exists a path starting at s and an initial prefix of the path such that q holds at the last state of the prefix and p holds at all other states along the prefix. $\text{EG}p$ is true at state s if there is at least a path starting at s such that p holds at each state on the path. Concerning the concept and role assertions that are in a goal condition, we call them goal concept assertions and goal role assertions, respectively. Intuitively, a goal concept assertion denotes a typical description logic problem: the *retrieval inference problem*. Let $x : C$ denote a goal concept assertion, the retrieval inference problem is the problem of finding for each state s all individuals mentioned in the ABox $\Delta(s)$ that are an instance of the concept C w.r.t. the given TBox \mathcal{T} . A non-optimized algorithm for a retrieval can be realized by testing for each individual occurring in the ABox whether it is an instance of the concept C . Once we have retrieved a set of instances $\{a\}$ for the goal concept assertion $x : C$, we can substitute x in the goal condition with the retrieved instances and check whether the condition holds. Therefore, a *goal condition* denotes a *set of specifications* to be checked instead of a single one.

A formal definition can be found in [7]. We also use the following syntactic abbreviation for CTL formulas:

- $AXp \equiv \text{NOT EX NOT}p$ which means that p holds at all successor states of the current state (p must hold at the next state).
- $EFp \equiv E[\text{true}\mathcal{U}p]$ which means that for some path there exists a state on the path at which p holds (p is possible in the future).
- $AFp \equiv \text{NOT EG NOT}p$ which means that for every path there exists a state on the path at which p holds (p is inevitable in the future).
- $AGp \equiv \text{NOT EF NOT}p$ which means that for every path at every node on the path p holds (p holds invariantly along all path).
- $A[p\mathcal{U}q] \equiv \text{NOT E}[(\text{NOT}q)\mathcal{U}(\text{NOT}p\&\text{NOT}q)]\& \text{NOT EG}(\text{NOT}q)$ which means that for every path there exists an initial prefix of the path such that q holds at the last state of the prefix and p holds at all other states along the prefix (p holds until q holds along all paths).
- $A[p\mathcal{B}q] \equiv \text{NOT E}[\text{NOT}p\mathcal{U}q]$ which means that for every path, if q ever happens in the future, it is strictly preceded by an occurrence of p .

Besides the condition expressed in our variant of CTL, a discovery requirement also defines two sets of concept assertions. The first one, that we call *input concept assertions*, can be seen as input parameters for the discovery requirements, such as desired show and dates, which can be assumed to exist in the ABox of the global ontology. The second set, called *output concept assertions*, describes the elements that have to be returned by the discovered web service, in our case the ticket and the payment. We now give a formal definition of discovery requirements.

Definition 3 (Process-level Discovery Requirement). *Let \mathcal{T} be the terminology for the composition problem. A discovery requirement is a tuple $\rho = \langle i, o, p \rangle$, where:*

- i is a set of input concept assertions for \mathcal{T} ;
- o is a set of output concept assertions for \mathcal{T} ;
- p is a goal condition on $\langle \mathcal{T}, i \cup o \rangle$.

Example 3. The discovery requirement of the ticket scenario is based on the following input concept assertions: x : Show (the show the client desires to attend to); y : CreditCard (the credit card the client desires to use for paying). The output concept assertions are: z : Ticket is the ticket returned by the seller; u : Payment is the payment of the ticket. As discussed in Example 2, the goal requires to book a ticket for a given show:

x :Show & y :CreditCard & z :Ticket & u :Payment
 $EF(z.\text{status} = \text{booked} \& z.\text{show} = x \& x.\text{title} = \text{"Cats"} \& z.\text{payment} = u \& u.\text{method} = y) \& A(z.\text{status} = \text{booked}) B(u.\text{status} = \text{sentData})$

The second line of the formula expresses the fact that we look for a service that can possibly lead to book the "Cats" show (operator EF – notice that the show can be booked only if there are available tickets). The third line expresses the condition that, in any case, we want the ticket to be booked before we sent the credit card details (operator $A[- B -]$).

Definition 4 (Process-level Discovery Requirement Satisfaction). Let $\Gamma = \langle \Sigma, \mathcal{T}, \Lambda \rangle$ be an annotated state transition system. Let $\rho = \langle i, o, p \rangle$ be a discovery requirement. Then, Γ satisfies ρ , written $\Gamma \models \rho$ iff for each initial state $s \in \mathcal{S}^0$, there exists a specification p' of the goal condition p such that $\langle \Sigma, \mathcal{T}, \Lambda(s) \cup i \cup o \rangle \models p'$.

In the discovery requirements, the semantic annotations introduced in the BPEL processes play a fundamental role for defining conditions on the outcomes of web service executions. Now that we have defined all the inputs of the discovery, we are ready to provide a formal definition of discovery problem.

Definition 5 (Process-level Discovery Problem). Let $\Gamma_1, \dots, \Gamma_n$ be a set of annotated state transition systems on the same terminology \mathcal{T} , and let $\rho = \langle i, o, p \rangle$ be a discovery requirement. The discovery problem for $\Gamma_1, \dots, \Gamma_n$ and ρ is the problem of finding an annotated state transition system Γ_i such that $\Gamma_i \models \rho$.

5 Web Service Discovery Approach

According to Definition 5, a process-level service discovery problem can be reduced to a model checking problem: Given a set of annotated state transition systems and a discovery requirement, the basic idea consists of model checking the goal condition for each annotated STS. However, the traditional model checkers cannot be used, as they are not able to deal with the ontological reasoning necessary to deal with the annotations to the states of the STS. In this section, we discuss an approach that make it possible to reuse as much as possible existing model checkers (e.g., NuSMV [3]), in order to exploit their very efficient and optimized verification techniques. This approach is based on the idea to solve *before the model checking task* the problem of knowing in which states the assertions contained in the goal condition hold. That is, before the application of the traditional model checking algorithm, we have to map ontological assertions (of the process annotations) into propositions. Furthermore, we have to transform the assertions contained in the goal condition into conditions on the state propositions. This is simply obtained by applying the retrieval inference service of \mathcal{ALN} for each goal concept assertion of the goal condition and for each state of the annotated STS. Therefore, for each goal concept assertion, we have a set of concept assertions to map into propositions. After that, we generate a set of role assertions for each goal role assertion and for each retrieved instance. Then, we can apply instance checking service of \mathcal{ALN} for each generated role assertion and for each state of the annotated STS. Each role assertion that holds in a given state is mapped in a proposition, as well. Concerning the complexity, the retrieval inference problem for \mathcal{ALN} has a polynomial complexity. Furthermore, we have to apply a polynomial number of time instance checking that, for \mathcal{ALN} has a polynomial complexity. The result is that the process of mapping the assertions in the goal condition has a polynomial complexity.

As a further step, we have to generate a set of specifications by substituting the retrieved instances in concept and role assertions in the goal condition. Finally, the model checking algorithm can be applied.

Example 4. Let us consider the goal condition reported in Example 3. Applying the retrieval inference service, we obtain that goal concept assertion x :Show can be mapped into annotation s :Show; y :CreditCard can be mapped into annotation cc ; z :Ticket can be mapped into annotation $/theatreOfferMsg/t$; and u :Payment can be mapped into annotation p . Applying the instance checking service, we obtain in which states the assertions reported in the goal condition hold, as reported in the following table (Table 1). After that, we have to generate all the specifications. In this example, we have only a possible specification to generate, namely the following specification:

s :Show & cc :CreditCard & z :Ticket & p :Payment
 $EF(/theatreOfferMsg/t.Ticket.status = booked \ \& \ /theatreOfferMsg/t.Ticket.show = s \ \& \ s.title = \text{"Cats"} \ \& \ /theatreOfferMsg/t.Ticket.payment = p \ \& \ p.method = cc) \ \& \ A(/theatreOfferMsg/t.Ticket.status = booked) \ B \ (p.status = sentData)$

This specification has been passed to the model checker and is resulted satisfied.

Table 1. State and Assertion after retrieval inference and instance checking service

Assertion	States
s :Show	PrepareOffer, invokeOffer, requestAck, transactionAck, receiveCCData, invokeCCService, END_ACK, END_NACK
cc :CreditCard	PrepareOffer, invokeOffer, requestAck, transactionAck, receiveCCData, invokeCCService, END_ACK, END_NACK
$/theatreOfferMsg/t$:Ticket	PrepareOffer, invokeOffer, requestAck, transactionAck, receiveCCData, invokeCCService, END_ACK, END_NACK, NotAvailableTickets, invokeNotAvailable, END_NA
p :Payment	PrepareOffer, invokeOffer, requestAck, transactionAck, receiveCCData, invokeCCService, END_ACK, END_NACK
$s.title = \text{"Cats"}$	PrepareOffer, invokeOffer, requestAck, transactionAck, receiveCCData, invokeCCService, END_ACK, END_NACK
$p.method = cc$	PrepareOffer, invokeOffer, requestAck, transactionAck, receiveCCData, invokeCCService, END_ACK, END_NACK
$p.status = sentData$	receiveCCData, invokeCCService, END_ACK
$/theatreOfferMsg/t.status = booked$	transactionAck, receiveCCData, invokeCCService, END_ACK
$/theatreOfferMsg/t.show = s$	PrepareOffer, invokeOffer, requestAck, transactionAck, receiveCCData, invokeCCService, END_ACK, END_NACK
$/theatreOfferMsg/t.payment = p$	PrepareOffer, invokeOffer, requestAck, transactionAck, receiveCCData, invokeCCService, END_ACK, END_NACK

We remark that, besides the approach considered above, other approaches are possible where the model checking tools are extended in order to be able to handle natively the ontological annotations in states and formulas. We intend to explore this approach in future work.

The use of model checking allows us to have requirements at the process-level, as we have exemplified in the above example. The use of semantic annotation allows us to have flexible discovery strategies. Indeed, we can navigate in the ontology in order to make a given requirement more general. For instance, a generalization according to this strategy consists in looking in the ontology for the most specific concepts that subsume concepts of goal concept assertions in the goal condition, and to reformulate the condition using these concepts, as shown in the following example.

Example 5. The goal in Example 3 contains the following concept assertion, y : CreditCard. In the terminology of Figure 1, there is the concept PayMethod that subsumes CreditCard. Taking this into account, the principal goal can be generalized:

x :Show & y :PayMethod & z :Ticket & u :Payment &
 $EF(z.status = booked \ \& \ z.show = x \ \& \ x.title = \text{"Cats"} \ \& \ z.payment = u \ \& \ u.method = y) \ \&$
 $A(z.status = booked) \ B(u.status = sentData)$

As result of this generalization, we can discover web services that allow different payment methods, not only by means of credit card.

The above strategy is a generalization of the goal condition in the sense that the discovered web services are a superset of web services discovered with the original condition. Indeed, the following result holds:

Theorem 1 (Completeness w.r.t. discovery).

Let ρ be a discovery requirement and ρ_g be the corresponding generalization w.r.t. the terminology T . Then if $\Gamma \models \rho$ then $\Gamma \models \rho_g$

6 Related work and conclusions

In this paper, we propose a novel solution to the problem of the discovery of web services based on requirements on their behavior. We show how behavioral requirements can be expressed in CTL, a well known temporal logic, enriched with concept and role assertions. We define an algorithm, based on model checking techniques, which, given a behavioral description of a service, enriched with semantic annotations, and a discovery goal that expresses behavioral requirements, can determine whether the service satisfies the discovery goal. The semantically annotated behavioral description used in this paper has been already applied in [10] for service composition. The usage of this model for discovery, and in particular the model checking approach for enriched CTL is a novel contribution of this paper.

Most of the work on the discovery of semantic web services addresses the problem at the functional level and does not take into account behavioral and temporal requirements. The work described in [13], describes a framework where service discovery can be guided by behavioral descriptions, e.g., provided as UML specifications. However, this work is orthogonal to ours, since it proposes a general framework that can support a developer in finding proper services, rather than providing an automated technique for discovery as we propose in this paper. The WSMO [12] framework recognize the importance of interfaces that describe behaviors of services, and proposes the use of mediators to match service behaviors against (discovery) goals. However, the WSMO framework does not provide any specific technique for the automatic discovery as we propose in this paper. More in general, none of the previous works addresses the problem of the discovery of semantic annotated BPEL descriptions, depending on temporal requirements.

References

1. T. Andrews, F. Curbera, H. Dolakia, J. Goland, J. Klein, F. Leymann, K. Liu, D. Roller, D. Smith, S. Thatte, I. Trickovic, and S. Weeravarana. Business Process Execution Language for Web Services (version 1.1), 2003.
2. F. Baader and W. Nutt. Basic Description Logics. In F. Baader, D. Calvanese, D. McGuinness, D. Nardi, and P. Patel-Schneider, editors, *The Description Logic Handbook*, pages 43–95. Cambridge University Press, 2003.
3. A. Cimatti, E. M. Clarke, F. Giunchiglia, and M. Roveri. NUSMV: a new symbolic model checker. *International Journal on Software Tools for Technology Transfer (STTT)*, 2(4), 2000.
4. The OWL Services Coalition. OWL-S: Semantic Markup for Web Services, 2003.
5. I. Constantinescu, B. Faltings, and W. Binder. Typed Based Service Composition. In *Proc. WWW'04*, 2004.
6. F. M. Donini. Complexity of Reasoning. In F. Baader, D. Calvanese, D. McGuinness, D. Nardi, and P. Patel-Schneider, editors, *The Description Logic Handbook*, pages 96–136. Cambridge University Press, 2003.
7. E. Emerson. Temporal and modal logic. In *Handbook of Theoretical Computer Science, Vol. B: Formal Models and Semantics*. Elsevier, 1990.
8. M. Paolucci, K. Sycara, and T. Kawamura. Delivering Semantic Web Services. In *Proc. WWW'03*, 2002.
9. M. Pistore, A. Marconi, P. Bertoli, and P. Traverso. Automated Composition of Web Services by Planning at the Knowledge Level. In *Proc. IJCAI'05*, 2005.
10. M. Pistore, L. Spalazzi, and P. Traverso. A minimalist approach to semantic annotations for web processes compositions. In *Proc. of the 3rd European Semantic Web Conference (ESWC 2006)*, Budva (Montenegro), 11–14 June, 2006. Springer-Verlag, Berlin, Germany.
11. A. Schaerf. *Query Answering in Concept-Based Knowledge Representation Systems: Algorithms, Complexity, and Semantic Issues*. Dottorato di Ricerca in Informatica, Università degli Studi di Roma “La Sapienza”, Italia, 1994.
12. SDK WSMO working group. The Web Service Modeling Framework - <http://www.wsmo.org/>.
13. Andrea Zisman and George Spanoudakis. Uml-based service discovery framework. In *Proc. ICSOC 2006*, volume 4294 of *Lecture Notes in Computer Science*, pages 402–414, 2006.

Hiding Sequences^{*}

Osman Abul, Maurizio Atzori, Francesco Bonchi, and Fosca Giannotti

Pisa KDD Lab, ISTI - CNR
Via Giuseppe Moruzzi, 1 - 56124 Pisa, Italy
{name.surname}@isti.cnr.it

Abstract. The process of discovering relevant patterns holding in a database, was first indicated as a threat to database security by O' Leary in [13]. Since then, many different approaches for knowledge hiding have emerged over the years, mainly in the context of association rules and frequent itemsets mining. Following many real-world data and applications demands, in this paper we shift the problem of knowledge hiding to contexts where both the data and the extracted knowledge have a sequential structure. We provide problem statement, some theoretical issues including NP-hardness of the problem, a polynomial sanitization algorithm and an experimental evaluation.

1 Introduction

Privacy-preserving data mining, i.e., the study of data mining side-effects on privacy, has rapidly become a hot and lively research area [6, 13, 3, 15], which has seen the proliferation of many completely different approaches having different objectives, application contexts and using different techniques. The chronologically first approach in privacy preserving data mining was aimed at avoiding the identification of the original database rows (by means of data *perturbation* or *obfuscation*) while at the same time allowing the reconstruction of the data distribution at an aggregate level, and thus the production of valid mining models [3, 2, 7, 8, 14].

Another approach, named *knowledge hiding*, aims at hiding some knowledge (i.e. rules/patterns) considered sensitive, that could be inferred from the data which is going to be published. This hiding is usually obtained by *sanitizing* the database in input in such a way that the sensitive knowledge can no longer be inferred, while the original database is changed as less as possible.

A novel approach has recently emerged in privacy preserving data mining, studying the privacy threats (and possibly appropriate solutions) arising when *publishing the data mining results* themselves instead of the data [12, 5, 4, 10].

Clearly, this is not meant to be an exhaustive overview of the existing approaches. The aim is to show that there is a large variety of different approaches,

^{*} This work was carried out during the tenure of first author's ERCIM fellowship. This work is also supported by the EU project GeoPKDD (IST-6FP-014915). Authors acknowledge Mirco Nanni for fruitful discussions.

that however have a common aspect: the kind of data and patterns considered. Due to the inherent challenges of protecting privacy while discovering knowledge, most of the work so far has focussed on simple, flat relations and on classical data mining tasks, such as decision trees, clustering, association rules and frequent itemsets. But nowadays, the real-world applications call for more advanced analysis of more structured and more complex data.

Consider spatio-temporal geo-referenced data daily collected by telecommunication companies exploiting mobile phones and other location-aware devices. The increasing availability of *space-time trajectories* of these personal devices and their human companions is expected to enable novel classes of applications, where the discovery of consumable, concise, and applicable knowledge is the key step. These mobile trajectories contain detailed information about personal and vehicular mobile behaviour, and therefore offer interesting practical opportunities to find behavioral patterns, to be used for instance in traffic and sustainable mobility management, e.g., to study the accessibility to services. Clearly, in these applications privacy is a concern, since location data enables intrusive inferences, which may reveal habits, social customs, religious and sexual preferences of individuals, and can be used for unauthorized advertisement and user profiling. Similarly, consider *web usage log data* that contain traces of sequences of actions taken by a user, or *biomedical patient data* that usually contain clinical measures at different moment in time. In each of these applications, (i) both the data and the kind of patterns of interest have some structure (i.e., sequences in time), (ii) there exist substantial privacy threats, as well as (iii) evident potential usefulness of knowledge discovered from these data.

In this paper we shift the problem of knowledge hiding from the usual *frequent itemsets*, to contexts where both the data and the extracted knowledge have a *sequential structure*, as a step toward data types used in current real-world applications like mobility data analysis and web mining.

2 Hiding Sequential Patterns

In this section we first introduce some notation and then we formally provide *The Sequence Hiding Problem* statement. It is then proven that the problem of finding an optimal sanitization is NP-Hard.

2.1 Problem Definition

In this paper we focus on the discovery of patterns that are a simple sequence of symbols (or events, or positions). Let \mathcal{D} be a database of sequences, where each $T \in \mathcal{D}$ is a finite sequence of symbols from an alphabet Σ : $T = \langle t_1, \dots, t_{T_n} \rangle$ where $t_i \in \Sigma, \forall i \in \{1, \dots, T_n\}$. We denote the set of all sequences as Σ^* . A sequence $U \in \Sigma^*$ is a subsequence of a sequence $V \in \Sigma^*$, denoted $U \sqsubseteq V$, if U can be obtained by deleting some symbols from V . More formally, $U = \langle u_1, \dots, u_m \rangle$ is subsequence of $V = \langle v_1, \dots, v_n \rangle$ if there are m indices $i_1 < \dots < i_m$ such that $u_1 = v_{i_1}, \dots, u_m = v_{i_m}$. The support of a sequence $S \in \Sigma^*$ is the number of sequences in \mathcal{D} that are supersequences of S : $sup_{\mathcal{D}}(S) = |\{T \in \mathcal{D} \mid S \sqsubseteq T\}|$.

The classical problem of mining frequent patterns requires, given a database \mathcal{D} and a minimum support threshold σ , to compute all patterns that have a support larger than σ : $\mathcal{F}(\mathcal{D}, \sigma) = \{S \in \Sigma^* \mid \text{sup}_{\mathcal{D}}(S) \geq \sigma\}$. The sequence hiding problem instead is defined as follows.

Problem 1 (The Sequence Hiding Problem) *Let $\mathcal{S}_h = \{S_1, \dots, S_n\}$ with $S_i \in \Sigma^*, \forall i \in \{1, \dots, n\}$, be the set of sensitive sequences that must be hidden from \mathcal{D} . Given a disclosure threshold ψ , the Sequence Hiding Problem requires to transform \mathcal{D} in a database \mathcal{D}' such that:*

1. $\forall S_i \in \mathcal{S}_h, \text{sup}_{\mathcal{D}'}(S_i) \leq \psi$;
2. $\sum_{S \in \Sigma^* \setminus \mathcal{S}_h} |\text{sup}_{\mathcal{D}}(S) - \text{sup}_{\mathcal{D}'}(S)|$ is minimized.

The problem requires to sanitize the input database \mathcal{D} in such a way that a set of sensitive patterns \mathcal{S}_h is hidden, while the most of the information in \mathcal{D} is maintained. The resulting database \mathcal{D}' is the released one.

The first requirement asks all sensitive patterns to be *hidden* in \mathcal{D}' , i.e., they must have a support not more than the given disclosure threshold. The second requirement asks to minimize the sanitization effects on all non-sensitive patterns (regardless of their frequency). Note that this is equivalent to keep \mathcal{D}' as similar as possible to \mathcal{D} . This is a very general definition which does not say *how* the sanitization is actually performed.

In the following, for sake of presentation, we assume:

1. to sanitize all occurrences of sensitive patterns, i.e., $\psi = 0$;
2. to sanitize sequences in the input database by means of an abstract operator, called *marking* that replaces a selected symbol in a position with a special symbol Δ which is not in Σ .

In the following section we show that the problem of sanitizing a sequence $T \in \mathcal{D}$, introducing the smallest number of Δ is a NP-Hard problem. Before that, we need to introduce the concept of matching set.

Definition 1 (Matching Set). *Given two sequences $S \in \mathcal{S}_h$ and $T \in \mathcal{D}$, we define the matching set of S in T , denoted \mathcal{M}_S^T , as the set of all sets with size $|S|$ of indices for which $S \sqsubseteq T$. For instance, let $S = \langle a, b, c \rangle$ and $T = \langle a, a, b, c, c, b, a, e \rangle$, in this case we got $\mathcal{M}_S^T = \{(1, 3, 4), (1, 3, 5), (2, 3, 4), (2, 3, 5)\}$. Moreover, given a sequence $T \in \mathcal{D}$ we define $\mathcal{M}_{\mathcal{S}_h}^T = \bigcup_{S \in \mathcal{S}_h} \mathcal{M}_S^T$.*

The notion of matching set is important to identify the point in the input database where the sanitization process should act. Clearly, if for a given sequence $T \in \mathcal{D}$ there is no match, i.e., $\mathcal{M}_{\mathcal{S}_h}^T = \emptyset$, then T does not support any sensitive sequence and thus it can be disclosed as it is. Otherwise it should be transformed such that all the matches in $\mathcal{M}_{\mathcal{S}_h}^T$ are removed. Given a sequence $T \in \mathcal{D}$ such that $\mathcal{M}_{\mathcal{S}_h}^T \neq \emptyset$, we need to introduce a certain number of Δ symbols in T in such a way that it is sanitized.

2.2 Optimal Sanitization is NP-Hard

We now prove that the problem of identifying the smallest possible set of positions to be sanitized by replacing their symbol with Δ is a NP-Hard problem.

Problem 2 (Sequence Sanitization) Given: A sequence T and a set of patterns \mathcal{S}_h to be hidden. **Objective:** Find a set of position indices of T such that, replacing the symbols in the positions with Δ results in $\mathcal{M}_{\mathcal{S}_h}^T = \emptyset$.

Theorem 1. *Optimal Sequence Sanitization Problem is NP-Hard.*

Proof (sketch). By reduction to the HITTING SET PROBLEM. See [1] for details.

Since the Optimal Sequence Sanitization Problem is NP-Hard we do not expect to find an optimal efficient solution. In other words, we need to resort to heuristics, as done in the next section.

3 A Polynomial Sanitization Algorithm

In this section we define a two-stage sanitization algorithm for hiding a set of sensitive sequences \mathcal{S}_h from a database \mathcal{D} . During the first stage, the sequences in the input database are sanitized introducing the necessary Δ symbols. During the second stage, we either delete Δ s or replace them with symbols from Σ . When the latter approach is chosen, we must take care of the possibility of re-generating fake patterns and also re-generating sensitive patterns.

In this paper, we do not consider neither deletion of Δ s, nor replacement, we just focus on the marking operation. Let us point out that the second stage can be skipped totally in case we allow existence of marking symbols Δ in \mathcal{D}' : in this case, these symbols may be interpreted as missing values. Note that the marking operation here does not create new subsequences, thus there is no fake patterns introduced by this process.

Therefore, in the rest of this section we focus on the first stage, i.e, sanitizing by marking. Here we got two problems to address: on the local scale, given a sequence $T \in \mathcal{D}$, how to choose the positions to mark, and on the global scale, which sequences $T \in \mathcal{D}$ to sanitize. One heuristic is provided for both problems.

Intuitively, if there are small number of matches then the sanitization can be done with less distortion. So, the size of $\mathcal{M}_{\mathcal{S}_h}^T$ is a crucial issue. Unfortunately this number is exponential in the length of the sequence in the worst case.

Lemma 1 (Size of Matching Set). *The size of the matching set \mathcal{M}_S^T is exponential in the length of sequence T in the worst case.*

Proof. We have $|\mathcal{M}_S^T| = \binom{|T|}{|S|}$ when S and T are sequences of the same single symbol and nothing else. The *middle binomial coefficient* (i.e., when $k = n/2$) is known for being the largest of the binomial coefficients $\binom{n}{k}$, which can be approximated by an application of Stirling's formula as: $\binom{n}{\frac{n}{2}} \sim \sqrt{\frac{2}{\pi}} n^{-1/2} 2^n$. \square

The key observation is that certain markings affect the number of matchings while others do not. In the following we use the usual array notation for sequences: i.e., $S[i]$ to denote the element of S at position i (with positions starting from 1).

Example 1. Consider again the case $S = \langle a, b, c \rangle$ and $T = \langle a, a, b, c, c, b, a, e \rangle$. In this situation marking the symbol e ($T[8]$) does not affect the matching set while marking the symbol b in $T[3]$ position will cause $\mathcal{M}_S^T = \emptyset$. Note that the latter marking removes all the matching which is equivalent of hiding all sensitive pattern instances and thus provides sanitization. Also note that marking $T[1]$ reduces the number of matches without providing sanitization, while marking $T[1]$ and $T[2]$ together provides sanitization.

Since there are many ways of providing sanitization by choosing different positions for marking, we need to choose the one with the minimum cost (the cost here is the number of non sensitive subsequences which are removed due to the marking). Our *local* heuristic is:

choose the marking position that is involved in most matches.

This operation is iterated until $\mathcal{M}_{S_h}^T = \emptyset$. We denote the number of matchings in which the i th position of T is involved as $\delta(T[i])$.

Example 2. Consider again the case $S = \langle a, b, c \rangle$ and $T = \langle a, a, b, c, c, b, a, e \rangle$. We got that $\delta(T[1]) = 2$, $\delta(T[2]) = 2$, $\delta(T[3]) = 4$, and so on. So, we choose position $T[3]$ for marking (this causes $T = \langle a, a, \Delta, c, c, b, a, e \rangle$). Since this selection removes all matchings, no further iteration is needed.

The considered heuristic requires the matching set $\mathcal{M}_{S_h}^T$ to be computed and then, for each position of T , the number of occurrences in $\mathcal{M}_{S_h}^T$ counted. It is shown (Lemma 1) that the size of $\mathcal{M}_{S_h}^T$ may grow exponentially and the time required to produce this set is thus also exponential. So, the procedure of generating this set and then counting occurrences is not feasible. However, we note that the heuristic only requires the knowledge of number of matching each position is involved in, and not necessarily the $\mathcal{M}_{S_h}^T$ itself. Fortunately, this can be computed in polynomial time as articulated next.

Lemma 2 (Computation of matching set size). *The computation of matching set size can be done in polynomial time.*

Proof. Consider $S \in \mathcal{S}_h$ of length m and $T \in \mathcal{D}$ of length n . Let $P_{1..m}^{1..n}$ denote the size of \mathcal{M}_S^T , and, for instance $P_{1..m}^{1..n-1}$ denote the size of the matching set when the last element of T is removed. It is clear that if $S[m] \neq T[n]$, then $P_{1..m}^{1..n} = P_{1..m}^{1..n-1}$. On the other hand, when $S[m] = T[n]$, then $P_{1..m}^{1..n} = P_{1..m}^{1..n-1} + P_{1..m-1}^{1..n-1}$. The last equation holds since, there are two options either match last symbols or not and these two events are disjoint and exhaustive. In the boundary conditions, $P_0^j = 1, \forall j \in \{0, 1, 2, \dots, n\}$ and $P_i^0 = 0, \forall i \in \{0, 1, 2, \dots, m\}$. By employing dynamic programming, an algorithm with the complexity of $O(n * m)$ can be obtained. \square

The key issue now is how to compute $\delta(T[i])$ for each position i in T .

Theorem 2. *Computing $\delta(T[i])$ for each position i in T , can be done in polynomial time.*

Proof. It is sufficient to show that this can be done in polynomial time for any given position i . All matchings in $\mathcal{M}_{\mathcal{S}_h}^T$ can be dichotomized based on whether including i or not. Consider $T' = T \setminus T[i]$, i.e., t from which we remove the i th element. The proof is based on two observations; 1) the set of matchings not involving i in the original matching set does not change 2) deletion of an element does not create any new subsequence. So, $\delta(T[i])$ can be computed as $|\mathcal{M}_{\mathcal{S}_h}^T| - |\mathcal{M}_{\mathcal{S}_h}^{T'}|$. \square

In the case $\psi = 0$ (sensitive sequences must be completely removed from \mathcal{D}), we apply the same algorithm considered above for all sequences in \mathcal{D} . Otherwise, we have to select the set of sequences to be sanitized. We rely on the following global heuristic:

sort the sequences in \mathcal{D} in ascending order of matching set size, and remove all matchings in top $|\mathcal{D}| - \psi$ input sequences.

We consider size of matching set as a sorting criteria because if the matching set is small then we can sanitize the respective sequence with less distortion.

Algorithm 1 Sequence Hiding Algorithm

Input: $\mathcal{D}, \mathcal{S}_h, \psi$

Output: \mathcal{D}'

- 1: $\mathcal{D}' \leftarrow \emptyset$
 - 2: **for all** $T \in \mathcal{D}$ **do**
 - 3: Compute size of $\mathcal{M}_{\mathcal{S}_h}^T$
 - 4: $\mathcal{D} \leftarrow$ Sort \mathcal{D} in ascending order w.r.t. $\mathcal{M}_{\mathcal{S}_h}^T$
 - 5: $\mathcal{D}_{\text{sanitize}} \leftarrow$ Top $|\mathcal{D}| - \psi$ sequences in \mathcal{D}
 - 6: **for all** $T \in \mathcal{D}_{\text{sanitize}}$ **do**
 - 7: $T' \leftarrow \text{Sanitize}(T, \mathcal{S}_h)$
 - 8: $\mathcal{D}' \leftarrow \mathcal{D}' \cup T'$
 - 9: $\mathcal{D}' \leftarrow \mathcal{D}' \cup (\mathcal{D} \setminus \mathcal{D}_{\text{sanitize}})$
 - 10: $\mathcal{D}' \leftarrow \text{ReplaceMarkingSymbol}(\mathcal{D}', \mathcal{S}_h, \psi)$
-

Algorithm 1 summarizes the method that we introduced. It computes the matching set size for all $T \in \mathcal{D}$ using the strategy described in Lemma 2, and then it sorts the sequences in ascending order. The first $|\mathcal{D}| - \psi$ sequences in the order are selected to be sanitized. The sanitization procedure $\text{Sanitize}(T, \mathcal{S}_h)$ eliminates all occurrences of \mathcal{S}_h within a sequence T by introducing the Δ symbol in some chosen positions. The positions are selected according to our heuristic: i.e., first select the position with the largest $\delta(T[i])$. Therefore for each position i of T it computes $\delta(T[i])$ using the algorithm described in the proof of Theorem 2, then select the best position and sanitize it by changing the actual symbol with Δ . The process is iterated until $\mathcal{M}_{\mathcal{S}_h}^T = \emptyset$.

4 Experimental Evaluation

In this section we report the experiments we conducted in order to assess the effectiveness of the proposed approach, in terms of distortion introduced by the sanitization. We experimented on two datasets; a real truck movement data used in [9], containing 273 trajectories, and a synthetic car movement data, containing 300 trajectories, generated in our laboratory [11]. The former dataset is referred TRUCKS and the latter SYNTHETIC throughout. In both of these datasets, the movement sequences are discretized using 10 by 10 grid where locations are indexed with $X_i Y_j$, where $i, j \in \{1, 2, \dots, 10\}$. Thus the trajectories are discretized in sequences over an alphabet Σ of 100 symbols. This discretization results in 20.1 (resp. 6.8) locations (i.e., symbols), on average, per trajectory for TRUCKS (resp. SYNTHETIC) datasets.

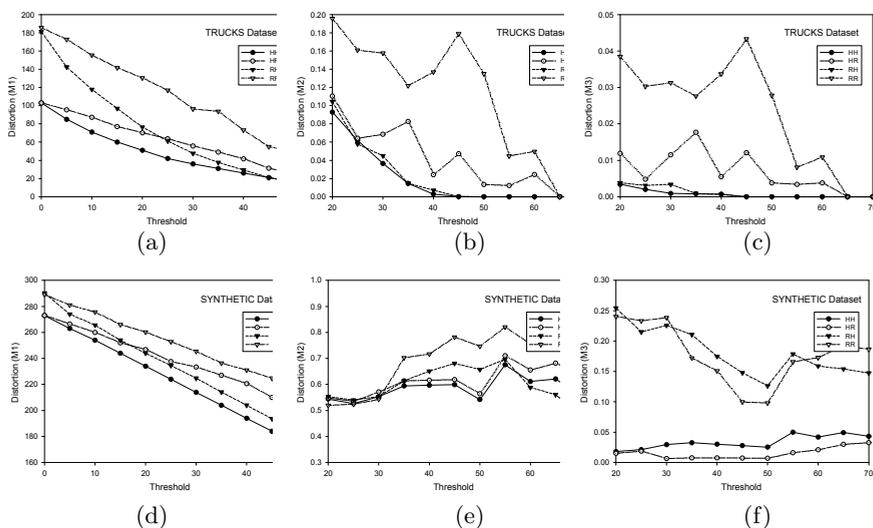


Fig. 1. Distortion empirical evaluation.

Since there is no other algorithm for sequence sanitization in the literature, we measure the effectiveness of the proposed algorithm against random sanitization. Recall that the algorithm employs two orthogonal heuristics: (i) for selecting the positions to be sanitized within a given sequence; (ii) for selecting the subset of sequences to be sanitized. We can easily employ random settings along the two orthogonal dimensions to get different algorithms. This way it becomes possible to measure the individual and collective contribution of the two heuristics proposed. To this end, we name these four algorithms as HH (Heuristic-Heuristic), HR (Heuristic-Random), RH (Random-Heuristic), and RR (Random-Random). HR is interpreted as using the heuristic in a given sequence sanitization but selecting a random subset to be sanitized, and other 3 schemes are defined accordingly.

It is worth noting that in random settings, the choice of positions in a sequence, and the choice of sequences to be sanitized, are not completely blind: the random choice is actually performed only among reasonable choices, i.e., positions and sequences that *need* to be sanitized.

We tested three distortion measures as defined next:

- M1 (Data distortion): total number of marking symbols in \mathcal{D}' .
- M2 (Frequent Pattern Distortion):

$$\frac{|\mathcal{F}(\mathcal{D}, \sigma)| - |\mathcal{F}(\mathcal{D}', \sigma)|}{|\mathcal{F}(\mathcal{D}, \sigma)|}$$

- M3 (Frequent Pattern Support Distortion):

$$\frac{1}{|\mathcal{F}(\mathcal{D}', \sigma)|} \sum_{S \in \mathcal{F}(\mathcal{D}', \sigma)} \frac{\text{sup}_{\mathcal{D}}(S) - \text{sup}_{\mathcal{D}'}(S)}{\text{sup}_{\mathcal{D}}(S)}$$

Note that the measure M1 focuses on the distortion on the whole data, while measures M2 and M3 focus only on the distortion on the frequent patterns. Moreover, M1 is an absolute measure while M2 and M3 are relative measures ranging between 0 and 1.

In the experiments, $\mathcal{S}_h = \{\langle X_6Y_3, X_7Y_2 \rangle, \langle X_4Y_3, X_5Y_3 \rangle\}$ for the TRUCKS dataset and $\mathcal{S}_h = \{\langle X_2Y_7, X_3Y_7 \rangle, \langle X_5Y_7, X_5Y_6 \rangle\}$ for the SYNTHETIC dataset are the sequences to be hidden. In the following we report the support of these sensitive patterns: this is an important information since it strongly influences the distortion introduced by the sanitization.

$\mathcal{D} = \text{TRUCKS}, \mathcal{D} = 273$	$\mathcal{D} = \text{SYNTHETIC}, \mathcal{D} = 300$
$\text{sup}_{\mathcal{D}}(\langle X_6Y_3, X_7Y_2 \rangle) = 36$	$\text{sup}_{\mathcal{D}}(\langle X_2Y_7, X_3Y_7 \rangle) = 99$
$\text{sup}_{\mathcal{D}}(\langle X_4Y_3, X_5Y_3 \rangle) = 38$	$\text{sup}_{\mathcal{D}}(\langle X_5Y_7, X_5Y_6 \rangle) = 172$
$\text{sup}_{\mathcal{D}}(\langle X_6Y_3, X_7Y_2 \rangle \vee \langle X_4Y_3, X_5Y_3 \rangle) = 66$	$\text{sup}_{\mathcal{D}}(\langle X_2Y_7, X_3Y_7 \rangle \vee \langle X_5Y_7, X_5Y_6 \rangle) = 200$

In Figure 1 we report all experimental results where random cases are averaged over 10 runs. The threshold on the X-axis is always the disclosure threshold ψ . For the experiments on measures M2 and M3 we always use a minimum frequency threshold equivalent to the disclosure threshold: i.e., $\sigma = \psi$.

Figure 1(a) and (d) report the experimental results for the M1 measures. They show that HH performs consistently best and RR consistently worst for both datasets at all thresholds. Comparing HR and RH reveals that up to a certain threshold, sequence level sanitization heuristics is more important than heuristically selecting the subset to be sanitized. And vice versa after that disclosure threshold. This shows the importance of heuristics at both level too.

Figure 1(b),(c) report the M2 and M3 distortion measures on the TRUCKS dataset, while Figure 1(e),(f) report the same measures on the SYNTHETIC dataset. The results confirm that HH algorithm achieves the best performance and RR achieves the worst for both measures on the TRUCKS dataset. This agrees with the results obtained for measure M1. However, achieving best performance for M1 do not imply the best performance for M2 and M3 as results show (Figure 1(e),(f)). Recall that the designed algorithm is aimed to reduce M1 not M2 and M3. But, on the average, it is also effective in reducing M2 and M3.

5 Conclusion and Future Work

To the best of our knowledge, this is the first work addressing the problem of knowledge hiding in sequential pattern mining. We have defined sequence hiding problem, proving that optimal sanitization is NP-Hard. Thus we have introduced a heuristic algorithm which aims less distortion while providing sanitization. The experiments demonstrate the effectiveness of the approach.

References

1. O. Abul, M. Atzori, F. Bonchi, and F. Giannotti. Hiding sequences. In *Proceedings of the Third International Workshop on Privacy Data Management (2007)*.
2. D. Agrawal and C. C. Aggarwal. On the design and quantification of privacy preserving data mining algorithms. In *Proceedings of the 20th ACM PODS*, pages 247–255, 2001.
3. R. Agrawal and R. Srikant. Privacy-preserving data mining. In *Proceedings of the 2000 ACM SIGMOD*, pages 439–450, 2000.
4. M. Atzori, F. Bonchi, F. Giannotti, and D. Pedreschi. Anonymity preserving pattern discovery. *VLDB Journal*. Accepted for publication 2006.
5. M. Atzori, F. Bonchi, F. Giannotti, and D. Pedreschi. Blocking anonymity threats raised by frequent itemset mining. In *Proceedings of the Fifth IEEE ICDM*, pages 561–564, 2005.
6. C. Clifton and D. Marks. Security and privacy implications of data mining. In *Proceedings of the 1996 ACM SIGMOD*, pages 15–19, Feb. 1996.
7. W. Du and Z. Zhan. Using randomized response techniques for privacy-preserving data mining. In *Proceedings of the Ninth ACM KDD*, pages 505–510, 2003.
8. A. Evfimievski, R. Srikant, R. Agrawal, and J. Gehrke. Privacy preserving mining of association rules. In *Proceedings of the Eighth ACM KDD*, pages 343–364, 2002.
9. E. Frentzos, K. Gratsias, N. Pelekis, and Y. Theodoridis. Nearest neighbor search on moving object trajectories. In *SSTD'05*, 2005.
10. A. Friedman, A. Schuster, and R. Wolff. k-anonymous decision tree induction. In *Proc. of PKDD '06*, pages 151–162, Berlin, Germany, 2006. Springer-Verlag.
11. F. Giannotti, A. Mazzoni, S. Puntoni, and C. Renso. Synthetic generation of cellular network positioning data. In *13th ACM International Symposium on Advances in Geographic Information Systems (ACM GIS'05)*, Bremen, Germany, 2005.
12. M. Kantarcioglu, J. Jin, and C. Clifton. When do data mining results violate privacy? In *Proceedings of the Tenth ACM KDD*, pages 599–604, 2004.
13. D. E. O'Leary. Knowledge discovery as a threat to database security. In G. Piatetsky-Shapiro and W. J. Frawley, editors, *Knowledge Discovery in Databases*, pages 507–516. AAAI/MIT Press, 1991.
14. S. Rizvi and J. R. Haritsa. Maintaining data privacy in association rule mining. In *Proceedings of the 28th VLDB*, 2002.
15. V. S. Verykios, E. Bertino, I. N. Fovino, L. P. Provenza, Y. Saygin, and Y. Theodoridis. State-of-the-art in privacy preserving data mining. *ACM SIGMOD Record*, 33(1):50–57, 2004.

Query Optimization for Wireless Sensor Network Databases in the MadWise system *

¹ Giuseppe Amato, ^{1,2} Paolo Baronti, and ^{1,2} Stefano Chessa

¹ISTI-CNR Pisa, Italy, ²Department of Computer Science, University of Pisa, Italy.

Abstract. We propose a comprehensive approach to distributed query processing in wireless sensor networks. In our proposal we reinterpret the classical approach to database system design according to the wireless sensor networks context, and we redefine the aspects related to the definition of a query language, data model, query algebra, and query optimization strategies. We show that our approach enables optimizations of the query plan which may reduce the costs, in terms of consumed energy, of orders of magnitude.

1 Introduction

Wireless Sensor Networks [4, 6] are composed of a set of (tiny) devices (hereafter called sensors), which are microsystems, each comprising a processor, a memory, a set of transducers, and a low-range, low-bandwidth radio transceiver. Sensors are powered by on board batteries thus their energy efficiency is critical in most applications. Applications of sensor networks include, among others, environment sampling, disaster areas monitoring, and health monitoring.

The MaD-WiSe system [8],[2] allows interaction with a wireless sensor network as a traditional database management system. In a traditional database system queries are used to search for data contained in a persistent storage repository. In a wireless sensor network, the data base consists of the environmental data that can be measured/acquired by the transducers available on the sensor nodes. Queries instruct nodes on the management, filtering, and processing of the data acquired from the environment. The wireless sensor network and the software running on the nodes are the means that allow data to be acquired when needed from the environment, exactly in the way that a traditional database software allows data to be accessed on disks. In a wireless sensor network data is not stored anywhere: environmental data is acquired by transducers of the nodes when needed, in accordance with the query that the network is being processing. A new data is available every time a transducer is activated.

The MaD-WiSe system consists of a set of modules that implement a distributed stream management system on a wireless sensor network. A part of the MaD-WiSe modules (the distributed query processor) run on the nodes of the wireless sensor network (network side) and the rest of the modules (query parser, and query optimizer) run on a generic client node connected to the wireless sensor network. MaD-WiSe is implemented on the MICAz motes platform [1], relying on TinyOS [3]. The query parser

* Work funded in part by the European Commission in the framework of the FP6 projects PERSONA (contract N. 045459) and INTERMEDIA (contract N.38419)

and optimizer are implemented in Java and run on a standard PC. A sample MaD-WiSe query is the following:

```

SELECT roomB.Temp
FROM roomA, roomB
WHERE roomA.Temp > roomB.Temp and roomA.Temp > 50
EVERY 10000

```

2 About the data model and the query algebra

A natural way to imagine data processed by a wireless sensor network is that of a stream or sequence of tuples. We define three different types of streams to model the different tuple access modes: *sensor streams*, which represent streams of data acquired by the transducers, *remote streams*, which model streams of data sent by a source node to a destination node, and *local streams*, which represent streams of data generated by execution of local operations and sent as input of other local operations.

We have considered two different modes for updating the tuple of a sensor stream: i) the *periodic update mode* where the transducer is activated with a fixed period (the *sampling period*) to update the stream (this mode can be used to collect transducer readings every x seconds) and the ii) *on-demand update mode* where the transducer is activated as a consequence of a read request and causes the stream update (this mode can be used to obtain transducers readings only under specific conditions).

Remote streams support the transfer of intermediate query results between remote nodes. Note that remote streams require communication between nodes through their radio interfaces. Since radio activation is one of the primary causes of energy consumption, it should be carefully managed.

Local streams model data transfers between operators located on the same node in order to enable pipelined executions of the operations. We distinguish local from remote streams since they have different costs in terms of energy consumption. Local streams mainly consume memory resources, while their energy consumption is negligible.

Differently than the relational query algebra [7] our operators supporting the query language deal with streams of tuples according to the model defined above. These operators have a strictly pipelined behavior that avoids the use of temporary buffers for producing results.

In particular we have changed the definition of the *join* operator to adapt to the needs of data management in wireless sensor networks. This because in this context it is very useful to relate tuples generated at the same instant (or approximatively the same instant) by different transducers and/or nodes. For this reason we define a join operator, $\bowtie (S_{I_1}, S_{I_2})$, that relates tuples having the same timestamp **TS**. For every new tuple read on one of the input streams the join operator checks if the last tuple read from the other stream has the same timestamp. This operation is clearly non-blocking and its execution requires only single-position buffers. We also provide a further implementation of the join operator called *sync-join*, $\bowtie_{sync} (S_{I_1}, S_{I_2})$, where S_{I_2} is an on-demand stream. The sync-join requests the activation of S_{I_2} only when a tuple arrives on S_{I_1} .

3 The cost model

We measure the cost of a query execution plan by evaluating the power P needed to process the query in terms of energy E consumed per unit of time ($P = E/t$). We estimate the cost in terms of the energy required to send records across streams because the cost required by an operator to process a data is negligible with respect to the cost of sending data in a stream

Let $E(S, s)$ be the energy required to send a single record of size s across the stream S and $f(S)$ be the frequency of records traversing the stream S . The cost of stream S , that is its power, is $P(S) = f(S)E(S, s)$. The cost of a query execution plan, say QEP , is the sum of the cost of its streams. Let \mathbf{S} be the set of streams contained in the query execution plan QEP . The cost of QEP is $P(QEP) = \sum_{S \in \mathbf{S}} P(S)$.

Energy required for sending a record: The cost of sensor streams is dominated by the energy required to sample a value with the associated transducer. The cost of a local stream is that needed to store the records in the temporary buffer. The cost of a remote stream is dominated by the activity of the radio interfaces of the nodes in the multi-hop path from the source node to the destination node.

The energy required to store a record in a local stream is negligible with respect to the cost incurred by the other types of stream. Thus we can reliably consider a zero cost in this case. Given a local stream LS we have $E(LS, s) = 0$.

We suppose that the records of sensor streams have fixed size and structure, thus the cost solely depends on the transducer used. Given a sensor stream SS associated with transducer TR , we have $E(SS, s) = \text{energy_per_sample}(TR)$. Thermistor, Accelerometer, and Magnetometer transducers embedded in the Crossbow [1] sensor boards consume respectively 0.0000891, 0.03222, and 0.2685 mJ per sample.

In case of remote streams the situation is a bit more complex. Transmission of a tuple along a remote stream requires that the nodes involved in the corresponding multi-hop path collaborate to forward the tuple toward the destination node.

Let $E_t(s)$ be the energy required for transmitting a record of size s over the radio interface, and $E_r(s)$ be the energy required for receiving it. Thus the energy required to send a record of size s over a remote stream RS along a n hops path can be approximated by $E(RS, s) = n(E_t(s) + E_r(s))$. In many real cases the number of hops between two nodes is proportional to the distance between the two nodes [5]. Therefore, we can express the energy needed to send a record of size s across a remote stream S , where source and destination nodes have distance d as $E(S, s) = d \cdot c \cdot (E_t(s) + E_r(s))$, where c is a tuning parameter that depends on the density and transmission range of the nodes in the network.

The energy required to send and receive a 50 bytes packet by the MICAz platform [1] is respectively 0.1494225 mJ and 0.161445 mJ.

Frequency of records in streams: The frequency of records across streams depends on the periodicity of the data acquisition of the sensor streams, and on the specific operators used to connect streams.

In case of sensor streams, we distinguish between periodic and on-demand streams. In a periodic stream S_p with period p data are acquired and sent with frequency $f(S_p) = 1/p$. An on-demand stream is intended to be used as input to a sync-join operator as in $S_O \leftarrow \bowtie_{sync} (S, S_{od})$, where S_{od} , is the on-demand sensor stream. We have that

Table 1. Frequency for local and remote streams connecting various operators

Operator	$f(S_O)$
$S_O \leftarrow \pi.(S_I)$	$f(S_I)$
$S_O \leftarrow \sigma_{pred}(S_I)$	$f(S_I) \cdot \Pr(pred = true)$
$S_O \leftarrow \bowtie (S_{I_1}, S_{I_2})$	$\min\{f(S_{I_1}), f(S_{I_2})\}$

$f(S_{od}) = f(S)$ given that a record is requested from S_{od} every time a record arrives from S .

The frequency of local and remote streams depends on the operators that write in the streams and on the stream(s) where that operators read. The various possibilities are summarized in Table 1.

4 Query optimization

In this paper we consider an algebraic optimization approach, that is based on transformation rules transforming a query plan into a semantically equivalent one with a lower cost. The final query plan is obtained by applying successive transformations to an initial query plan built from the MW-SQL query. We will also discuss some issues related to the ordering of the operators, which can be affected by the transformation rules.

Several transformation rules proposed in the literature to optimize traditional database query execution can be applied in our context. For instances rules to push-down selections and projections, and selectivity-based ordering of selections are very useful since they contribute to reduce the amount of data to be transferred upward in a query plan. This implicitly reduce the amount of data traversing remote streams, and, in turn, it reduces the amount of radio activity and of energy consumed.

Here we discuss some peculiar transformation rules that are particularly useful in our context since they make optimal use of the data model and of the operators that we have defined. Specifically we consider rules according the following guidelines:

1. Sync-join and on-demand streams should be used whenever possible.
2. Given that a sync-join requires a sensor stream on the right side, trees representing query plans should be unbalanced to the left (Left Deep Join Trees). In this way, the chance that a sensor stream (a leaf node) is found as the right argument of a join is increased.
3. Unary operators such as selections, projections, and temporal aggregates (which reduce the amount of data being forwarded) should be moved as close as possible to the node where data is acquired.

Transforming a join into a sync-join: According to our previous observations we define rules that transform a join into a sync-join. The idea is that if a periodic sensor stream is on the right side of a join, the join can be transformed into a sync-join and the sensor stream into an on-demand sensor stream. If there are some unary operators between the sensor stream and the join, the unary operators can be moved after the join (to process the output of the join) and the transformation can still take place. Note that

even if the unary operators are moved up, this is not a problem, given that the sensor stream is activated only if needed.

Formally the transformation rule is the following:

$$\frac{\bowtie^k (B, \xi^*(\check{S}^h))}{\widetilde{\xi}^h(\bowtie_{sync}^h (B, \check{S}^h))} \quad \text{where } \xi^* \neq \emptyset \vee k \neq h \quad (1)$$

where \check{S}^h is a sensor stream on node h , $\widetilde{\xi}^h$ is obtained from the sequence ξ^* , where each π_X is transformed to $\pi_{X \cup Attr(B)}$, and all elements are localized on Node h .

Previous rule can be easily modified to consider the case where the sensor stream is on the right side of the join.

Obtain Left Deep Join Trees: The following rule rearranges the joins in a query plan to obtain a left deep join tree, and facilitate transformation of joins into sync-joins by means of the previous rules.

$$\frac{\bowtie (A^h, \xi^*(\bowtie (B^k, C^j)))}{\widetilde{\xi}^{j^*}(\bowtie^j (\bowtie^k (A^h, B^k), C^j))} \quad (2)$$

where $\widetilde{\xi}^{j^*}$ is obtained from ξ^* by transforming all π_X into $\pi_{X \cup Attr(A)}$, and all elements are located on Node j .

This rule can be profitably used with standard rules to push down selections and to order joins and selections so that the most selective selections are performed first, reducing the amount of data traversing the tree.

Moving unary operators close to the source of data: Unary operators are moved close to the source of the data, as suggested by the third observation, by using the following rules in addition to traditional push-down transformation rules.

$$\frac{\pi_X^h(A^k) \quad \text{where } h \neq k}{\pi_X^k(A^k)} \quad (3)$$

$$\frac{\sigma_c^h(A^k) \quad \text{where } h \neq k}{\sigma_c^k(A^k)} \quad (4)$$

Query optimization example: Let us suppose that we submit the following MW-SQL query:

```

SELECT *
FROM 1.Magnetism, 2.Acceleration, 3.Temperature
WHERE  $p_1$ (1.Magnetism) and  $p_2$ (2.Acceleration) and  $p_3$ (3.Temperature)
EVERY 1000

```

where p_1 , p_2 , and p_3 are some predicates on magnetism, acceleration and temperature readings, respectively, with probability $\Pr(p_1) = 0.01$, $\Pr(p_2) = 0.05$, $\Pr(p_3) = 0.1$, respectively. Figure 1 shows three possible equivalent query plans that can be used to process the above query. QP1 is obtained by applying the left deep join trees rule. QP2 is obtained from QP1 by using the selections push-down rule and their allocation

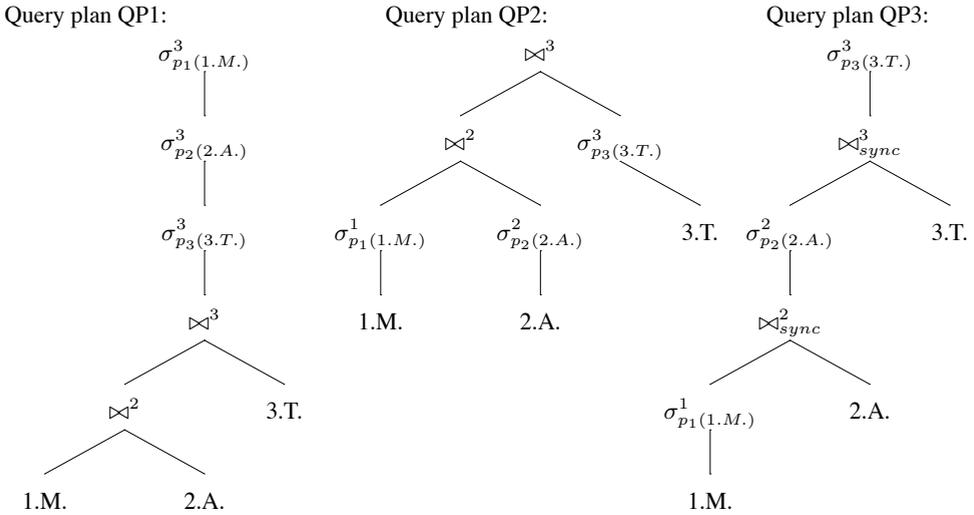


Fig. 1. Three possible execution plans for the same query.

Table 2. Costs of the three executions plans in Figure 1.

Action	Energy(mJ)	QP1:		QP2:		QP3:	
		Freq.	Power	Freq.	Power	Freq.	Power
Acquire M.	0.2685	1	0.2685	1	0.2685	1	0.2685
Send M.	0.31087	1	0.31087	0.01	0.00310	0.01	0.0031
Acquire A.	0.03222	1	0.03222	1	0.03222	0.01	0.00032
Send M.A.	0.31087	1	0.31087	0.0005	0.00016	0.0005	0.00016
Acquire T.	0.00009	1	0.00009	1	0.00009	0.0005	4.46E-08
Send M.A.T.	0.31087	5.0E-5	1.55E-05	5.0E-5	1.55E-05	5.0E-5	1.55E-05
Total Cost:			0.92256		0.30408		0.2721

on the node where data are generated. QP3 is obtained from QP2 by using rules for transforming joins into sync-joins.

As reported in Table 2, the cost of QP2 is approximately 1/3 of QP1. Cost of QP3 is a slightly better than QP2. This means that the expected lifetime of a network running QP2 or QP3 is about 3 times longer than the lifetime expected when running QP1. The lower cost of QP2 with respect to QP1 is due to the reduced number of communications that it requires. The lower cost of QP3 with respect to QP2 is due to the combined reduction of communications and acquisitions.

The performance improvement of QP3 with respect to QP2 is very limited. However, we will show that the use of sync-joins, as produced for QP3, with appropriate ordering of operators can provide significant performance improvements.

Ordering of operators: Here we discuss three different ordering criteria. Operators can be ordered so that i) more selective selections are pushed down in the tree; ii) less expansive transducers are pushed down in the tree; iii) short range communications are given priority (topological ordering).

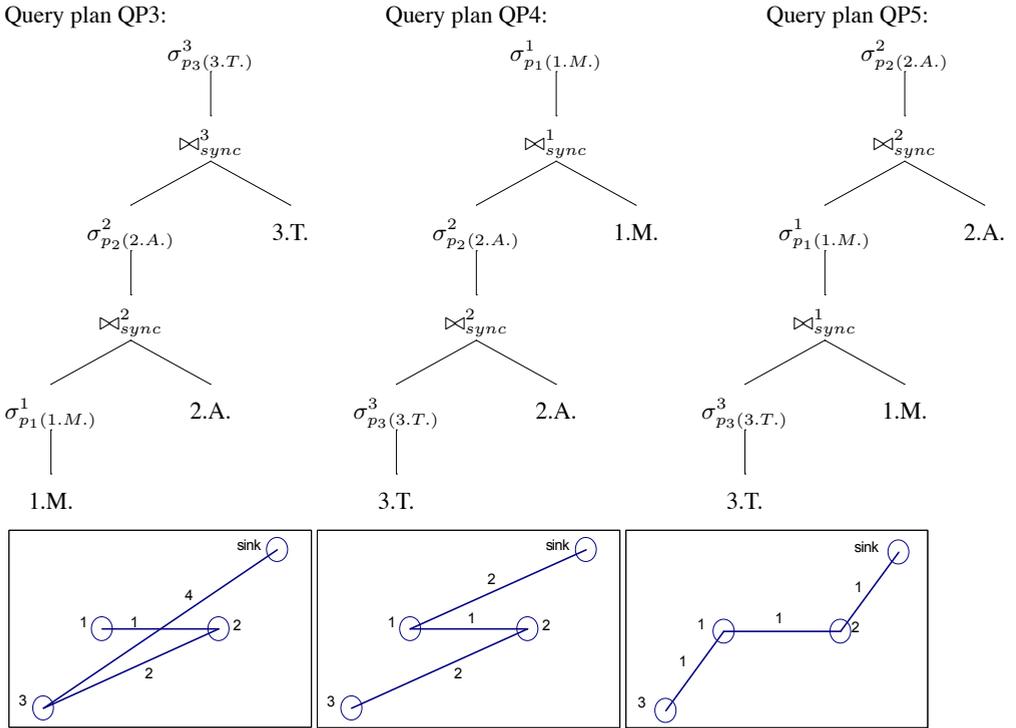


Fig. 2. Three possible execution plans for the same query using joins.

Examples of the application of these criteria are given in Figure 2. Differently from the previous section, multihop paths are taken into account here. In QP3 operators are ordered according to criterion i), criterion ii) is used in QP4, and criterion iii) is used in QP5. Their costs are given in Table 3. The cost of QP4 (0.068 mJ) is one order of magnitude smaller than the cost of QP3 (0.27 mJ). The cost of QP5 (0.058 mJ) is slightly smaller than the cost of QP4. Therefore, the expected lifetime of a network running QP4 or QP5 is about 5 times longer than running QP3.

However, this is not a proof that ordering according to the topology of the network is always the best solution. The results can vary depending on the selections selectivity and on the acquisitions costs. In general, there is not a best ordering strategy. The optimizer must generate different orderings according to the various criteria and choose the one providing the best performance. As shown in our example, this may lead to performance improvements of orders of magnitude.

5 Conclusions

In this paper we have presented a comprehensive and consistent approach to query processing in wireless sensor networks. In particular we have defined, analyzed, and discussed the aspects related to data modeling, query algebra, and query optimization.

Table 3. Cost of the query plans QP3, QP4, and QP5.

QP3:				QP4:			
Action	Energy(mJ)	Freq.	Power	Action	Energy(mJ)	Freq.	Power
Acquire M.	0.2685	1	0.2685	Acquire T.	0.00009	1	0.00009
Send M.	0.31087	0.01	0.00311	Send T.	0.62174	0.1	0.06217
Acquire A.	0.03222	0.01	0.00032	Acquire A.	0.03222	0.1	0.00322
Send M., A.	0.62174	0.0005	0.00031	Send T., A.	0.31087	0.005	0.00155
Acquire T.	0.00009	0.0005	4.46E-08	Acquire M.	0.2685	0.005	0.00134
Send M., A., T.	1.24347	0.00005	6.21E-05	Send T., A., M.	0.62174	0.00005	3.11E-05
Total Cost:			0.2723	Total Cost:			0.06841

QP5:			
Action	Energy(mJ)	Freq.	Power
Acquire T.	0.00009	1	0.00009
Send T.	0.31087	0.1	0.03109
Acquire M.	0.2685	0.1	0.02685
Send T., M.	0.31087	0.001	0.00031
Acquire A.	0.03222	0.001	0.00003
Send T., M., A.	0.31087	0.00005	1.55E-05
Total Cost:			0.05838

Our approach offers many opportunities for query optimization according to the network topology, data statistics, and types of transducers. We show that accurate query optimization may provide a reduction of the query execution cost of some orders of magnitude. Our approach separates the aspects of communication, data acquisition, data representation, and data processing, and it gives the opportunity to experiment new strategies related to each of these aspects without affecting the entire system design.

References

1. Crossbow Technology Inc., <http://www.xbow.com>.
2. MaD-WiSe: Management of Data in Wireless Sensor networks. <http://mad-wise.isit.cnr.it>.
3. TinyOS. <http://www.tinyos.net/>.
4. I. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci. A survey on sensor networks. *IEEE Communication Magazine*, 40(8):102–114, 2002.
5. C. Antonio, C. Tamalika, and C. Stefano. Bounds on Hop Distance in Greedy Routing Approach in Wireless Ad Hoc Networks. *International Journal on Wireless and Mobile Computing*. To appear.
6. P. Baronti, P. Pillai, V. Chook, S. Chessa, A. Gotta, and Y. F. Hu. Wireless sensor networks: a survey on the state of the art and the 802.15.4 and zigbee standards. Technical Report ISTI-2006-TR-18, ISTI-CNR, 2006. <http://dienst.isti.cnr.it/>.
7. E. F. Codd. A relational model of data for large shared data banks. *Commun. ACM*, 13(6):377–387, 1970.
8. ISTI-CNR, Via G. Moruzzi, 1, 56124, Pisa, IT. *SensorViz/MaD-WiSe*, version 1.3 edition, July 2006. http://mad-wise.isti.cnr.it/manual_13.pdf.

Query Answering in Expressive Variants of DL-Lite^{*}

A. Artale¹, D. Calvanese¹, R. Kontchakov² and M. Zakharyashev²

¹Faculty of Computer Science - Free University of Bozen-Bolzano - Italy

²School of Comp. Science and Inf. Sys. - Birkbeck College - London WC1E 7HX, UK

¹{lastname}@inf.unibz.it ²{roman,michael}@dcs.bbk.ac.uk

Abstract. The use of ontologies in various application domains, such as Data Integration, the Semantic Web, or ontology-based data management, where ontologies provide the access to large amounts of data, is posing challenging requirements w.r.t. the trade-off between the expressive power of a Description Logic and the efficiency of reasoning. The logics of the *DL-Lite* family were specifically designed to meet such requirements and optimized w.r.t. the data complexity of answering complex types of queries. In this paper, we propose *DL-Lite_{bool}*, an extension of *DL-Lite* with full Booleans and number restrictions, and study the complexity of reasoning in *DL-Lite_{bool}* and its significant sub-logics. We obtain our results, together with useful insights into the properties of the studied logics, by a novel reduction to the one-variable fragment of first-order logic. We study the computational complexity of satisfiability and subsumption, and the data complexity of answering positive existential queries (which extend unions of conjunctive queries). Notably, we extend the LOGSPACE upper bound for the data complexity of answering unions of conjunctive queries in *DL-Lite* to positive queries and to the possibility of expressing also number restrictions, hence local functionality.

1 Introduction

Description Logics (DLs) provide the formal foundation for ontologies, and the tasks related to the use of ontologies in various application domains are posing new and challenging requirements w.r.t. the trade-off between the expressive power of a DL and the efficiency of reasoning over knowledge bases (KBs) expressed in the DL. On the one hand, it is expected that the DL provides the ability to express TBoxes without limitations. On the other hand, tractable reasoning is essential in a context where ontologies become large and/or are used to access large amounts of data. This is a scenario emerging, e.g., in Data Integration [1], the Semantic Web [2], P2P data management [3, 4], ontology-based data

^{*} This paper is an abridged version of a paper published in the Proceedings of the 22nd National Conference on Artificial Intelligence (AAAI 2007). **Acknowledgements.** Authors partially supported by the U.K. EPSRC grant GR/S63175, the FET project TONES (Thinking ONtologies), funded within the EU 6th Framework Programme under contract FP6-7603, the KnowledgeWeb and InterOp EU funded projects and by the PRIN 2006 project NGS (New Generation Search), funded by MIUR.

access [5, 6], and biological data management. These new requirements have led to the proposal of novel DLs with PTIME algorithms for reasoning over KBs (composed of a TBox storing intensional information, and an ABox representing the extensional data), such as those of the \mathcal{EL} -family [7, 8] and of the *DL-Lite* family [9, 10].

The logics of the *DL-Lite* family, in addition to having inference that is polynomial in the size of the whole KB, have been designed with the aim of providing efficient access to large data repositories. The data that need to be accessed are assumed to be stored in a standard relational database (RDB), and one is interested in expressing, through the ontology, sufficiently complex queries to such data that go beyond the simple *instance checking* case (i.e., asking for instances of single concepts and roles). The logics of the *DL-Lite* family are tailored towards such a task, in other words, they are specifically optimized w.r.t. *data complexity*. More precisely, for the various versions of *DL-Lite*, answering conjunctive queries or their union (UCQs) [11] can be done in LOGSPACE in data complexity [9]. Indeed, the aim of the original line of research on the *DL-Lite* family was precisely to establish the maximal subset of DLs constructs for which one can devise query answering techniques that leverage on RDB technology, and thus guarantee performance and scalability (see FOL-reducibility in [9]). Clearly, a requirement for this is that the data complexity of query answering stays within LOGSPACE.

In this paper, we pursue a similar objective and aim at providing useful insights for the investigation of the computational properties of the logics in the *DL-Lite* family. We extend the basic *DL-Lite* with full Booleans and number restrictions, obtaining the logic we call *DL-Lite_{bool}*, and introduce two sublanguages of it, *DL-Lite_{krom}* and *DL-Lite_{horn}*. Notably, the latter strictly extends basic *DL-Lite* with number restrictions, and hence *local* (as opposed to global) functionality. We then characterize the first-order logic nature of this class of newly introduced DLs by showing their strong connection with the *one variable fragment* \mathcal{QL}^1 of first-order logic. The gained understanding allows us also to derive novel results on the computational complexity of inference for the newly introduced variants of *DL-Lite*.

Specifically, we show that KB satisfiability (or subsumption w.r.t. a KB) is NLOGSPACE-complete for *DL-Lite_{krom}*, P-complete for *DL-Lite_{horn}*, and NP-complete (resp. CONP-complete) for *DL-Lite_{bool}*. We prove that data complexity of both satisfiability and instance checking is in LOGSPACE for *DL-Lite_{bool}*. We then look into the data complexity of answering *positive existential queries*, which extend the well-known class of UCQs by allowing for an unrestricted interaction of conjunction and disjunction. We extend the LOGSPACE upper bound already known for UCQs in *DL-Lite* to positive existential queries in *DL-Lite_{horn}*. Due essentially to the presence of disjunction, the problem is CONP-hard for *DL-Lite_{krom}*, and hence for *DL-Lite_{bool}* [10].

The *DL-Lite_{bool}* family has been shown to be expressive enough to capture conceptual data models like UML and Extended ER [12]. Such correspondence

provided new complexity results for reasoning over various fragments of the Extended ER language.

The rest of the paper is structured as follows. In the next section we introduce the three variants of *DL-Lite* mentioned above. Then we exhibit the translation to \mathcal{QL}^1 and derive the complexity results for satisfiability and subsumption. We proceed with the analysis of data complexity, and conclude with techniques and data complexity results for answering positive existential queries. (All proofs can be found at <http://www.dcs.bbk.ac.uk/~roman.>)

2 The *DL-Lite* family

We begin by introducing the following extension *DL-Lite_{bool}* of the description logic *DL-Lite* [9, 10]. The language of *DL-Lite_{bool}* contains *object names* a_0, a_1, \dots , *concept names* A_0, A_1, \dots and *role names* P_0, P_1, \dots . Complex roles R and *concepts* C of *DL-Lite_{bool}* are defined as follows:

$$\begin{aligned} R &::= P_k \mid P_k^-, & B &::= \perp \mid A_k \mid \geq q R, \\ C &::= B \mid \neg C \mid C_1 \sqcap C_2, \end{aligned}$$

where $q \geq 1$. Concepts of the form B are called *basic concepts*. A *DL-Lite_{bool} TBox*, \mathcal{T} , consists of axioms of the form $C_1 \sqsubseteq C_2$, and an *ABox*, \mathcal{A} , of assertions of the form $A_k(a_i)$ or $P_k(a_i, a_j)$. Together \mathcal{T} and \mathcal{A} constitute a *DL-Lite_{bool} knowledge base* (KB) $\mathcal{K} = (\mathcal{T}, \mathcal{A})$. (Note that, assertions involving complex concepts $C(a_i)$ and inverse roles $P_k^-(a_i, a_j)$ can be expressed as $A_C(a_i)$, $A_C \sqsubseteq C$ and $P_k(a_j, a_i)$, respectively, where A_C is a fresh concept name.)

A *DL-Lite_{bool} interpretation* is a structure of the form

$$\mathcal{I} = (\Delta, a_0^{\mathcal{I}}, a_1^{\mathcal{I}}, \dots, A_0^{\mathcal{I}}, A_1^{\mathcal{I}}, \dots, P_0^{\mathcal{I}}, P_1^{\mathcal{I}}, \dots), \quad (1)$$

where $\Delta \neq \emptyset$, $a_i^{\mathcal{I}} \in \Delta$, $A_k^{\mathcal{I}} \subseteq \Delta$, $P_k^{\mathcal{I}} \subseteq \Delta \times \Delta$, and $a_i^{\mathcal{I}} \neq a_j^{\mathcal{I}}$, for all $i \neq j$. The role and concept constructors are interpreted in \mathcal{I} as usual:

$$\begin{aligned} (P_k^-)^{\mathcal{I}} &= \{(y, x) \in \Delta \times \Delta \mid (x, y) \in P_k^{\mathcal{I}}\}, & (\perp)^{\mathcal{I}} &= \emptyset, & (\neg C)^{\mathcal{I}} &= \Delta \setminus C^{\mathcal{I}}, \\ (\geq q R)^{\mathcal{I}} &= \{x \in \Delta \mid \#\{y \in \Delta \mid (x, y) \in R^{\mathcal{I}}\} \geq q\}, & (C_1 \sqcap C_2)^{\mathcal{I}} &= C_1^{\mathcal{I}} \cap C_2^{\mathcal{I}}. \end{aligned}$$

We also use the standard abbreviations $\exists R \equiv \geq 1 R$ and $\leq q R \equiv \neg(\geq q + 1 R)$.

The *satisfaction relation* \models is defined in the standard way:

$$\begin{aligned} \mathcal{I} \models C_1 \sqsubseteq C_2 &\text{ iff } C_1^{\mathcal{I}} \subseteq C_2^{\mathcal{I}}, \\ \mathcal{I} \models A_k(a_i) &\text{ iff } a_i^{\mathcal{I}} \in A_k^{\mathcal{I}}, \\ \mathcal{I} \models P_k(a_i, a_j) &\text{ iff } (a_i^{\mathcal{I}}, a_j^{\mathcal{I}}) \in P_k^{\mathcal{I}}. \end{aligned}$$

A KB $\mathcal{K} = (\mathcal{T}, \mathcal{A})$ is *satisfiable* if there is an interpretation, called a *model for* \mathcal{K} , satisfying all axioms of \mathcal{T} and \mathcal{A} .

We also consider two sublanguages of *DL-Lite_{bool}*: the *Krom fragment*, *DL-Lite_{krom}*, and the *Horn fragment*, *DL-Lite_{horn}* (in the following, B_i, B are basic concepts).

- A TBox of a $DL-Lite_{krom}$ KB only contains axioms of the form: $B_1 \sqsubseteq B_2$, or $B_1 \sqsubseteq \neg B_2$, or $\neg B_1 \sqsubseteq B_2$. KBs with such TBoxes will be called *Krom KBs*.
- A TBox of a $DL-Lite_{horn}$ KB only contains axioms of the form: $\prod_k B_k \sqsubseteq B$. KBs with such TBoxes will be called *Horn KBs*.

Note that the restricted negation of the original variants of $DL-Lite$ [9, 10] can only express disjointness of basic concepts, while full negation in $DL-Lite_{bool}$ allows one to define a concept as the complement of another one. In $DL-Lite_{horn}$ we can express disjointness of basic concepts by using \perp in the right-hand side of axioms. Also, the explicit functionality assertions of $DL-Lite$ (and $DL-Lite_{\mathcal{F}, \sqcap}$ in [10]) stating that some roles R are globally functional can be expressed in $DL-Lite_{bool}$ and its sublanguages $DL-Lite_{horn}$ and $DL-Lite_{krom}$ as $\geq 2 R \sqsubseteq \perp$. Moreover, *local functionality* of a role, i.e., functionality restricted to a (basic) concept B , can be expressed in $DL-Lite_{bool}$ and $DL-Lite_{krom}$ as $B \sqsubseteq \neg(\geq 2 R)$, and in $DL-Lite_{horn}$ as $B \sqcap \geq 2 R \sqsubseteq \perp$. Thus, $DL-Lite_{horn}$ strictly extends $DL-Lite$ and $DL-Lite_{\mathcal{F}, \sqcap}$ with local functionality of roles and, more generally, with number restrictions.

3 Embedding $DL-Lite$ into the one-variable fragment of first-order logic

Our main aim in this section is to show that satisfiability for $DL-Lite_{bool}$ KBs can be polynomially reduced to the satisfiability problem for the *one-variable fragment* \mathcal{QL}^1 of first-order logic without equality and function symbols.

Let $\mathcal{K} = (\mathcal{T}, \mathcal{A})$ be a $DL-Lite_{bool}$ KB. Denote by $role(\mathcal{K})$ the set of role names occurring in \mathcal{T} and \mathcal{A} , by $role^\pm(\mathcal{K})$ the set $\{P_k, P_k^- \mid P_k \in role(\mathcal{K})\}$, and by $ob(\mathcal{A})$ the set of object names in \mathcal{A} . Let $q_{\mathcal{T}}$ be the maximum numerical parameter in \mathcal{T} . Note that $q_{\mathcal{T}} \geq 2$ if the functionality axiom ($\geq 2 R \sqsubseteq \perp$) is present in \mathcal{T} . With every object name a_i in $ob(\mathcal{A})$ we associate the individual constant a_i of \mathcal{QL}^1 and with each concept name A_k the unary predicate $A_k(x)$ from the signature of \mathcal{QL}^1 . For each role $R \in role^\pm(\mathcal{K})$, we also introduce $q_{\mathcal{T}}$ fresh unary predicates $E_q R(x)$, for $1 \leq q \leq q_{\mathcal{T}}$. Intuitively, $E_1 P_k(x)$ and $E_1 P_k^-(x)$ represent the domain and range of P_k —i.e., $E_1 P_k(x)$ and $E_1 P_k^-(x)$ are the sets of points with *at least one* P_k -successor and *at least one* P_k -predecessor, respectively. Predicates $E_q P_k(x)$ and $E_q P_k^-(x)$ represent the sets of points with *at least* q distinct P_k -successors and *at least* q distinct P_k -predecessors, respectively. Additionally, for every $P_k \in role(\mathcal{K})$, we take two fresh individual constants dp_k and dp_k^- of \mathcal{QL}^1 which will serve as ‘representatives’ of the points from the domain of P_k and P_k^- , respectively (provided that they are not empty). Furthermore, for each pair of objects $a_i, a_j \in ob(\mathcal{A})$ and each $R \in role^\pm(\mathcal{K})$, we take a fresh *propositional variable* $Ra_i a_j$ of \mathcal{QL}^1 to encode $R(a_i, a_j)$. By induction on the construction of a $DL-Lite_{bool}$ concept C we define the \mathcal{QL}^1 -formula C^* :

$$\begin{aligned} (\perp)^* &= \perp, & (A_k)^* &= A_k(x), & (\geq q R)^* &= E_q R(x), \\ (\neg C)^* &= \neg C^*(x), & (C_1 \sqcap C_2)^* &= C_1^*(x) \wedge C_2^*(x), \end{aligned}$$

where A_k is a concept name and R is a role. Then a *DL-Lite_{bool}* TBox \mathcal{T} corresponds to the \mathcal{QL}^1 -sentence

$$\mathcal{T}^* = \bigwedge_{C_1 \sqsubseteq C_2 \in \mathcal{T}} \forall x (C_1^*(x) \rightarrow C_2^*(x)). \quad (2)$$

It should be also clear how to translate an ABox \mathcal{A} into \mathcal{QL}^1 :

$$\mathcal{A}^\dagger = \bigwedge_{A_k(a_i) \in \mathcal{A}} A_k(a_i) \wedge \bigwedge_{P_k(a_i, a_j) \in \mathcal{A}} P_k a_i a_j. \quad (3)$$

The following \mathcal{QL}^1 -sentences express some natural properties of the role domains and ranges: for every $R \in \text{role}^\pm(\mathcal{K})$,

$$\varepsilon(R) = \forall x (E_1 R(x) \rightarrow \text{inv}(E_1 R(dr))), \quad (4)$$

$$\delta(R) = \bigwedge_{q=1}^{q_{\mathcal{T}}-1} \forall x (E_{q+1} R(x) \rightarrow E_q R(x)), \quad (5)$$

where $\text{inv}(E_1 R(dr))$ is $E_1 P_k^- (dp_k^-)$ if $R = P_k$, and $E_1 P_k (dp_k)$ if $R = P_k^-$. Sentence (4) says that if the domain of, say, P_k is not empty then its range is not empty either: it contains the representative dp_k^- . We also need formulas relating each $R a_i a_j$ to the unary predicates for the role domain and range. For each $R \in \text{role}^\pm(\mathcal{K})$, let R^\dagger be the conjunction of the following \mathcal{QL}^1 -sentences

$$\bigwedge_{q=1}^{q_{\mathcal{T}}} \bigwedge_{\substack{a, a_{j_1}, \dots, a_{j_q} \in \text{ob}(\mathcal{A}) \\ j_i \neq j_{i'} \text{ for } i \neq i'}} \left(\bigwedge_{i=1}^q R a a_{j_i} \rightarrow E_q R(a) \right), \quad (6)$$

$$\bigwedge_{a_i, a_j \in \text{ob}(\mathcal{A})} (R a_i a_j \rightarrow \text{inv}(R) a_j a_i), \quad (7)$$

where $\text{inv}(R) a_j a_i$ is the propositional variable $P_k^- a_j a_i$ if $R = P_k$, and $P_k a_j a_i$ if $R = P_k^-$. Finally, for \mathcal{K} , we set

$$\mathcal{K}^\dagger = \left[\mathcal{T}^* \wedge \bigwedge_{R \in \text{role}^\pm(\mathcal{K})} (\varepsilon(R) \wedge \delta(R)) \right] \wedge \left[\mathcal{A}^\dagger \wedge \bigwedge_{R \in \text{role}^\pm(\mathcal{K})} R^\dagger \right].$$

It is worth noting that all of the conjuncts of \mathcal{K}^\dagger are *universal* sentences.

Theorem 1. *A DL-Lite_{bool} KB \mathcal{K} is satisfiable iff the \mathcal{QL}^1 -sentence \mathcal{K}^\dagger is satisfiable.*

The translation \mathcal{K}^\dagger of \mathcal{K} is too lengthy to provide us with reasonably low complexity results. However, we can define a more concise equi-satisfiable translation \mathcal{K}^b of \mathcal{K} , whose size is linear in the size of \mathcal{K} , *no matter whether the numerical parameters are coded in unary or in binary* (see <http://www.dcs.bbk.ac.uk/~roman> for the details).

Theorem 2. *Satisfiability is NP-complete for DL-Lite_{bool} KBs, NLOGSPACE-complete for DL-Lite_{krom} KBs and P-complete for DL-Lite_{horn} KBs.*

Many other reasoning tasks are reducible to the satisfiability problem. Consider, for example, the *subsumption problem*: given a KB \mathcal{K} and two concepts C and D , decide whether $\mathcal{K} \models C \sqsubseteq D$. The following complexity results hold:

Theorem 3. *The subsumption problem is CONP-complete for DL-Lite_{bool}, NLOGSPACE-complete for DL-Lite_{krom} and P-complete for DL-Lite_{horn}.*

4 Data complexity

In terms of the classification suggested in [13], so far we have been considering the *combined complexity* of the satisfiability problem. When the size of data is the crucial parameter (as in ontologies for huge data sets) the most relevant complexity measure becomes *data* (or *ABox*) *complexity*, where the complexity is only measured in terms of the size of the ABox \mathcal{A} , while the knowledge in the TBox \mathcal{T} is assumed to be fixed. In this section we show that as far as data complexity is concerned, reasoning problems for *DL-Lite_{bool}* KBs can be solved using only logarithmic space in the size of the ABox.

In what follows, without loss of generality, we assume that all role names of a given KB $\mathcal{K} = (\mathcal{T}, \mathcal{A})$ occur in its TBox and write $role^\pm(\mathcal{T})$ instead of $role^\pm(\mathcal{K})$. Let $\Sigma(\mathcal{T})$ be the set $\{E_1R(dr) \mid R \in role^\pm(\mathcal{T})\}$ and, for $\Sigma_0 \subseteq \Sigma(\mathcal{T})$,

$$\begin{aligned} core_{\Sigma_0}(\mathcal{T}) &= \bigwedge_{E_1R(dr) \in \Sigma_0} E_1R(dr) \quad \wedge \quad \bigwedge_{R \in role^\pm(\mathcal{T})} \left(\mathcal{T}^*[dr] \wedge \bigwedge_{R' \in role^\pm(\mathcal{T})} (\varepsilon(R')[dr] \wedge \delta^b(R')[dr]) \right), \\ proj_{\Sigma_0}(\mathcal{K}, a) &= \bigwedge_{inv(E_1R(dr)) \in \Sigma(\mathcal{T}) \setminus \Sigma_0} \neg E_1R(a) \quad \wedge \quad \mathcal{T}^*[a] \quad \wedge \quad \bigwedge_{R' \in role^\pm(\mathcal{T})} \delta^b(R')[a] \wedge \mathcal{A}^b(a), \end{aligned}$$

where $\mathcal{T}^*[c]$, $\varepsilon(R')[c]$ and $\delta^b(R')[c]$ are instantiations of the universal quantifier in the respective formulas with the constant c , and $\mathcal{A}^b(a)$ is the maximal subformula of \mathcal{A}^b containing only occurrences of predicates with a as their parameter.

Lemma 1. *\mathcal{K}^b is satisfiable iff there is $\Sigma_0 \subseteq \Sigma(\mathcal{T})$ such that $core_{\Sigma_0}(\mathcal{T})$ and the $proj_{\Sigma_0}(\mathcal{K}, a)$, for $a \in ob(\mathcal{A})$, are all satisfiable.*

Note that $core_{\Sigma_0}(\mathcal{T})$ and the $proj_{\Sigma_0}(\mathcal{K}, a)$, for $a \in ob(\mathcal{A})$, are in essence propositional Boolean formulas and their size does not depend on the size of \mathcal{A} . This is clearly the case for $core_{\Sigma_0}(\mathcal{T})$ and the first three conjuncts of $proj_{\Sigma_0}(\mathcal{K}, a)$. As for the last conjunct of $proj_{\Sigma_0}(\mathcal{K}, a)$, its length does not exceed the number of concept names in \mathcal{T} plus $q_{\mathcal{T}} \cdot |role^\pm(\mathcal{T})|$ and, therefore, only depends on the structure of \mathcal{T} . The above lemma states that satisfiability of a *DL-Lite_{bool}* KB can be checked locally: first, for the elements dr representing the domains and ranges of all roles, and second, for every object name in the ABox. This observation suggests a high degree of parallelism in the satisfiability check.

Theorem 4. *The data complexity of the satisfiability and instance checking problems for *DL-Lite_{bool}* KBs is in LOGSPACE.*

5 Query answering

By a *positive existential query* $q(x_1, \dots, x_n)$ we understand any first-order formula constructed by means of conjunction, disjunction and existential quantification starting from atoms of the form $A_k(t)$ and $P_k(t_1, t_2)$, where A_k is a concept name, P_k is a role name, and t, t_1, t_2 are *terms* taken from the list of variables y_0, y_1, \dots and the list of object names a_0, a_1, \dots , i.e.,

$$q ::= A_k(t) \mid P_k(t_1, t_2) \mid q_1 \wedge q_2 \mid q_1 \vee q_2 \mid \exists y_i q.$$

The free variables of q are called its *distinguished variables* and the bound ones its *non-distinguished variables*. We write $q(x_1, \dots, x_n)$ for a query with distinguished variables x_1, \dots, x_n . A *conjunctive query* (CQ) is a positive existential query which contains no disjunction—that is, constructed from atoms by means of conjunction and existential quantification. Given a query $q(\mathbf{x})$, with $\mathbf{x} = x_1, \dots, x_n$, and an n -tuple \mathbf{a} of object names, we write $q(\mathbf{a})$ for the result of replacing every occurrence of x_i in $q(\mathbf{x})$ with the i th member of \mathbf{a} . Queries containing no distinguished variables will be called *ground*.

Let \mathcal{I} be a *DL-Lite_{bool}* model of the form (1). An *assignment* \mathbf{a} in Δ is a function associating with every variable y an element $\mathbf{a}(y)$ of Δ . We will use the following notation: $a_i^{\mathcal{I}, \mathbf{a}} = a_i^{\mathcal{I}}$ and $y^{\mathcal{I}, \mathbf{a}} = \mathbf{a}(y)$. Define the *satisfaction relation* for positive existential formulas with respect to a given assignment \mathbf{a} :

$$\begin{aligned} \mathcal{I} \models^{\mathbf{a}} A_k(t) &\text{ iff } t^{\mathcal{I}, \mathbf{a}} \in A_k^{\mathcal{I}}, & \mathcal{I} \models^{\mathbf{a}} q_1 \wedge q_2 &\text{ iff } \mathcal{I} \models^{\mathbf{a}} q_1 \text{ and } \mathcal{I} \models^{\mathbf{a}} q_2, \\ \mathcal{I} \models^{\mathbf{a}} P_k(t_1, t_2) &\text{ iff } (t_1^{\mathcal{I}, \mathbf{a}}, t_2^{\mathcal{I}, \mathbf{a}}) \in P_k^{\mathcal{I}}, & \mathcal{I} \models^{\mathbf{a}} q_1 \vee q_2 &\text{ iff } \mathcal{I} \models^{\mathbf{a}} q_1 \text{ or } \mathcal{I} \models^{\mathbf{a}} q_2, \\ \mathcal{I} \models^{\mathbf{a}} \exists y_i q &\text{ iff } \mathcal{I} \models^{\mathbf{b}} q, & &\text{ for some } \mathbf{b} \text{ that may differ from } \mathbf{a} \text{ on } y_i. \end{aligned}$$

For a ground query $q(\mathbf{a})$ the satisfaction relation does not depend on the assignment \mathbf{a} , thus we write $\mathcal{I} \models q(\mathbf{a})$ instead of $\mathcal{I} \models^{\mathbf{a}} q(\mathbf{a})$. Given a KB $\mathcal{K} = (\mathcal{T}, \mathcal{A})$, we say that a tuple \mathbf{a} of objects from $ob(\mathcal{A})$ is an *answer* to $q(\mathbf{x})$ and write $\mathcal{K} \models q(\mathbf{a})$ if $\mathcal{I} \models q(\mathbf{a})$ whenever $\mathcal{I} \models \mathcal{K}$.

The *query answering problem* we analyse here is formulated as follows: given a *DL-Lite_{bool}* KB $\mathcal{K} = (\mathcal{T}, \mathcal{A})$, a query $q(\mathbf{x})$ and a tuple \mathbf{a} of object names from $ob(\mathcal{A})$, decide whether $\mathcal{K} \models q(\mathbf{a})$. The variant of this problem requiring to ‘list all the answers \mathbf{a} to $q(\mathbf{x})$ with respect to \mathcal{K} ’ is LOGSPACE-equivalent to the previous one [11, Exercise 16.13]. We are interested in the *data complexity* of the query answering problem. We first recall known results [14, 10, 15] for the case of conjunctive queries and obtain the following:

Theorem 5. *The data complexity of the conjunctive and positive existential query answering problems for both DL-Lite_{krorn} and DL-Lite_{bool} KBs is CONP-complete.*

Next, we show that the LOGSPACE data complexity upper bound for conjunctive queries over *DL-Lite* KBs established in [9, 10], can be extended to positive existential queries over *DL-Lite_{horn}* KBs:

Theorem 6. *The data complexity of the positive existential query answering problem for DL-Lite_{horn} KBs is in LOGSPACE.*

6 Conclusions

The LOGSPACE data complexity result for query answering provides the basis for the development of algorithms that operate on a KB whose ABox is stored in a relational database (RDB), and that evaluate a query by relying on the query answering capabilities of a RDB management system, cf. [9]. The known

algorithms for *DL-Lite* are based on rewriting the original query using the TBox axioms. We aim at developing a similar technique also for answering positive existential queries in *DL-Lite_{horn}*.

We are further investigating the complexity of logics obtained by adding further constructs to *DL-Lite*. Preliminary results show that already by adding role inclusion axioms to *DL-Lite_{bool}* the combined complexity raises to EXPTIME.

References

1. Lenzerini, M.: Data integration: A theoretical perspective. In: Proc. of the 21st ACM Symp. on Principles of Database Systems (PODS). (2002) 233–246
2. Heflin, J., Hendler, J.: A portrait of the Semantic Web in action. *IEEE Intelligent Systems* **16**(2) (2001) 54–59
3. Calvanese, D., De Giacomo, G., Lenzerini, M., Rosati, R.: Logical foundations of peer-to-peer data integration. In: Proc. of the 23rd ACM Symp. on Principles of Database Systems (PODS). (2004) 241–251
4. Franconi, E., Kuper, G.M., Lopatenko, A., Zaihrayeu, I.: Queries and updates in the coDB peer to peer database system. In: Proc. of the 30th Int. Conf. on Very Large Data Bases (VLDB). (2004) 1277–1280
5. Borgida, A., Brachman, R.J., McGuinness, D.L., Resnick, L.A.: CLASSIC: A structural data model for objects. In: Proc. of the ACM SIGMOD Int. Conf. on Management of Data. (1989) 59–67
6. Calvanese, D., De Giacomo, G., Lembo, D., Lenzerini, M., Rosati, R.: Tailoring OWL for data intensive ontologies. In: Proc. of the Workshop on OWL: Experiences and Directions (OWLED). (2005)
7. Baader, F., Brandt, S., Lutz, C.: Pushing the \mathcal{EL} envelope. In: Proc. of the 19th Int. Joint Conf. on Artificial Intelligence (IJCAI). (2005) 364–369
8. Baader, F., Lutz, C., Suntisrivaraporn, B.: Is tractable reasoning in extensions of the description logic \mathcal{EL} useful in practice? In: Proc. of the Methods for Modalities Workshop (M4M 2005). (2005)
9. Calvanese, D., De Giacomo, G., Lembo, D., Lenzerini, M., Rosati, R.: DL-Lite: Tractable description logics for ontologies. In: Proc. of the 20th Nat. Conf. on Artificial Intelligence (AAAI). (2005) 602–607
10. Calvanese, D., De Giacomo, G., Lembo, D., Lenzerini, M., Rosati, R.: Data complexity of query answering in description logics. In: Proc. of the 10th Int. Conf. on the Principles of Knowledge Representation and Reasoning (KR). (2006) 260–270
11. Abiteboul, S., Hull, R., Vianu, V.: Foundations of Databases. Addison Wesley (1995)
12. Artale, A., Calvanese, D., Kontchakov, R., Ryzhikov, V., Zakharyashev, M.: Complexity of reasoning in Entity Relationship models. In: Proc. of the 2007 Description Logic Workshop (DL). (2007)
13. Vardi, M.Y.: The complexity of relational query languages. In: Proc. of the 14th ACM SIGACT Symp. on Theory of Computing (STOC). (1982) 137–146
14. Donini, F.M., Lenzerini, M., Nardi, D., Schaerf, A.: Deduction in concept languages: From subsumption to instance checking. *J. of Logic and Computation* **4**(4) (1994) 423–452
15. Ortiz, M.M., Calvanese, D., Eiter, T.: Characterizing data complexity for conjunctive query answering in expressive description logics. In: Proc. of the 21st Nat. Conf. on Artificial Intelligence (AAAI). (2006)

Towards an Effective Semi-Automatic Technique for Image Annotation^{*}

Ilaria Bartolini and Paolo Ciaccia

DEIS, University of Bologna, Italy
 {ibartolini,pciaccia}@deis.unibo.it

Abstract. In this paper we explore the opportunities offered by graph-based link analysis techniques in the development of a semi-automatic image captioning system. The approach we propose is appealing since predicted terms for an image: 1) are in variable number, depending on the image content, 2) represent correlated terms, and 3) can also represent abstract concepts. We present preliminary results on our prototype system and discuss possible extensions.

1 Introduction

The advent of digital photography calls for effective techniques for managing growing amounts of color images. Even if content-based image retrieval (CBIR) systems represent a completely automatic solution to image retrieval [12], low level features, such as color and texture, are not always able to properly characterize the actual image content. This is due to the semantic gap existing between the user subjective notion of similarity and the one according to which a low level feature-based retrieval system evaluates two images to be similar. An effective way to alleviate such gap is to exploit user feedback to understand which images are actually relevant to the query [11, 3, 1]. However, the quality of results for queries in which the user is looking for images matching some high-level concept (e.g., landscape) is still far to reach the optimal 100% precision value.

A possible way to fill the semantic gap is to (semi-)automatically assign meaningful terms to images, so as to indeed allow a high-level, concept-based, retrieval. Several (semi-)automatic techniques [10, 5, 7, 9, 8] have been proposed in recent years and the first image annotation prototypes are now available on Internet (e.g., ALIPR.com¹ and Behold²). We can group state-of-the-art solutions into two main classes, namely *semantic propagation* and *statistical inference*. In both cases, the problem to be solved remains the same: *Given a training set of annotated color images, discover affinities between low-level image features and terms that describe the image content, with the aim of predicting “good” terms to annotate a new image.*

^{*} This work is partially supported by a Telecom Italia grant.

¹ ALIPR.com: <http://www.alipr.com/>.

² Behold: <http://go.beholdsearch.com/searchvis.jsp>.

With propagation models [8], a supervised learning technique that compares image similarity at a low-level and then annotates images by propagating terms over the most similar images is adopted. Working with statistical inference models [9, 5, 7, 10], an unsupervised learning approach tries to capture correspondences between low-level features and terms by estimating their joint probability distribution. Both approaches improve the annotation process and the retrieval on large image databases. However, among the predicted terms for unlabelled images, still too many irrelevant ones are present.

In this paper we explore the opportunities offered by graph-based link analysis techniques in the development of an effective semi-automatic image captioning system. In our approach each image is characterized as a set of *regions* from which low-level features are extracted. The training set is built by associating a variable number of terms to each image. In this way, not only terms related to a particular region of the image, but even abstract concepts associated to the whole image (e.g., “landscape” and “pasture”) are possible.

We turn the annotation problem into a set of *graph-based* problems. First, we try to discover *affinities* between terms and an unlabelled image, which is done using a *Random Walk with Restart* (RWR) algorithm on a graph that models current annotations as well as regions’ similarities. Then, since the RWR step might predict unrelated, or even contradictory, terms, we compute pairwise *term correlations*. Again, this relies on the analysis of links in a (second-order) graph. Finally, we combine the results of the two steps to derive a set of terms which are both *semantically correlated* each other and affine to the new image. This final step amounts to solve an instance of the Maximum Weight Clique Problem (MWCP) on a small graph. Doing this way, the number of terms we predict for each new image is variable, and dependent on the actual image content.

The paper is organized as follows: In Section 2 we define the problem. Section 3 shows how to compute affinities between an image and the terms of the training set and Section 4 analyzes correlations of terms. In Section 5 we show how we derive the most correlated affine terms and provide some preliminary results obtained from our prototype system. Section 6 concludes and discusses possible extensions.

2 Problem Definition

Before presenting our image annotation technique, we need to precisely define the problem. We are given a dataset of N manually annotated images that constitute the image *training set* \mathcal{I} . Each image $I_i \in \mathcal{I}$ is characterized as a set of *regions* R_j , for each of which a D -dimensional feature vector is automatically extracted. For instance, features could represent the color and the texture of R_j [2]. Moreover, each image $I_i \in \mathcal{I}$ is manually annotated with m_i *terms* $\{T_{i_1}, \dots, T_{i_{m_i}}\}$. Thus, each image I_i is represented as $I_i = (\{R_{i_1}, \dots, R_{i_{n_i}}\}, \{T_{i_1}, \dots, T_{i_{m_i}}\})$.

Problem 1 *Given an unlabelled (or query) image I_q , with regions $\{R_{q_1}, \dots, R_{q_{n_q}}\}$, exploit the knowledge of images in \mathcal{I} to predict a “good” set of terms $\{T_{q_1}, \dots, T_{q_{m_q}}\}$ able to effectively characterize the content of I_q .*

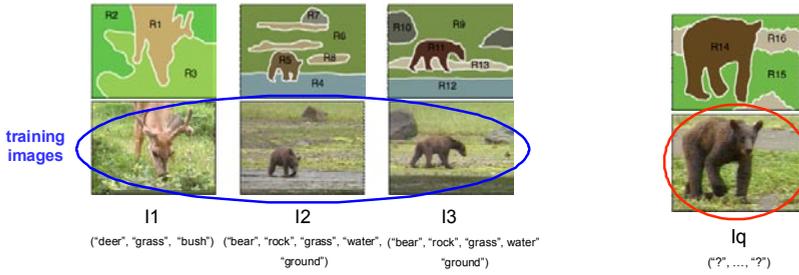


Fig. 1. Visual example of the image annotation problem.

We turn the annotation problem, an instance of which is depicted in Figure 1, into a *graph*-based problem that is split into three main steps:

1. **Affinities of terms and query image:** Starting from the training images \mathcal{I} , we build a graph G_{MMG} and “navigate” it so as to establish possible *affinities* between the query image I_q and the terms associated to images in \mathcal{I} .
2. **Correlation of terms:** Starting from G_{MMG} , we derive a *second-order* graph G_T^2 from which to compute the *similarity* among terms.
3. **Correlated affine terms:** In this step we combine the results of the first two steps and derive the set of most *semantically correlated* terms to label the query image I_q .

3 Affinities of terms and query image

As for the implementation of the 1st step, we follow the Mixed Media Graph approach [10].

Graph Construction. The Mixed Media Graph (MMG) $G_{MMG} = (V, E)$ is a 3-level undirected graph, where each node represents an image (identifier), a region, or a term, in the training set. More precisely, if \mathcal{T} is the set of terms and \mathcal{R} is the set of regions, then $V = \mathcal{I} \cup \mathcal{T} \cup \mathcal{R}$. Edges in E are of two types. An *object-attribute-value* (OAV) edge connects an image node with either a region or a term node. Therefore for each image $I_i \in \mathcal{I}$, there are edges (I_i, R_j) for all regions R_j in I_i , and similarly for terms. *Nearest neighbor* (NN) edges connect a region to its k ($k \geq 1$) nearest neighbors regions in \mathcal{R} , where the similarity between two regions is computed based on the regions’ feature vectors. The graph G_{MMG} can be extended, so as to account for a new unlabelled image I_q , into the graph $G_q = (V_q, E_q)$ by adding nodes for I_q and its regions, and NN edges for the regions of I_q . Figure 2 shows the G_q graph for the example in Figure 1.

Graph Navigation. As we turn the annotation problem into a graph problem, methods for determining how related a node X is to a “start” node S are needed to establish the affinity between the query image I_q and the terms in G_{MMG} . For this task we find appropriate to adopt the *random walk with restart*

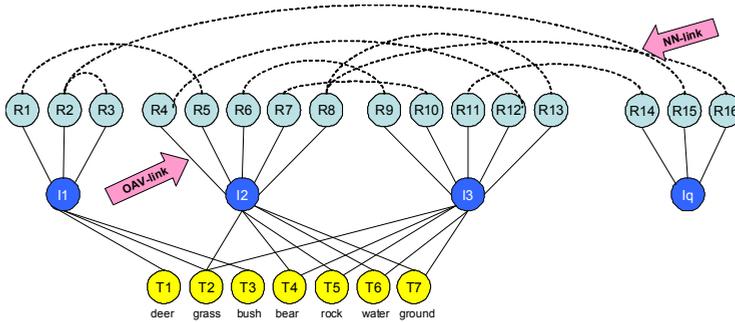


Fig. 2. The G_q graph for the example depicted in Figure 1, assuming $k = 1$

(RWR) technique [10]. The basic idea of RWR is to consider a random walker that starts from node S and at each step chooses to follow an edge, randomly chosen from the available ones. Further, at each step, with probability p the walker can go back to node S (i.e., it *restarts*). The *steady state probability* that the random walker will find itself at node X , denoted $u_S(X)$, can be interpreted as a measure of affinity between X and S . In our case it is $S = I_q$ and relevant steady state probabilities are only those of term nodes (i.e., $X \in \mathcal{T}$). Intuitively, if $u_{I_q}(T_j)$ is high, this is an evidence that T_j is a good candidate for annotating I_q . Details on how the steady state probabilities can be efficiently computed even for large graphs can be found in [13].

Limits of MMG. Even if MMG with RWR is usually able to find some relevant terms for annotating a query image, it suffers some limits. First of all, the predicted terms are those that have been crossed most frequently during the graph navigation. It can be argued that using only frequency to evaluate the relevance of each term for annotating a new image is rather imprecise. For instance, when using MMG, querying our prototype system with an image representing a “horse” often returned as result the term “cow”. Indeed, one should bear in mind that the MMG + RWR method heavily relies on the NN edges involving the regions of I_q , thus on low-level similarities. If a region R_{q_i} of I_q is (highly) similar to a region R_j of an image I , which however has some terms unrelated to I_q , this might easily lead to have such terms highly scored by RWR.

Another shortcoming of MMG regards the number of terms, PT , with the highest steady state probabilities that are to be used for annotation. There are two alternatives here. If one insists to take only the best PT terms, then each image will be annotated with a same number of terms, thus independently of the actual image content. Note that setting PT to a high value might easily lead to wrong annotations, whereas a low value might easily miss relevant terms. The same problem would occur should the predicted terms be all those whose steady state probability exceeds a given threshold value.

4 Analyzing Correlations of Terms

The approach we take to overcome MMG limitations is to perform a link analysis on a sub-graph of G_{MMG} so as to find highly-correlated terms. In turn, this is evidence that such terms are also semantically related, thus good candidates to annotate a new image.

Link Analysis. From the graph $G_{MMG} = (V, E)$, we derive the sub-graph $G_T = (V_T, E_T)$, where $V_T = \mathcal{I} \cup \mathcal{T}$, i.e., G_T is derived from G_{MMG} by deleting region nodes. With the aim of estimating the similarity between couples of terms, we derive from G_T a *second-order* (bipartite) graph $G_T^2 = (V_T^2, E_T^2)$. A node in V_T^2 is either a pair of images (I_i, I_j) , $I_i, I_j \in \mathcal{I}$, or a pair of terms (T_r, T_s) , $T_r, T_s \in \mathcal{T}$. An edge between nodes (I_i, I_j) and (T_r, T_s) is added to E_T^2 iff the two edges (I_i, T_r) and (I_j, T_s) (equivalently, (I_i, T_s) and (I_j, T_r)) are both in E_T . This is to say that each image I_i and I_j contains (at least) one of the two terms, and the two images, when taken together, contain both terms. Notice that when $I_i = I_j$, then terms T_r and T_s appear together in image I_i .

Given the second-order graph G_T^2 , the problem of estimating the correlation of two terms transforms into the problem of assigning a score to nodes corresponding to pairs of terms. For this one can use any link-based algorithm, such as those adopted for ranking Web pages [6]. We denote with $corr(T_r, T_s)$ the (correlation) score computed by such an algorithm for the node in V_T^2 corresponding to the pair of terms (T_r, T_s) . Note that this step can be performed off-line, since it is independent of the query image.³

5 Putting it All Together

In this last step we combine the results of the previous phases. As to the output of the MMG + RWR step, we always take the set of PT terms with the highest steady state probabilities, $\mathcal{T}_{MMG} = \{T_1, \dots, T_{PT}\}$. This will be possibly reduced considering terms correlations, $corr(T_r, T_s)$, so as to obtain a set of terms to annotate the query image I_q that: 1) are affine to I_q , and, at the same time, 2) are all tightly correlated each other.

We solve the problem by modelling it as an instance of the Maximum Weight Clique Problem (MWCP) [4]:

Definition 1 (MWCP) *Let $G = (V, E, w)$ be an undirected and weighted graph, where the j -th component of the weight vector w is the weight of the j -th node in V . A clique $G' = (V', E')$ is a complete sub-graph of G , i.e., $V' \subseteq V$, and there is an edge in E' between every pair of nodes in V' . The weight of clique G' is the sum of weights of the nodes in V' , $W(G') = \sum_{j \in V'} w_j$. The Maximum Weight Clique Problem (MWCP) is to find the clique, G'_{max} , with the maximum weight.*

³ We are currently studying how correlations can be efficiently updated in front of insertions in the training set.

The correspondence with our problem is almost immediate. The set of nodes in the graph consists of the terms in \mathcal{T}_{MMG} (i.e., $V = \mathcal{T}_{MMG}$), and each node T_j is weighted by its steady state probability $u_{I_q}(T_j)$ (i.e., $w_j = u_{I_q}(T_j)$). As to edges, we only add to E those between nodes (terms) whose correlation exceeds a given threshold value c , i.e., $(T_r, T_s) \in E$ iff $corr(T_r, T_s) > c$. Doing this way, solving the MWCP amounts to find the subset \mathcal{T}_{OPT} of *optimal terms* in \mathcal{T}_{MMG} such that: 1) all terms in \mathcal{T}_{OPT} are highly correlated, and 2) there is no other set of terms satisfying the same condition whose global affinity is higher.⁴

To give an example, Figure 3 shows a sample graph in which $PT = 6$. Numbers within each node represent unnormalized steady state probabilities (normalizing would not change the net effect). Solving MWCP, the optimal terms (maximum weight clique) turn to be $\mathcal{T}_{OPT} = \{grass, bear, ground, water\}$. Notice that, without taking into account terms correlations, the affinity of *rock* is higher than that of *water*. However, *rock* is loosely correlated with almost all terms, thus does not enter into the solution.

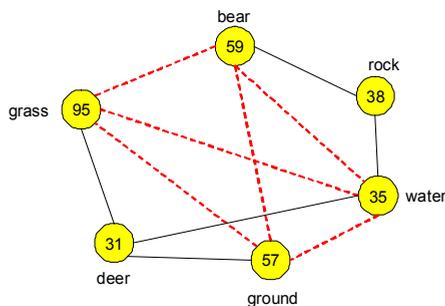


Fig. 3. Dashed edges define the clique with the maximum weight.

We have implemented all above-described algorithms on top of the Windsurf system [2]. In details, each image is automatically segmented into a set of homogeneous regions which convey information about color and texture features. Each region corresponds to a cluster of pixels and is represented through a 37-dimensional feature vector. With respect to regions comparison (thus, to define the NN edges of G_{MMG}) the Bhattacharyya metric is used. The dataset we used was extracted from the IMSI collection.⁵ We trained our prototype system by manually annotating about 50 images with one, two, or three terms. Table 1 summarizes the parameters used by our system, together with their default values which we used in our preliminary experiments.

⁴ Although the MWCP problem is NP-hard, the graphs we deal with are rather small (e.g., tens of nodes), so the computational overhead is negligible.

⁵ IMSI MasterPhotos 50,000: <http://www.imsisoft.com/>.

parameter	default value
Average number of regions per image	4.4
Number of NN edges per region	$k = 5$
Maximum number of terms per image	$PT = 6$
RWR restart probability	$p = 0.8$
Correlation threshold	$c = 0.3$

Table 1. Parameters used by our system together with their default values.

Figure 4 shows an example of our prototype system in action. In this case, the optimal terms that our system returns are *sheep* and *grass*, which are indeed the only appropriate ones among the $PT = 6$ predicted by MMG.

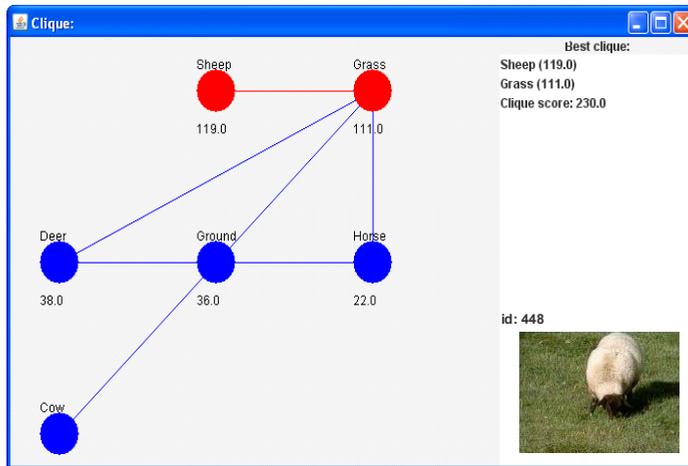


Fig. 4. The maximum weight clique for the image on the right.

We experimentally observed a similar accuracy for most of the images we tried (about 50). Although it happened sometimes that optimal terms also included irrelevant ones, the average precision of the result was always better than that of MMG alone, thus validating the effectiveness of correlation analysis.

6 Conclusions and Future Directions

In this paper we have presented an effective solution for semi-automatic image annotation based on link analysis techniques. Our approach is able to predict terms for unlabelled images that are highly correlated each other, which improves the accuracy of the annotation. Admittedly, our experimental results are preliminary, thus we are currently working on a more accurate evaluation. Further, we

plan to extend our term analysis by means of ontologies, so as to exploit, besides term correlations, also their semantic relationships (e.g., “the sheep browses on grass”). This will likely lead to further improve the precision of our approach.

References

1. I. Bartolini. Context-Based Image Similarity Queries. *Adaptive Multimedia Retrieval: User, Context, and Feedback, AMR 2005, Revised Selected Papers (Lecture Notes in Computer Science)*, 3877:222–235, 2006.
2. I. Bartolini, P. Ciaccia, and M. Patella. A Sound Algorithm for Region-Based Image Retrieval Using an Index. In *Proceedings of the 4th International Workshop on Query Processing and Multimedia Issue in Distributed Systems (QPMIDS 2000)*, pages 930–934, Greenwich, London, UK, Sept. 2000.
3. I. Bartolini, P. Ciaccia, and F. Waas. FeedbackBypass: A New Approach to Interactive Similarity Query Processing. In *Proceedings of the 27th International Conference on Very Large Data Bases (VLDB 2001)*, pages 201–210, Rome, Italy, Sept. 2001.
4. I. Bomze, M. Budinich, P. Pardalos, and M. Pelillo. *The Maximum Clique Problem*, volume 4. Kluwer Academic Publishers, Boston, MA, 1999.
5. P. Duygulu, K. Barnard, J. F. G. de Freitas, and D. A. Forsyth. Object Recognition as Machine Translation: Learning a Lexicon for a Fixed Image Vocabulary. In *Computer Vision - ECCV 2002, 7th European Conference on Computer Vision*, pages 97–1123, Copenhagen, Denmark, May 2002.
6. D. Fogaras and B. Rácz. Scaling Link-based Similarity Search. In *Proceedings of the 14th International Conference on World Wide Web (WWW 2005)*, pages 641–650, Chiba, Japan, May 2005.
7. J. Jeon, V. Lavrenko, and R. Manmatha. Automatic Image Annotation and Retrieval Using Cross-media Relevance Models. In *Proceedings of the 26th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 119–126, Toronto, Canada, Aug. 2003.
8. O. Maron and A. L. Ratan. Multiple-instance Learning for Natural Scene Classification. In *Proceedings of the 15th International Conference on Machine Learning (ICML 1998)*, pages 341–349, San Francisco, CA, USA, July 1998.
9. Y. Mori, H. Takahashi, and R. Oka. Image-to-word Transformation Based on Dividing and Vector Quantizing Images with Words. In *Proceedings of the 1st International Workshop on Multimedia Intelligent Storage and Retrieval Management (MISRM 1999)*, 1999.
10. J.-Y. Pan, H.-J. Yang, C. Faloutsos, and P. Duygulu. Automatic Multimedia Cross-modal Correlation Discovery. In *Proceedings of the 10th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 653–658, Seattle, USA, Aug. 2004.
11. Y. Rui, T. S. Huang, M. Ortega, and S. Mehrotra. Relevance Feedback: A Power Tool for Interactive Content-Based Image Retrieval. *IEEE Transactions on Circuits and Systems for Video Technology*, 8(5):644–655, 1998.
12. A. W. M. Smeulders, M. Worring, S. Santini, A. Gupta, and R. Jain. Content-Based Image Retrieval at the End of the Early Years. *IEEE Trans. Pattern Anal. Mach. Intell.*, 22(12):1349–1380, 2000.
13. H. Tong, C. Faloutsos, and J.-Y. Pan. Fast Random Walk with Restart and Its Applications. In *Proceedings of the 6th IEEE International Conference on Data Mining (ICDM 2006)*, pages 613–622, Hong Kong, China, Dec. 2006.

A New Type of Metadata for Querying Data Integration Systems

Sonia Bergamaschi¹, Francesco Guerra², Mirko Orsini¹, and Claudio Sartori³

¹ DII-Università di Modena e Reggio Emilia
via Vignolese 905, 41100 Modena

² DEA-Università di Modena e Reggio Emilia
v.le Berengario 51, 41100 Modena, Italy

³ DEIS-Università di Bologna

Viale Risorgimento 2, 40136 Bologna, Italy

{bergamaschi.sonia, guerra.francesco, orsini.mirko}@unimore.it,
claudio.sartori@unibo.it

Abstract. Research on data integration has provided languages and systems able to guarantee an integrated intensional representation of a given set of data sources. A significant limitation common to most proposals is that only intensional knowledge is considered, with little or no consideration for extensional knowledge.

In this paper we propose a technique to enrich the intension of an attribute with a new sort of metadata: the “relevant values”, extracted from the attribute values. Relevant values enrich schemata with domain knowledge; moreover they can be exploited by a user in the interactive process of creating/refining a query. The technique, fully implemented in a prototype, is automatic, independent of the attribute domain and it is based on data mining clustering techniques and emerging semantics from data values. It is parametrized with various metrics for similarity measures and is a viable tool for dealing with frequently changing sources.

1 Introduction

Integration of data from multiple sources is one of the main issues facing the database and artificial intelligence research communities. A common approach for integrating information sources is to build a mediated schema as a synthesis of them. By managing all the collected data in a common way, a mediated schema allows the user to pose a query according to a global perception of the handled information. A query over the mediated schema is translated into a set of sub-queries for the involved sources by means of automatic unfolding-rewriting operations taking into account the mediated and the sources schemata. Results from sub-queries are finally unified by data reconciliation techniques (see [9, 1] for an overview).

Research on data integration has provided languages and systems able to guarantee an integrated representation of a given set of data sources. A significant limitation common to most proposals is that only intensional knowledge is considered, with little or no consideration for extensional knowledge.

In this paper, we describe a technique for providing metadata related to attribute values. Such metadata represent a synthesized and meaningful information emerging

from the data. We call these metadata “relevant values” as they provide the users with a synthetic description of the values of the attribute which refer to by representing with a reduced number of values its domain. We claim that such metadata are useful for querying an integrated database, since integration puts together in the same global class a number of local *semantically similar* classes coming from different sources and a set of global attributes which generalize the local classes. Consequently, the name/description of a global class/global attribute is often generic and this fact could significantly limit the effectiveness of querying. Let us suppose, for instance, that the user has a good knowledge of a single source, say “S”, and that she/he is interested in items whose global attribute “A” contains the word “x”, as of terminology of source “S”. The user could completely miss the fact that in source “T” the word “y” refers to a very similar concept, and therefore a query with target “x” would return only a partial result, w.r.t. the contents of the global class. Moreover, ignoring the values assumed by a global attribute may generate meaningless, too selective or empty queries. On the other hand, knowing all the data collected from a global class is infeasible for a user: databases contain large amount of data which a user cannot deal with. A metadata structure derived from an analysis of the attribute extension could be of great help in overcoming such limitation.

This work is done in the context of the MOMIS (Mediator envirOnment for Multiple Information Sources) project⁴ [4], a framework to perform information extraction and integration from both structured and semi-structured data sources, plus a query management environment able to process incoming queries through the navigation of the mediated schema. The MOMIS integration process gives rise to a Global Virtual View (GVV) in the form of Global Classes and global attributes of the a set of data sources. In [3], we proposed a partial solution to the semantic enrichment of a GVV by providing a semantic annotation of all the Global Classes of the GVV with respect to the WordNet lexical database⁵, and thus providing each term with a well-understood meaning. Relevant Values will semantically enrich a GVV, since they provide semantic information about the data sources the GVV refers to. Moreover, in [2] a first heuristic for calculating relevant values was described.

In this paper we improve the approach proposed in [2], by providing a flexible parametric technique to deal with string attributes. It is not a severe limitation, as: (1) data coming from web-site wrappers are generally represented as strings; (2) several techniques have been developed in literature for clustering numeric values where it is easy to define element orderings (see [8] for a survey). The method was implemented in a prototype called *RELEVANT* (RELEVant VALue geNeraTor) we describe in section 3.

The outline of the paper is the following: next section defines the technique to elicit relevant values for a selected attribute, section 3 describe the implemented prototype and section 4 describes how relevant values may be exploited for querying data sources. Finally section 5 sketches out some conclusions and future works.

⁴ See <http://www.dbgroup.unimo.it> for more publications about the project.

⁵ <http://wordnet.princeton.edu/>

2 Eliciting Relevant Values from Data

There are several models for representing knowledge bases. Without loss of generality, let us refer to the concepts of MOMIS. The Global Virtual View built with MOMIS is composed of Global Classes, with Global Attributes (GA). Our goal is to extract the relevant values of a GA. Each relevant value is described by a relevant value name and a set of values of the attribute domain.

The idea is that analyzing an attribute domain, we may find values which may be clustered because *strongly related*. Providing a name to these clusters, we may refer to a relevant value name which encompasses a set of values. More formally, given a class C and one of its attributes At , a **relevant value** for it, rv^{At} is a pair $rv^{At} = \langle rvn^{At}, values^{At} \rangle$. rvn^{At} is the name of the relevant value set, while $values^{At}$ is the set of values referring to it.

Now we should answer two questions: how can we cluster the values of the domain in order to put together in a relevant value a set of values which are strongly related? How can we choose the relevant value names? The first question will be answered by means of clustering techniques, adapted to the problem on hand; the second will require the intervention of the designer, but we will provide, in section 3.4, an effective *assistant*.

Like most cluster tasks with non-numeric attributes, the problems are related to find an effective representation of the points (i.e. the attribute values) in a space, and to devise a suitable similarity function to be exploited by the clustering algorithm. The technique we propose builds a binary representation of the attribute values, and exploits two different kinds of measure to build some structure upon the flat set binary representation: 1) the *syntactic* similarity, mapping all the words of the attribute values in an abstract space, and defining a syntactic similarity function in such space; and 2) the *domination* measure, expressed by the root elements described later on. Such measures are automatically extracted: the manual annotation of each attribute value (e.g. with reference to a given ontology) would be a time-consuming and error-prone operation also discouraged by the high number of values and the update frequency.

The similarity measures we propose are then used by a clustering algorithm (in *RELEVANT* the user may generate both partitions and overlapped clusters). The clusters of values produced are so far called *sets of relevant values*. The user may balance the weight of the two different similarity measures.

2.1 The syntactic similarity

Terms related to the same object may have the same etymology and then share a common root: several similarity measures are based on this idea (e.g. the Levenshtein distance and the other metrics derived from it). In the same way, we may assume that related attribute values share terms. By means of this measure, we group different attribute values sharing common terms.

It is trivial to show that a term may be polysemous, i.e. it may be used in different attribute values with different meanings, especially in multi-word values. In our experience, the syntactic similarity alone could not be sufficient, but in conjunction with the similarities described below, it provides satisfactory results.

2.2 Domination: the root elements

A similarity measure may be extracted from the *Domination* relationships between the attribute values. Considering two attribute values a_1 and a_2 , we say that a_1 dominates a_2 if a_1 is more “general” than a_2 . Any partial order on attribute values could be used to define domination. On the basis of an analysis of several databases, we observed that it is frequent to have string domains with values composed of many words, also with abbreviations. We observed also that the same word, or group of words, may be further qualified (i.e. specialized) with multiple words in many ways. For example, the attribute describing a kind of production for a mechanical enterprise may contain the value “Mould” and the values “Mould ejectors, Mould engineering, ...”. Thus, we approximate the domination between attribute values, a semantic property, with the *Contains* function, a syntactic property. *Contains* is a function based on string containment: $Contains(X, Y) = true$ iff $stem(X) \supseteq stem(Y)$, where X and Y are sets of words and *stem* is a *stemming operator* for words⁶. Then we say that Y dominates X if it is contained in X . The domination is a partial order and can be represented by an oriented graph. Let us say, for instance, that an edge goes from the dominating value (more general) to the dominated one (more specific). The integration designer should verify how much the graph represents the general notion of a value being “more general” than another, but in our experience the graph is usually sound.

Our idea is to exploit the domination for building clusters of values around *root elements*. A root element is an attribute value with only outgoing edges in the domination graph, and can be taken as a representative of the cluster composed by the nodes recursively touched by its outgoing edges.

3 The *RELEVANT* prototype

RELEVANT is a software tool for calculating relevant values. Giving as input a list of attribute values, *RELEVANT* generates a set of relevant values according to the designer selections. Figure 1 shows the *RELEVANT* functional architecture, which is organized into five blocks:

1. **Data pre-processing:** two binary representations of the values of an attribute are obtained with two matrices representing the different kinds of similarity measure.
2. **Similarity Computation:** the designer selects how to measure the similarity (metrics selection) and which kinds of similarity are used (the syntactic similarity, the domination measure or a combination of the two).
3. **Clustering technique selection:** we implement some clustering algorithms to compute the set of relevant values on the basis of the choices made at step 2.
4. **Name selection:** for each group of values defined in step 3, a name representative of all the values has to be identified.
5. **Validation:** we implemented some standard techniques to evaluate cluster quality. Additional work is necessary to go beyond the simple evaluation, so as to provide effective assistance to the designer in the critical task of parameter configuration.

⁶ a standard operator in natural language processing.

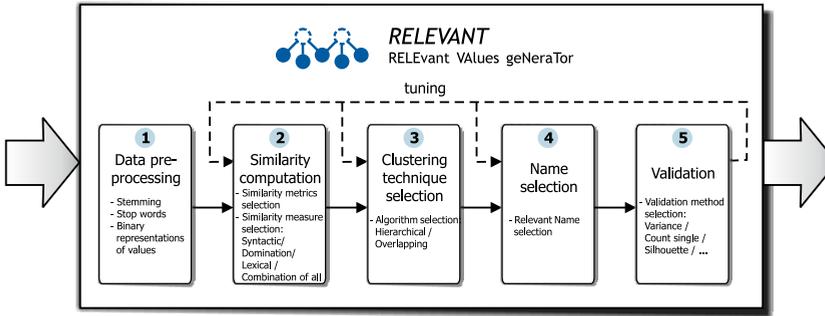


Fig. 1: The *RELEVANT* functional architecture

3.1 Step 1: Binary Representation of attribute values

The *RELEVANT* starting point is the creation of two binary matrices, according to the different measures introduced in section 2: *MT* and *MTR*.

The *Syntactic Matching Table (MT)* is a binary representation of all the values of an attribute *At* w.r.t. the universe of words considered (i.e. is the union of the words included in the extension of *At*). Notice that multi-word attributes contribute to the universe of words with multiple words. *MT* is typically sparse: for each row there is a number of elements different from zero that is equal to the number of words contained in the associated attribute, except for the stop-words.

The *Root Elements Matching Table (MTR)* shows the root elements associated to the attribute values: each column of the matrix is a root element and the rows are the attribute values.

3.2 Step 2: Similarity computation

Two tasks are executed in this step: the selection of the metrics for computing the similarity on the matrices created in the previous step and the computation of the affinity matrices *AM* and *AMR* derived from the matching tables *MT*, *MTR* respectively.

Concerning the first task, the tool implements some of the metrics commonly adopted in information retrieval (Simple Matching, Russel & Rao measure, Tanamoto Coefficient, Sorensen measure, Jaccard's Similarity [11]). Due to the sparseness of the binary matrix, the Jaccard similarity metric, which only considers the positive values in both the attribute value representations⁷, is used in this paper and set as default.

Concerning the second task, two new matrices express the two different affinity measures calculated by applying the selected similarity metrics on *MT* and *MTR*. The

⁷ Let us define B_{11} as the total number of times a bit is ON in both bit strings, B_{00} as the total number of times a bit is OFF in both bit strings, and L as the length of the bit string, the Jaccard metric is defined as $B_{11}/(L - B_{00})$

matrices are built as follows. Given a matrix $AM(AMR)$ a generic element $e_{i,j}$ is obtained computing the similarity between the e_i and e_j rows of the matrix $MT(MTR)$, on the basis of the selected metrics.

Finally AM and AMR are linearly combined into the Global Affinity Matrix $GAM = \|gam_{hk}\|$. An element $gam_{hk} = lc_y \times am_{hk} + lc_m \times amr_{hk}$, where the values of lc_y and lc_m are chosen by the designer such that $lc_y, lc_m \in [0, 1]$ and $lc_y + lc_m = 1$.

3.3 Step 3: Clustering technique selection

The prototype implements two different clustering algorithms: a classical agglomerative hierarchical clustering algorithm performs a partition of the values set, a second algorithm generates overlapping clusters (a variation of the algorithm in [5] is implemented).

The hierarchical clustering algorithm. A hierarchical clustering algorithm classifies elements into groups at different levels of affinity, forming a tree [6]. The hierarchical clustering procedure is applied to the matrix GAM . Once the affinity tree has been built, clusters are interactively computed on the basis of the numerical affinity values in the affinity tree and a threshold-based mechanism for cluster selection specified by the designer. High values of threshold return small, highly fragmented clusters. By decreasing the threshold value, bigger clusters are generated.

The overlapping clustering algorithm. The algorithm is based on the technique described in [5] and it is based on the idea of extending some sets of values given as input with other data set elements. In particular, the algorithm starts from a set of *poles* $\mathcal{P} = \{P_1, \dots, P_l\}$ where P_i is a subset of the considered values set and $P_i \cap P_j = \{\} \forall i \neq j$. Then, a membership degree is calculated for each elements of the values set with respect to each pole. Finally, by means of a specific similarity measure evaluating the membership degrees, each element is assigned to one or more poles similar to it.

It is trivial to show that the results are highly dependent on the heuristic used for calculating the initial set of poles. Using the similarities available in our specific model, we implemented two techniques for calculating poles: the first one considers the results of the hierarchical clustering as poles, the second one considers the root elements as poles. The results are different: in the first case the similarity measures assume a key role; in the second case, no similarity measure is computed since the algorithm exploits only the domination.

3.4 Step 4: Name selection

A relevant value name is typically the most general value among the *values*, i.e. given a generic $rv_i = \langle rvn_i, values_i \rangle$, rvn_i is the most general value of $values_i$. The simplest way to detect a list of rvn_i candidates is to use the *Contains* function. The designer may select the most appropriate name among them.

3.5 Step 5: Validation

The results of clustering algorithms must be assessed with quality measures. We implemented a set of standard quality measures to support the designer in the tuning activity.

As stated in [7], two main cluster validation methods could be defined: external criteria, which are based on a comparison with a pre-specified cluster structure, provided by external knowledge (in our case human domain experts), and internal criteria, which are based on quantities that involve the vectors of the data set themselves. The measures we consider are the following:

- **countRV**: number of relevant values obtained for the configuration;
- **average, max_elements, variance**: the descriptive statistics over the number of elements; in particular, average expresses the average number of values belonging to a relevant value, max_elements indicates the dimension of the largest cluster and the variance shows the variance degree among the dimensions of the clusters; for values sets equally distributed on the domain max_elements is close to the average value and variance is low; the average value also gives an idea of the effectiveness of metadata representativity and "compression", all the elements included in non-outlier cluster are represented by the associated relevant value;
- **percentage of outliers**: best results are when it is close to that of reference set;
- **Rand Statistic index, Jaccard index, Folkes and Mallows index [7]**: compute the closeness of two sets of clusters evaluating couples of values that belong to the same cluster in both the sets;
- **silhouette [10]** (only if a hierarchical clustering algorithm is used): calculates for each object a silhouette value, which ranges from -1 (badly clustered) to 1 (well clustered); then, for each cluster calculates the average index; the global index in the table is the weighted average over all the clusters, excluding outliers;
- **overlapping degree** (only if an overlapping clustering algorithm is used): indicates the percentage of elements which belong to more than one relevant value.

Notice that the Rand, Jaccard, Folkes and Mallows indexes compare two different sets of clusters. We use these indexes both to compare the *RELEVANT* results w.r.t. a reference set, and to compare the differences between two different parameter settings. The reference set is provided by a domain expert or is a "gold standard".

4 Querying with Relevant Values

Thanks to the knowledge provided by relevant values, the user has two new ways of formulating queries, according to two scenarios.

1. The user has only a general idea of what she/he is searching for and composes a query predicate for instance by selecting a value x among the relevant value names. Note that instead of using the classical equality or LIKE operator, we should consider a new one, say RELATED TO, taking into account the mapping between relevant value names and values. It is beyond the scope of this paper to discuss such operator, but a naive implementation could be to substitute At RELATED TO x where x is a relevant value name, with At IN (SELECT *values* FROM METADATA.At WHERE $r_{vn}=x$)

To give a flavor of the novelty of the approach, we should observe that: (a) The user seldom has a deep knowledge of all the integrated data, so the list of the relevant value names, elicited from data, is of great help in providing insight on the

value domain, and in assisting query formulation; (b) w.r.t. the base SQL predicate $At \text{ LIKE } ' \%x\% '$ we propose a rewriting of the query which is guided by the semantics of clustering and string containment, and uses also, as base tools, the information retrieval techniques of stemming and stop words.

2. The user knows that the result must include tuples satisfying the predicate $At = v$, but she/he is aware that, due to the integration process, tuples with values v' similar to v might also be relevant. In this case the query could be transformed in a query of type 1 above by substituting $At = v$ with $At \text{ RELATED TO } rvn$, where $v \in \text{values}(rvn)$, or possibly with a disjunction of predicates like that, if overlapping clustering is used.

5 Conclusions and future work

In this paper we defined a new type of metadata, the relevant values of an attribute domain. The experimental results evaluated by means of *RELEVANT* show that the technique produce results close to the relevant values provided by a domain expert. The best results are obtained by applying the overlapping clustering algorithms.

Future work will be addressed on improving the relevant values selection by automatically calculating some indicators for evaluating the quality of the relevant values. In this way, the designer may be supported in the parameters selection. Moreover, we will study the problem of the generation of the relevant value set for multiple attributes and that of quantitative evaluation of cluster quality in the overlapping case.

References

1. D. Beneventano and S. Bergamaschi. Semantic Search Engines based on Data Integration Systems. In *Semantic Web: Theory, Tools and Applications* (Ed. Jorge Cardoso). Idea Group Publishing, 2006.
2. D. Beneventano, S. Bergamaschi, S. Bruschi, F. Guerra, M. Orsini, and M. Vincini. Instances navigation for querying integrated data from web-sites. In *In Int. Conf. on Web Information Systems and Technologies*, Setubal, Portugal, April 2006.
3. D. Beneventano, S. Bergamaschi, F. Guerra, and M. Vincini. Synthesizing an integrated ontology. *IEEE Internet Computing*, pages 42–51, Sep-Oct 2003.
4. S. Bergamaschi, S. Castano, D. Beneventano, and M. Vincini. Semantic integration of heterogeneous information sources. *Data & Knowledge Engineering*, 36(1):215–249, 2001.
5. G. Cleuziou, L. Martin, and C. Vrain. PoBOC: An overlapping clustering algorithm, application to rule-based classification and textual data. In *Proceedings of the 16th ECAI conference*, pages 440–444, 2004.
6. B. S. Everitt. *Cluster Analysis*. Edward Arnold and Halsted Press, 1993.
7. M. Halkidi, Y. Batistakis, and M. Vazirgiannis. On clustering validation techniques. *J. Intell. Inf. Syst.*, 17(2-3):107–145, 2001.
8. A. K. Jain, M. N. Murty, and P. J. Flynn. Data clustering: A review. *ACM Comput. Surv.*, 31(3):264–323, 1999.
9. M. Lenzerini. Data integration: A theoretical perspective. In Lucian Popa, editor, *PODS*, pages 233–246. ACM, 2002.
10. P.J. Rousseeuw. Silhouettes: a graphical aid to the interpretation and validation of cluster analysis. *J. Comput. Appl. Math.*, 20:53–65, 1987.
11. C. J. van Rijsbergen. *Information Retrieval*. Butterworth, 1979.

Towards Constraint-Based Subgraph Mining

Michele Berlingerio^{1,2} Francesco Bonchi¹ Fosca Giannotti¹

¹Pisa KDD Laboratory, ISTI - CNR, Via Giuseppe Moruzzi, 1 - Pisa, Italy

²IMT, School for Advanced Studies, Via San Michele, 3 - Lucca, Italy
{name.surname}@isti.cnr.it

Abstract. The traditional motivations for constraint-based pattern mining, i.e., user-controlled focus in the mining process and gain in efficiency, are even stronger when dealing with graphs. On the one hand, mining graphs faces larger computational demands than itemsets or sequences, on the other hand, there are many application domains, such as cheminformatics or proteomics, where meaningful constraints naturally arise. Thus it is important to develop a framework for constraint-based graph mining, individuating properties of interesting constraints and developing adequate computational techniques. In this paper we first introduce a large variety of constraints on graphs and we show that they are all either anti-monotone or monotone; we then provide preliminary results on subgraph mining under a conjunction of these kinds of constraints.

1 Introduction

Whether dealing with sets, sequences or graphs, the extraction of frequent substructures from a database of structures is a fundamental task in data mining: the extracted patterns can directly represent interesting and actionable knowledge, or they can be used as basic bricks to build more complex models. But such mining task always faces the same computational aspects: a usually very large input, an exponential search space, and a too large solution set. This has two main consequences: (i) mining is often inefficient and sometimes simply unfeasible, and (ii) the identification of the fragments of interesting knowledge, blurred within a huge quantity of mostly useless patterns, is difficult. The paradigm of *constraint-based pattern mining* [11, 14] was introduced as an helpful tool for both these problems: constraints drive the mining process towards potentially interesting patterns, moreover they can be pushed deep inside the mining algorithm in order to fight the exponential search space curse, and to achieve better performance. So far, the most of the work in constraint-based pattern mining has been devoted to the simplest kind of pattern, the *itemset*: properties of constraints have been studied comprehensively, and on the basis of such properties (e.g., anti-monotonicity, succinctness [11], monotonicity [7, 4], convertibility [13], loose anti-monotonicity [5]), efficient computational strategies have been defined. Also successful applications have been developed in medicine [12] and in biology [2], and pattern discovery systems based on constraints have been developed [3]. The motivations supporting the need for constraint-based pattern discovery (i.e., efficiency and focus on interesting knowledge) are even stronger when dealing with *graphs*. In fact, it is well known that subgraph isomorphism (i.e., deciding whether a graph is subgraph of another one) is a NP-complete problem: as a consequence, subgraph miners are exponential in runtime and memory consumption [10, 18, 6, 16]. We believe that the constraint-based paradigm could play

an important role in enhancing frequent subgraph pattern discovery effectiveness and usability. For instance, in *cheminformatics*, where the graphs represent chemical compounds and frequent fragments help finding new drugs [1, 9], the interesting patterns are rarely just arbitrary subgraphs which are frequent enough, but in most of the cases the interest is on typical *rigid* substructures such as *rings* and *chains*. In our vision, the analyst should be allowed to express such structural constraints, or, for instance to require that extracted patterns contain or not some specific atoms. Consider now *proteomics*, where identifying structural regularities in the tertiary structure of proteins (represented as 3D graphs) help the biologists to understand functions of new proteins [8, 15]: in this context, complex geometric or spatial constraints could be expressed as function of the 3D coordinates of vertices. Therefore we got plenty of motivations to develop a framework for constraint-based frequent structural pattern discovery, individuating properties of interesting constraints and developing adequate computational techniques.

Surprisingly, only little work has been done in this direction. The most related proposal is the *CabGin* framework [17], which extends *ADI-Mine* [16] to deal with some classes of constraints. In particular, following the work done on itemsets [11], the classes of anti-monotone, succinct and monotone constraints are individuated. But, as in [11], while anti-monotone and succinct constraints are pushed deep inside the mining process, monotone constraints are just checked and do not participate to prune the search space anyhow.

2 Problem Definition

As most of the works on frequent subgraph mining, in this paper we focus on undirected connected labeled graphs without multiple edges. A labeled graph is a 4-tuple $G = (V, E, L, l)$, where V is a set of vertices, $E \subseteq V \times V$ is a set of edges, L is a set of labels, and $l : V \cup E \rightarrow L$ is a labeling function that assigns a label to an edge or a vertex. A graph G is called *connected* if for any vertices $u, v \in V$, there exist vertices $w_1, \dots, w_n \in V$ such that $\{(u, w_1), (w_1, w_2), \dots, (w_{n-1}, w_n), (w_n, v)\} \subseteq E$. Let \mathbb{G} be the domain of all possible connected graphs, a constraint on graph patterns is a function $\mathcal{C} : \mathbb{G} \rightarrow \{true, false\}$. We denote $Th(\mathcal{C}) = \{G \in \mathbb{G} | \mathcal{C}(G) = true\}$ the set of all graphs satisfying a constraint \mathcal{C} . The *frequency constraint* is defined on the basis of subgraph isomorphism. A graph $G' = (V', E', L', l')$ is a subgraph of a graph $G = (V, E, L, l)$ (denoted as $G' \subseteq G$), if there exists a subgraph isomorphism from G' to G . A subgraph isomorphism from G' to G is an injective function $f : V' \rightarrow V$ such that: (1) for any vertex $u \in V'$, $f(u) \in V$ and $l'(u) = l(f(u))$; and (2) for any edge $(u, v) \in E'$, $(f(u), f(v)) \in E$ and $l'(u, v) = l(f(u), f(v))$.

Given a graph database \mathcal{D} , the support of a graph G' in \mathcal{D} , denoted as $sup_{\mathcal{D}}(G')$ is the number of graphs in the database that are supergraphs of G' , i.e., $sup_{\mathcal{D}}(G') = |\{G \in \mathcal{D} | G' \subseteq G\}|$. Given a minimum support threshold σ , the frequency constraint $\mathcal{C}_{freq[\mathcal{D}, \sigma]}$ is satisfied by all the graphs G such that $sup_{\mathcal{D}}(G) \geq \sigma$. According to this notation, the classical frequent subgraph mining problem requires to compute $Th(\mathcal{C}_{freq})$ (we omit \mathcal{D} and σ when reasonable). If we want our substructures to satisfy also a conjunction of other constraints \mathcal{C} , we obtain the *constraint-based frequent pattern mining problem*: $Th(\mathcal{C}_{freq}) \cap Th(\mathcal{C})$. In particular, in this paper we focus on the problem of pushing *monotone* constraints in the frequent pattern computation. A monotone constraint \mathcal{C}_M is such that, if satisfied by a graph G , then it is satisfied by any connected

supergraph of G . The frequency constraint \mathcal{C}_{freq} instead behaves the opposite: if satisfied by a graph G , then it is satisfied by any connected subgraph of G . Constraints that behave as \mathcal{C}_{freq} are said *anti-monotone* and denoted \mathcal{C}_{AM} . For sake of simplicity in this paper we focus on the problem $Th(\mathcal{C}_{freq}) \cap Th(\mathcal{C}_M)$, but since a conjunction of \mathcal{C}_{AM} constraints is still a \mathcal{C}_{AM} constraint, the proposed solution works more generally for any conjunction of anti-monotone and monotone constraints: i.e., $Th(\mathcal{C}_{AM}) \cap Th(\mathcal{C}_M)$.

In this paper we address the problem of subgraph mining when a conjunction of monotone constraints is conjoined to the frequency anti-monotone constraint, and we introduce $\mathcal{G}amp$ (*Graphs anti-monotone and monotone pruning*), a pre-processing algorithm, which reduces dramatically the input database, hence inducing a strong pruning of the search space. Being a pre-processor, $\mathcal{G}amp$ can be coupled with any subgraph miner proposed so far. To the best of our knowledge, is the first work showing how to effectively push monotone constraints in frequent subgraph mining, using them to reduce input data and to prune the search space.

3 Constraints on Subgraph Patterns

In this section we introduce the kinds of constraints that we study in the rest of the paper, and we discuss their properties (anti-monotonicity or monotonicity). Since the proofs of these properties are always straightforward, we avoid providing them. The variety of constraints presented is not meant to be exhaustive, since meaningful constraints will be suggested by the application at hand: the objective here is to show that most of the fundamental properties of graphs can be modelled as either monotone or anti-monotone constraints. In the following we distinguish between *topological constraints*, i.e., those constraints regarding only the topological structure of a graph pattern; and *labeled graph constraints*, i.e., those constraints that involve not only the structure of a graph pattern but also its labels.

Given a graph $G = (V, E, L, l)$, a topological constraint is such that, if satisfied or not by G can be decided on the basis of just the V and E components of G . The first simple example of topological constraint is the constraint on the size, in terms of number of vertices or edges, that a substructure should have in order to qualify as interesting pattern.

Definition 1 (Size Constraint). *The size constraint is denoted $\mathcal{C}_{size[S,\theta,\alpha]}$, where $S \in \{V, E\}$, $\theta \in \{\geq, \leq\}$, and $\alpha \in \mathbb{N}$. It is satisfied by a graph G iff $|S| \theta \alpha$. It is trivial to see that the size constraint $\mathcal{C}_{size[S,\leq,\alpha]}$ is anti-monotone, while $\mathcal{C}_{size[S,\geq,\alpha]}$ is monotone.*

Often in molecular biology we are not interested in generic substructures, but in rigid structures such as chains and rings. These can be expressed as topological constraints.

Definition 2 (Cycle Constraint). *Given $\alpha \in \mathbb{N}$, the cycle constraint $\mathcal{C}_{cycle[\alpha]}$ is satisfied by $G = (V, E, L, l)$ iff G contains a simple cycle of size $\geq \alpha$, i.e., there exist $n \geq \alpha$ distinct vertices $v_1, \dots, v_n \in V$ such that $\{(v_1, v_2), \dots, (v_{n-1}, v_n), (v_n, v_1)\} \subseteq E$. The cycle constraint $\mathcal{C}_{cycle[\alpha]}$ is monotone.*

One could be interested in mining frequent *acyclic* substructures. This topological constraint can be obtained by negating a cycle constraint, i.e., we can define $\mathcal{C}_{acyc} \equiv \neg \mathcal{C}_{cycle[2]}$. Since the negated of a monotone constraint is anti-monotone (and viceversa), \mathcal{C}_{acyc} is anti-monotone. Since we consider only connected substructures, also

the *chain constraint* \mathcal{C}_{chain} , which is satisfied only by graphs that are chains, is anti-monotone: any connected substructure of a chain is still a chain, or the other way around, if a graph is not a chain, all its supergraphs are not chains as well. Many other constraints, which are either monotone or anti-monotone, can be defined on the topological structure of the graph patterns we are mining: the point is to individuate the meaningful constraints for the given application. Given a graph $G = (V, E, L, l)$, a labeled graph constraint is such that, if satisfied or not by G can not be decided simply on the basis of its topology (the V and E components), but also the labels are needed. The first simple example of constraint on labels specifies which particular elements (both vertices or edges labels) must be contained or not in the patterns.

Definition 3 (Labels Constraint). *Let L' be a set of labels, and $G = (V, E, L, l)$ a graph. We define the following 4 constraints on vertices labels:*

- $\mathcal{C}_{vrtx[L', \sqsubseteq]}(G) \equiv \forall l \in L', \exists v \in V : l(v) = l.$
- $\mathcal{C}_{vrtx[L', \supseteq]}(G) \equiv \forall v \in V, l(v) \in L'.$
- $\mathcal{C}_{vrtx[L', \cap]}(G) \equiv \exists l \in L', \exists v \in V : l(v) = l.$
- $\mathcal{C}_{vrtx[L', \cap]}(G) \equiv \forall l \in L', \nexists v \in V : l(v) = l.$

Similarly, we define the same 4 constraints for edges labels (using the notation \mathcal{C}_{edge}).

It is straightforward to see that: $\mathcal{C}_{vrtx[L', \sqsubseteq]}(G)$ is monotone, $\mathcal{C}_{vrtx[L', \supseteq]}(G)$ is anti-monotone, $\mathcal{C}_{vrtx[L', \cap]}(G)$ is monotone, and $\mathcal{C}_{vrtx[L', \cap]}(G)$ is anti-monotone. The same properties hold for constraints on edges labels. The first two constraints in the conjunction are anti-monotone, while the second two are monotone: this is the kind of query that we address in this paper.

We can define also constraints based on aggregates of the labels (either of vertices or edges). Such aggregates can be computed directly over the labels, or on some attributes of the labels. In the second case, such attributes must be in functional dependency with the labels. For instance, suppose that labels of vertices are atom types, we can define a constraint on the *sum of atomic weights*, where *atomic weight* is an attribute of the label atom type, in functional dependency (i.e., the same atom type must have a unique atomic weight all over the database).

Example 1. Suppose that from a molecules database \mathcal{D} we want to mine frequent ($\sigma = 30$) substructures, that contain at least 3 atoms of oxygen, and have sum of atomic weights larger than 120. This query can be expressed as:

$$Th(\mathcal{C}_{freq[\mathcal{D}, 30]} \wedge \mathcal{C}_{count[V=O, \geq, 3]} \wedge \mathcal{C}_{sum[V.weight, \geq, 120]})$$

4 Data Reduction Properties

Here we show how the ExAnte property[4] can be used when dealing with graphs. Such a property states that a transaction (i.e., a graph in the database, in our case) which does not satisfy the given monotone constraint can be deleted from the input database since it will never contribute to the support of any solution substructure.

Proposition 1 (Monotone Pruning). *Given a conjunction of monotone constraints \mathcal{C}_M , we define the monotone pruning of a database of graphs \mathcal{D} as the dataset resulting from removing the graphs that do not satisfy \mathcal{C}_M : $Mp_{[\mathcal{C}_M]}(\mathcal{D}) = \{G \in \mathcal{D} \mid$*

$G \in Th(\mathcal{C}_M)\}$. It holds that this data reduction does not affect the support of solution patterns: $\forall G' \in Th(\mathcal{C}_{AM}) \cap Th(\mathcal{C}_M) : sup_{\mathcal{D}}(G') = sup_{\mathcal{M}p[\mathcal{C}_M]}(\mathcal{D})(G')$.

A major consequence of reducing the input database in this way is that it implicitly reduces the support of a large amount of edges that do not satisfy \mathcal{C}_M as well, resulting in a reduced number of patterns visited during the mining. This is not the whole story, in fact, infrequent edges can not only be removed from the search space together with all their supergraphs, for the same anti-monotonicity property of \mathcal{C}_{freq} , they can be deleted also from all graphs in \mathcal{D} .

Proposition 2 (Anti-monotone Pruning). Given a conjunction of \mathcal{C}_{AM} constraints, we define the anti-monotone pruning of a graph $G = (V, E, L, l)$ as the graph obtained removing the edges that do not satisfy \mathcal{C}_{AM} from G : we denote this pruning $\mathcal{AM}p[\mathcal{C}_{AM}](G)$. Given a database of graphs \mathcal{D} we define the anti-monotone pruning of \mathcal{D} as the anti-monotone pruning of all graphs in \mathcal{D} : $\mathcal{AM}p[\mathcal{C}_{AM}](\mathcal{D}) = \{G' \mid G \in \mathcal{D} \wedge G' = \mathcal{AM}p[\mathcal{C}_{AM}](G)\}$. It holds that this data reduction does not affect the support of solution patterns: $\forall G' \in Th(\mathcal{C}_{AM}) \cap Th(\mathcal{C}_M) : sup_{\mathcal{D}}(G') = sup_{\mathcal{AM}p[\mathcal{C}_{AM}](\mathcal{D})}(G')$.

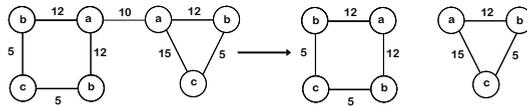


Fig. 1. Example of $\mathcal{AM}p$ pruning.

Removing edges from graphs has got another positive effect: after the anti-monotone pruning, a growing number of graphs that do not satisfy \mathcal{C}_M can be found in the input database. Therefore the anti-monotone pruning creates new opportunities for monotone pruning, and viceversa: we are inside a loop where two different kinds of pruning reduce the input data, strengthening each other step by step! Moreover, removing edges from the connected graphs in the input database, can disconnect these graphs. Since we are interested in mining frequent *connected* substructures, we can strengthen the $\mathcal{M}p$ pruning to remove any *connected component*, i.e., *maximal connected subgraph*, in the input database which does not satisfy \mathcal{C}_M .

Example 2. Let $G \in \mathcal{D}$ be the graph in Figure 1. Consider the query: $Th(\mathcal{C}_{freq[\mathcal{D},30]} \wedge \mathcal{C}_{sum[E,\geq,35]})$. We got that $\mathcal{C}_{sum[E,\geq,35]}(G) = true$, since the sum of edge labels of G is 76. Suppose now that the edge $(a, 10, a)$ results to be infrequent. $\mathcal{AM}p$ removes it from G , which results to be divided in two connected components. When we go to check satisfaction of \mathcal{C}_M we must not consider the sum of edge labels of the whole G , but that of all its connected components in isolation. In this case none of the two connected components satisfies \mathcal{C}_M and thus they will be both deleted by the subsequent $\mathcal{M}p$.

Proposition 3 (Connected Components Pruning). Let $G \in \mathcal{D}$ be a graph not necessarily connected. Given a conjunction of monotone constraints \mathcal{C}_M , we define the monotone pruning $\mathcal{M}p[\mathcal{C}_M](G)$ as the graph resulting by removing from G all its connected components that do not satisfy \mathcal{C}_M . We define the monotone pruning of the whole database \mathcal{D} as the monotone pruning of all graphs in \mathcal{D} : $\mathcal{M}p[\mathcal{C}_M](\mathcal{D}) = \{G' \mid G \in$

$\mathcal{D} \wedge G' = \mathcal{M}p_{[\mathcal{C}_M]}(G)$. This data reduction does not affect the support of solution patterns: $\forall G' \in Th(\mathcal{C}_{AM}) \cap Th(\mathcal{C}_M) : sup_{\mathcal{D}}(G') = sup_{\mathcal{M}p_{[\mathcal{C}_M]}(\mathcal{D})}(G')$.

In this section we have described a loop where two different kinds of pruning ($\mathcal{A}M_p$ and $\mathcal{M}p$) cooperate to reduce the search space and the input dataset, strengthening each other step by step until no more pruning is possible (a fix-point has been reached). In the end, the reduced dataset resulting from this fix-point computation is usually much smaller than the initial dataset, and it can feed any frequent substructure miner for a much smaller (but complete) computation. This is the $\mathcal{G}amp$ pre-processing method, which will be further clarified by the following example.

4.1 Run-through Example

Let \mathcal{D} in Figure 2(a) be our input database of graph and suppose that we want to mine patterns having support ≥ 2 and sum of edge labels ≥ 35 , i.e., $Th(\mathcal{C}_{freq[\mathcal{D},2]} \wedge \mathcal{C}_{sum[E,\geq,35]})$. If we start counting the support of edges, we discover that they are all frequent. Therefore it could seem that no pruning is possible at this time. But if we take in consideration the monotone constraint $\mathcal{C}_{sum[E,\geq,35]}$, we discover that graph G_2 does not satisfy it. Therefore G_2 can be removed from \mathcal{D} (pruning $\mathcal{M}p$).

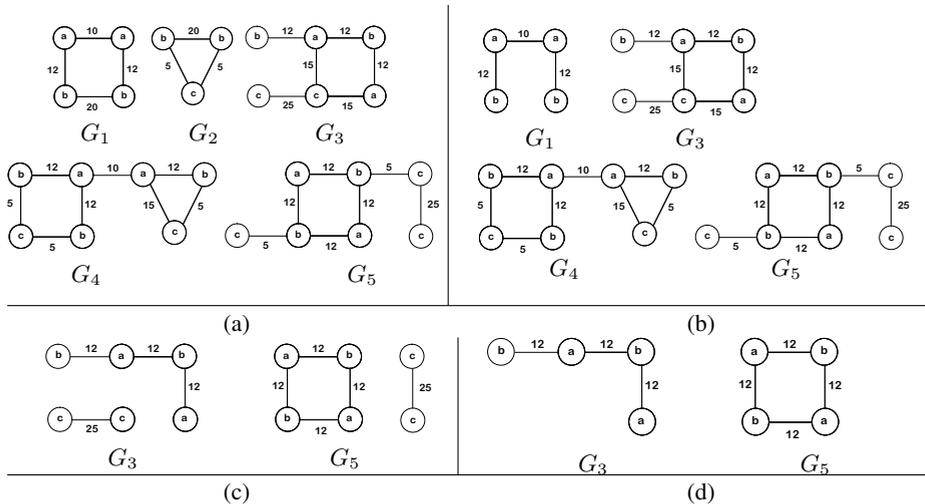


Fig. 2. The input database (a) After the first $\mathcal{M}p$ and $\mathcal{A}M_p$ (b) After the third $\mathcal{M}p$ and $\mathcal{A}M_p$ (c) The final reduced database (d)

This removal makes the support of the edge $(b, 20, b)$ shrink to 1, and thus such edge which was initially frequent, is now infrequent and can be removed in all its occurrences from \mathcal{D} (pruning $\mathcal{A}M_p$). In particular, it is removed from graph G_1 . The database after the first round of $\mathcal{M}p$ and $\mathcal{A}M_p$ pruning is as in Figure 2(b). After the removal of $(b, 20, b)$, the graph G_1 has a sum of edges which does no longer satisfy \mathcal{C}_M , and thus it will be $\mathcal{M}p$ pruned during the second iteration. After the removal of G_1 , the edge $(a, 10, a)$ turns out to be infrequent, and thus it is $\mathcal{A}M_p$ pruned. This pruning, which corresponds to the one in Figure 1, has the effect of dividing G_4 in two connected components, that do not satisfy \mathcal{C}_M . Therefore, both components can be $\mathcal{M}p$ pruned. At this point both $(a, 15, c)$ (supported only by G_3) and $(b, 5, c)$ (supported only by G_5)

turn out to be infrequent and they are $\mathcal{AM}p$ pruned. After this pruning the input database has been reduced as in Figure 2(c). Both G_3 and G_5 are divided in two connected components. During the next $\mathcal{M}p$ the components not satisfying \mathcal{C}_M are deleted. In this case, in both graphs, there is one connected component made only by the edge $(c, 25, c)$: since it does not satisfy the sum constraint it is removed from both graphs. Note that, even if the edge $(c, 25, c)$ is frequent it is removed by $\mathcal{M}p$ and not by $\mathcal{AM}p$. At this point the input database is as in Figure 2(d). The unique edge which is present in both graphs $(a, 12, b)$ is frequent, and thus no other pruning is possible. This is the final database, the result of our $\mathcal{G}amp$ pre-processing, which can feed any frequent substructure miner for a much smaller (but complete) computation. Note that on this toy-example we even do not need to run a mining algorithm: the unique pattern solution to our query (i.e., a graph with 3 $(a, 12, b)$ edges appropriately connected, with support = 2 and sum of edge labels = 36) is evident in Figure 2(d).

5 The $\mathcal{G}amp$ Algorithm

This section describes the $\mathcal{G}amp$ pre-processing idea. In $\mathcal{G}amp$ we use an extension of the \mathcal{ADI} structure [16], called \mathcal{EADI} . This structure allows us to efficiently perform operations on graphs and edges, and permits components management. Details and experimental results are omitted due to lack of space. Given the mining problem $Th(\mathcal{C}_{freq[\mathcal{D}, \sigma]}) \cap Th(\mathcal{C}_M)$, the $\mathcal{G}amp$ pre-processing (Algorithm 1) outputs a reduced database \mathcal{D}' obtained applying the $\mathcal{AM}p$ and $\mathcal{M}p$ pruning procedures directly on the \mathcal{EADI} structure, one after the other, until the number of frequent edges stops shrinking. At the end, the result is a reduced database.

Algorithm 1 $\mathcal{G}amp$

Input: $\mathcal{D}, \sigma, \mathcal{C}_M$

Output: A reduced database \mathcal{D}' .

```

1:  $\mathcal{D}'' \leftarrow \mathcal{D}, \mathcal{D}' \leftarrow NULL;$ 
2: while  $(\mathcal{D}' \neq \mathcal{D}'')$  do
3:    $\mathcal{D}' \leftarrow \mathcal{D}'';$ 
4:    $\mathcal{D}'' \leftarrow \mathcal{AM}p_{[\sigma]}(\mathcal{D}')$ ;
5:    $\mathcal{D}'' \leftarrow \mathcal{M}p_{[\mathcal{C}_M]}(\mathcal{D}'')$ ;

```

6 Conclusions and Future Work

In this paper we have addressed the problem of mining frequent connected substructures from a database of graphs, under a conjunction of user-defined constraints. We have provided a wide range of meaningful graph constraints and we have studied their monotonicity properties to reduce the data and the search space. We have then showed how this properties can be made cooperate with the underlying constraint of mining only connected subgraphs, obtaining a stronger pruning. We have described $\mathcal{G}amp$, a pre-processing algorithm which reduces dramatically the input database and hence the subsequent mining computation. We are developing a constraint-based graph miner that exploits the ideas presented in this paper during the whole mining of frequent substructures instead of in the pre-processing only. We plan to investigate other constraints properties (e.g., loose anti-monotonicity [5]) and how they can be exploited in the context of graph mining. Our short term objective is to produce a robust and efficient constraint-based graph miner able to handle a large variety of constraints, to amalgamate it in our CONQUEST [3] system, and to study its usability and possible improvements in tight collaboration with molecular biologists.

References

1. D. M. Bayada, R. W. Simpson, and A. P. Johnson. An algorithm for multiple common subgraphs problem. *Journal of Chemical Information and Computer Science*, 32:680–685, 1992.
2. J. Besson, C. Robardet, J. F. Boulicaut, and S. Rome. Constraint-based concept mining and its application to microarray data analysis. *Intelligent Data Analysis Journal*, 9(1):59–82, 2005.
3. F. Bonchi, F. Giannotti, C. Lucchese, S. Orlando, R. Perego, and R. Trasarti. ConQueSt: a constraint-based querying system for exploratory pattern discovery. In *Proc. of the 22nd IEEE Int. Conf. on Data Engineering (ICDE'06)*.
4. F. Bonchi, F. Giannotti, A. Mazzanti, and D. Pedreschi. ExAnte: Anticipated data reduction in constrained pattern mining. In *Proc. of the 7th European Conf. on Principles and Practice of Knowledge Discovery in Databases (PKDD'03)*.
5. F. Bonchi and C. Lucchese. Pushing tougher constraints in frequent pattern mining. In *Proc. of the 9th Pacific-Asia Conf. on Knowledge Discovery and Data Mining (PAKDD'05)*.
6. C. Borgelt and M. R. Berthold. Mining molecular fragments: Finding relevant substructures of molecules. In *Proc. of the 2nd IEEE Int. Conf. on Data Mining (ICDM'02)*.
7. C. Bucila, J. Gehrke, D. Kifer, and W. White. DualMiner: A dual-pruning algorithm for itemsets with constraints. In *Proc. of the 8th ACM Int. Conf. on Knowledge Discovery and Data Mining (SIGKDD'02)*.
8. L. P. Chew, D. Huttenlocher, K. Kedem, and J. Kleinberg. Fast detection of common geometric substructure in proteins. In *Proc. of the 3rd annual Int. Conf. on Computational Molecular Biology (RECOMB'99)*.
9. L. Dehaspe, H. Toivonen, and R. D. King. Finding frequent substructures in chemical compounds. In *Proceeding of the 4th Int. Conf. on Knowledge Discovery and Data Mining (SIGKDD'98)*.
10. A. Inokuchi, T. Washio, and H. Motoda. An apriori-based algorithm for mining frequent substructures from graph data. In *Proc. of the 4th European Conf. on Principles and Practice of Knowledge Discovery in Databases (PKDD'00)*.
11. R. T. Ng, L. V. S. Lakshmanan, J. Han, and A. Pang. Exploratory mining and pruning optimizations of constrained associations rules. In *Proc. of the ACM Int. Conf. on Management of Data (SIGMOD'98)*.
12. C. Ordonez, E. Omiecinski, L. de Braal, C. A. Santana, N. Ezquerro, J. A. Taboada, D. Cooke, E. Krawczynska, and E. V. Garcia. Mining constrained association rules to predict heart disease. In *Proc. of the 1st IEEE Int. Conf. on Data Mining (ICDM'01)*.
13. J. Pei, J. Han, and L. V. S. Lakshmanan. Mining frequent item sets with convertible constraints. In *17th IEEE Int. Conf. on Data Engineering (ICDE'01)*.
14. R. Srikant, Q. Vu, and R. Agrawal. Mining association rules with item constraints. In *Proc. 3rd ACM Int. Conf. on Knowledge Discovery and Data Mining, (SIGKDD'97)*, 1997.
15. S. Su, D. J. Cook, and L. B. Holder. Knowledge discovery in molecular biology: identifying structural regularities in proteins. *Intelligent Data Analysis*, 3:413–436, 1999.
16. C. Wang, W. Wang, J. Pei, Y. Zhu, and B. Shi. Scalable mining of large disk-based graph databases. In *Proc. of the 10th ACM Int. Conf. on Knowledge Discovery and Data Mining (SIGKDD'04)*.
17. C. Wang, Y. Zhu, T. Wu, W. Wang, and B. Shi. Constraint-based graph mining in large database. In *Proc. of the 7th Asia-Pacific Web Conf., (APWeb'05)*.
18. X. Yan and J. Han. gspan: Graph-based substructure pattern mining. In *Proc. of the 2nd IEEE Int. Conf. on Data Mining (ICDM'02)*.

Generating Extended Conceptual Schemas from Business Process Models

Marco Brambilla¹, Jordi Cabot², Sara Comai¹

¹ Dipartimento di Elettronica e Informazione, Politecnico di Milano
Piazza L. Da Vinci, 32. I20133 Milano, Italy
{mbrambill, comai}@elet.polimi.it

² Estudis d'Informàtica i Multimèdia, Universitat Oberta de Catalunya
Av. Tibidabo 39 E08035 Barcelona, Spain
jcabot@uoc.edu

Abstract. The specification of business processes is becoming a more and more critical aspect for organizations. Such processes are specified as workflow models expressing the logical precedence among the different business activities (i.e., the units of work). Up to now, workflow models have been commonly managed through specific subsystems, called workflow management systems. In this paper we advocate for an integration of the workflow specification in the domain conceptual schema. This workflow-extended schema is automatically derived from the workflow model and comprises some new entity and relationship types for describing the workflow elements and a set of integrity constraints to ensure the workflow precedence rules.

1. Introduction

Specification of complex business applications usually requires the definition of a workflow model to express logical precedences and constraints among the different business activities (i.e. the units of work). Workflow models are usually implemented with the help of dedicated workflow management systems (e.g., [5], [10]) which are heavy-weight applications focused on the control aspects of the workflow enactment. In this paper we adopt a different approach and advocate for the integration of the workflow model with the domain *conceptual schema (CS)*. Starting from the workflow model it is possible to define a full-fledged conceptual schema enriched with the entity and relationship types needed to record the required workflow information and with a set of process constraints over such types to control the correct workflow execution. We refer to this resulting conceptual schema as the *workflow-extended CS*. We will represent it using UML class diagrams [8] and will use OCL [7] to specify the process constraints.

The main characteristic of a workflow-extended CS is that it automatically ensures a consistent behavior of all enterprise applications with respect to the business process specification. As long as the applications properly update the workflow information in the CS, the process constraints defined in the schema enforce that the different tasks are done according to the initial workflow model. Another advantage

of a workflow-extended CS is its technological-independence. Indeed, any method and tool designed for managing a generic CS can benefit from a workflow-extended CS, no matter the target technology platform or the purpose of the tool, spawning from direct application execution, to automatic generation of the implementation, to property analysis, and to metrics measurement. Moreover, workflow-extended CS enables the definition of more expressive constraints, including timing conditions or involving both workflow and domain information. These constraints are generally not allowed in workflow definition languages since they require using a general-purpose (textual) sublanguage [4], and on the other hand may be quite complex and tedious to specify using a general-purpose CS alone.

To our knowledge, this is the first proposal where both workflow information and process constraints are automatically derived from a workflow model and integrated within the conceptual schema. In literature, workflow metadata and OCL constraints have only been used in [3] to manually specify workflow access control constraints and derive authorization rules, in [1] to express constraints with respect to the distribution of work to teams, and in ArgoUML [6] to check for well-formedness in the design of process models.

2. Basic workflow concepts

Several visual notations, languages, and methodologies to specify workflow models have been proposed, with different expressive power, syntax and semantics. In our work we have adopted the Workflow Management Coalition terminology [11] and the BPMN [9] notation.

The workflow model is hence based on the concepts of *Process* (the description of the business process), *Case* (a process instance), *Activity* (the elementary unit of work composing a process), *Activity instance* (an instantiation of an activity within a case), *Actor* (a user role intervening in the process), *Event* (some punctual situation that happens in a case), and *Constraint* (logical precedence among activities and rules enabling activities execution). Processes can be internally structured using a variety of constructs: sequences of activities; gateways implementing AND, OR, XOR splits, respectively realizing splits into independent, alternative and exclusive threads; gateways implementing joins, i.e., convergence point of two or more activity flows; conditional flows between two activities; loops among activities or repetitions of single activities. Each construct may involve several constraints over the activities.

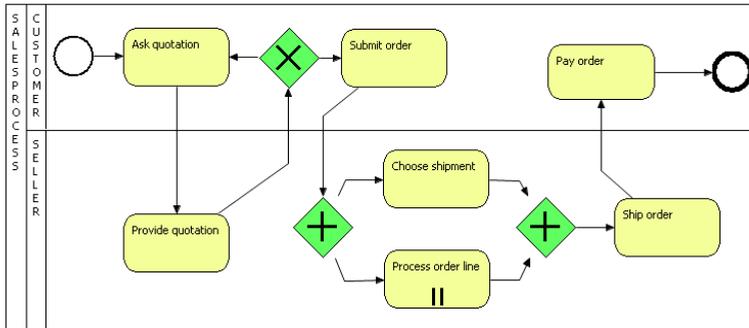


Fig. 2.1 - Example of a workflow schema

In the sequel, we will exemplify the proposed approach on a case study consisting of a workflow implementing a simplified purchase process, as illustrated in Fig. 2.1. The process involves two actors: a customer and a seller. The customer starts the workflow by asking for a quotation about a set of products (*Ask Quotation* activity). The seller provides the quotation and the customer may decide (exclusive choice) to modify his request or to accept it; in the former case, the quotation request and response cycle is repeated. In the latter case the customer submits the order and the seller takes care of it. The order management requires two parallel activities to be performed: the choice of the shipment options and the internal management of each order line (multi-instance activity *Process OrderLine*). Once all order lines have been processed and the shipment has been decided (i.e., after the synchronization of the two branches by means of the AND merge), the order is shipped and the customer pays the corresponding amount.

3. Extending conceptual schemas with workflow information

Assuming that the application is initially specified by a conceptual schema describing the domain information, the conceptual schema of the workflow-based application can be obtained by extending the domain schema with some additional elements. In our case study, the domain schema (see the bottom part of Fig. 3.1) includes entity types *Product*, *Quotation*, *QuotationLine*, and *Order*. A *Quotation* is defined as a set of *QuotationLines*, each of them referring to a *Product*. When accepted by the customer, a *Quotation* generates an *Order*. Then, the quotation lines of the quotation associated to the submitted order are referred to as order lines.

Given an initial domain model, the workflow-extended domain model is obtained by adding new elements derived from the workflow specification. Clearly, the workflow-extended domain model is more complex than the original domain model. However, we believe that this increased complexity is compensated by the fact that it may be automatically generated and processed (with, for instance, code-generation tools). Moreover, the size of the extension is constant regardless the size of the domain model and linear with respect to the number of activities in the workflow.

The extensions to the schema include: (i) *user*-related information, (ii) *workflow*-related information, (iii) a set of possible relationships between the domain conceptual schema, the workflow information and the user information, and (iv) a set of process constraints guaranteeing a consistent state of the whole conceptual schema with respect to the workflow definition. This extended model can be (semi-) automatically derived from the workflow model.

We define a workflow-extended conceptual schema as follows. Given an initial conceptual schema CS with entity types $E=\{e_1, \dots, e_n\}$, representing the knowledge about the domain, and a workflow model w with activities $A=\{a_1, \dots, a_m\}$, the workflow-extended CS is obtained in the following way:

- i) *User subschema*: User-related information is added to the CS by means of two entity types (see the top-left part of Fig. 3.1):
 - Entity type *User* represents workflow actors.
 - Entity type *Group* represents groups of users, having access to the same set of tasks. A user may belong to different groups, and this is specified by a relationship type between *User* and *Group*.
- ii) *Workflow subschema*: Workflow-related information (see the top-right part of Fig. 3.1) includes the following entity types and their relationship types:
 - Entity type *Process* represents the supported workflow.
 - Entity type *Case* denotes an instance of a process; its status can be: ready, active, cancelled, aborted, or completed.
 - Entity type *ActivityType* represents the classes of activities that compose a process. Activity types are assigned to groups of users, which are responsible of managing them.
 - Entity type *ActivityInstance* denotes the occurrence of an activity within a case. Its status can be: ready, active, cancelled, aborted, or completed. Only one user can execute a particular activity instance (this is represented by relationship *Performs*). The *Precedes* relationship between activities keeps track of their execution order.
 - Entity type *EventType* represents the events that may affect the sequence or timing of activities of a process (e.g., temporal events, messages etc.). In BPMN it is possible to define three different kinds of events (*eventKind* attribute): start, intermediate, and end event. For start and intermediate events a triggering mechanism may be defined (*eventTrigger*). For end events, we may define how they affect the case execution (*eventResult*).
 - Entity type *EventInstance* denotes the occurrence of an event.
 - For each activity $a \in A$, a new subtype s_a is added to the entity type *ActivityInstance* (*ActivityType* is a *powertype* for this set of generalization relationships). The name of the subtype is the name of a (e.g., in Fig. 3.1 we introduced *ProcessOrderLine*, *AskQuotation*, *ShipOrder*, and so on).
- iii) *Relationship types between workflow subschema and domain subschema*: each subtype s_a is related with a (possibly empty) set of entity types $E_a \subseteq E$. These new relationship types are required to evaluate conditions appearing in the process constraints or to record the entities modified during the execution of a certain activity. In the case study (see relationships between the workflow and domain information in Fig. 3.1), a set of relationship types are established: *Quotations* are

associated to the activities *Ask Quotation* and *Provide Quotation*; *QuotationLines* are associated to the *ProcessOrderLine* activity, and so on. This means that the domain information is managed by the respectively connected activities.

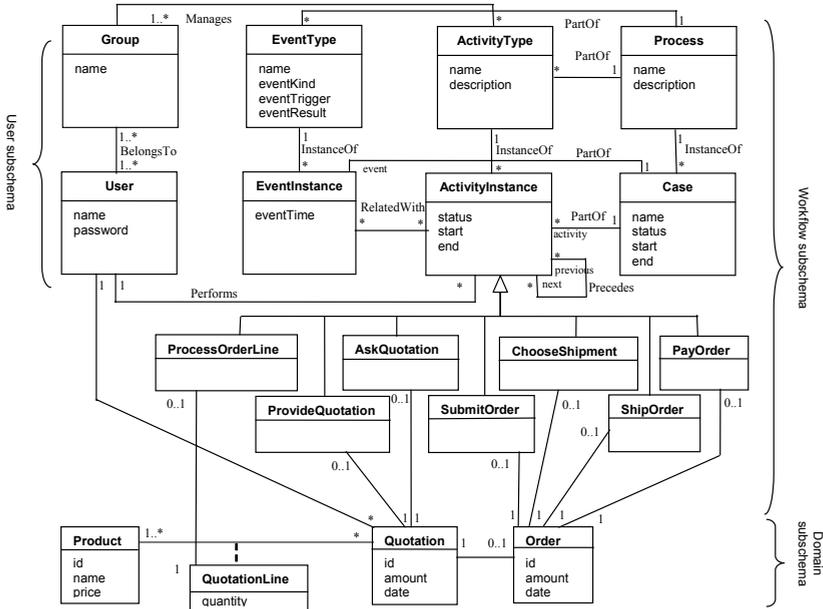


Fig. 3.1 - Example of a workflow-extended conceptual schema

The associations between the domain and the workflow subschemata must be manually specified by the designer, since the workflow model does not contain enough information to automatically relate both subschemata. All the other components of the workflow-extended CS can be instead automatically generated.

This workflow-extended CS contains all types required to record the process information. It can be possibly enriched with further relationship types and/or attributes, according to the requirements of the specific workflow application, or alternative solutions can be proposed. Next section complements the CS with a set of constraints, deduced from the workflow model, to ensure a consistent state of the CS regarding the workflow definition.

4. Translation patterns for process constraints

The structure of the workflow model implies a set of constraints regarding the execution order of the different activities, the number of possible instances of each activity in a given case, the conditions that must be satisfied in order to start a new activity, and so forth. These constraints are usually referred to as *process constraints* and are independent of the workflow notation chosen in this paper.

Process constraints are translated as OCL constraints over the population of the s_{a1}, \dots, s_{am} activity instance entity subtypes. The translation of process constraints guarantees that any update event over the population of one of these subtypes (for instance, the creation of a new activity instance or the modification of its status) will be consistent with the process constraints defined in the workflow model.

Invariants in OCL are defined in the context of a specific type, the *context type*. The actual OCL expression stating the constraint condition is called the *body* of the constraint. The *body* is always a boolean expression (i.e., it evaluates to a boolean value) and must be satisfied by all instances of the context type. The body expression may refer to attributes and relationships of the entity type. For instance, a constraint like: *context A inv: condition*, implies that all instances of *A* must verify *condition*.

Next paragraphs introduce a subset of the patterns for the generation of the process constraints corresponding to the main constructs appearing in the workflow running case. The patterns can be combined to produce the full translation of a workflow model. The full list and all the details of the workflow patterns can be found in [2].

Sequences of activities. A sequence flow between two activities A and B indicates that the activity A must be completed before starting activity B. If the first activity is completed within a given case, the second one must be eventually started* before ending the case. This behavior can be enforced in the workflow-extended CS by means of the following set of constraints:

Sequence	Process constraints
	<ul style="list-style-type: none"> - For all activity instances of type <i>B</i> the preceding activity instance must be of type <i>A</i> and must have been already completed <i>context B inv seq₁: previous->size()=1 and previous->exists(a a.oclsTypeOf(A) and a.status='completed')</i> - The creation of two different <i>B</i> instances related with the same <i>A</i> activity instance must be prevented: <i>context B inv seq₂: B.allInstances()->isUnique(previous)</i> - When the case is completed there exists a <i>B</i> activity instance for each completed <i>A</i> activity instance. This <i>B</i> instance must be the only instance immediately following the <i>A</i> activity instance. <i>context Case inv seq₃: status='completed' implies self.activity->select(a a.oclsTypeOf(A) and a.status='completed')->forAll(a a.next->exists(b b.oclsTypeOf(B)) and a.next->size()=1)</i>

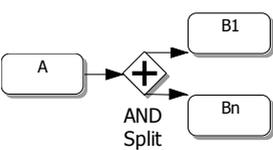
In the running case this pattern applies, for example, to the activities A=AskQuotation and B=ProvideQuotation.

Split gateways. A split gateway is a location within a workflow where the sequence flow can take two or more alternative paths. The different split gateways differ on the number of possible paths that can be taken during the execution of the workflow. For

* We do not require the second activity to be completed, since, for instance, it could be interrupted by the trigger of an intermediate exception event.

XOR-split gateways only a single path can be selected. In *OR-splits* several of the outgoing flows may be chosen. For *AND-splits* all outgoing flows must be followed.

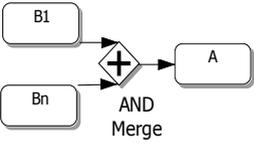
The following table shows the process constraints to enforce the AND split gateway. Beside the process constraints appearing in the table, we must also add to all the activities $B_1...B_n$ the previous constraints seq_1 and seq_2 to verify that the preceding activity A has been completed and that no two activity instances of the same activity B_i are related with the same preceding activity A . We also require that the activity instance/s following A is of type B_1 or ... or B_n .

Split gateway	Process constraints
	<ul style="list-style-type: none"> - If A is completed, all $B_1..B_n$ activities must be eventually started <p><i>context Case inv: status='completed' implies activities->select(a a.oclsTypeOf(A) and a.status='completed')->forAll(a a.next->exists(b b.oclsTypeOf(B₁)) and ... and a.next->exists(b b.oclsTypeOf(B_n)))</i></p>

In the running case, the AND split is applied to activities $A=SubmitOrder$, $B_1=ChooseShipment$, $B_2=ProcessOrderLine$.

Merge gateways. Merge gateways are useful to join or synchronize alternative sequence flows. Depending on the kind of merge gateway, the outgoing activity may start every time a single incoming flow is completed (*XOR-Merge*) or must wait until all incoming flows have finished in order to synchronize them (*AND-Merge* gateways).

The following table presents the translation pattern for the AND merge. Beside the constraints included in the table, a constraint over A should be added to verify that no two A instances are created for the same incoming set of activities.

Merge gateway	Process constraints
	<ul style="list-style-type: none"> - An activity instance of type A must wait for a set of activities $B_1..B_n$ to be completed <p><i>context A inv: previous->exists(b b.oclsTypeOf(B₁) and b.status='completed') and ... and previous->exists(b b.oclsTypeOf(B_n) and b.status='completed')</i></p> <ul style="list-style-type: none"> - Each set of completed $B_1..B_n$ activity instances must be related with an A activity instance. <p><i>context Case inv: status='completed' implies not (activity->exists(b b.oclsTypeOf(B₁) and b.status='completed' and not b.next->exists(a a.oclsTypeOf(A)) and ... and activity->exists(b b.oclsTypeOf(B_n) and b.status='completed' and not b.next->exists(a a.oclsTypeOf(A)))</i></p>

In the running case this pattern is applied to the activities $A=ShipOrder$, $B_1=ChooseShipment$, $B_2=ProcessOrderLine$.

Condition constraints. The sequence flow and the OR-split and XOR-split gateways may contain condition expressions to control the flow execution at run-time. As an

example, Fig. 4.2 shows a conditional sequence flow. In the example, the activity *B* cannot start until *A* is completed and the condition *cond* is satisfied.

A new constraint over the destination activity must be added to ensure that the preceding activity satisfies the specified condition:

context B inv: previous->forAll(a| a.cond)



Fig. 4.2 – A conditional sequence flow

5. Conclusions

In this paper we have presented an automatic approach to integrate the semantics of business process specifications within conceptual schemas. The main contribution of this work is the demonstration of the applicability of the conceptual modelling approach to the specification of workflow-based applications. We described how to build a workflow-extended conceptual schema by means of extending the domain conceptual schema with the definition of a set of new entity and relationship types for workflow status tracking; and the rules for generating the integrity constraints on such types, needed for enforcing the business process specification. The proposed approach has been implemented in a prototype that comprises a visual editor of BPMN diagrams and an XSLT transformation for the automatic generation of the workflow subschema and constraints corresponding to the designed BPMN diagram. The workflow extended models are currently exploited in several experiences related to the model-driven development of Web applications implementing business processes.

References

1. Aalst, W. M. P. v. d., Kumar, A.: A reference model for team-enabled workflow management systems. *Data & Knowledge Engineering* 38 (2001) 335-363
2. Brambilla, M., Cabot, J., Comai, S.: “Automatic Generation of Workflow-extended Conceptual Schemas”, *Tech. Rep.2007.40*, DEI, Politecnico di Milano.
3. Domingos, D., Rito-Silva, A., Veiga, P.: Workflow Access Control from a Business Perspective. In: *Proc. Int. Conf. Enterprise Information Systems*, 3 (2004) 18-25
4. Embley, D. W., Barry, D. K., Woodfield, S.: *Object-Oriented Systems Analysis. A Model-Driven Approach*. Yourdon Press Computing Series. Yourdon (1992)
5. IBM. MQSeries Workflow. <http://www.ibm.com/software/integration/wmqwf/>
6. Knapp, A., Koch, N., Zhang, G., Hassler, H.: Modeling Business Processes in Web Applications with ArgoUWE. In: *Proc.*, (2004) 69-83
7. OMG: UML 2.0 OCL Specification. *OMG Adopted Specification (ptc/03-10-14)*
8. OMG: UML 2.0 Superstructure Specification. *OMG Adopted Specification (ptc/03-08-02)*
9. OMG/BPMI: Business Process Management Notation v.1. *OMG Adopted Specification*
10. Oracle Workflow. <http://www.oracle.com/technology/products/ias/workflow/index.html>
11. WfMC. Workflow Management Coalition. Available: www.wfmc.org

RADAR: Research of Anomalous Data through Association Rules

Giulia Bruno¹, Paolo Garza¹, Elisa Quintarelli², and Rosalba Rossato²

¹ Dipartimento di Automatica e Informatica, Politecnico di Torino
{giulia.bruno, paolo.garza}@polito.it

² Dipartimento di Elettronica e Informazione, Politecnico di Milano
{quintarelli, rossato}@elet.polimi.it

Abstract. The anomaly detection problem has the double purpose of discovering interesting exceptions and identify incorrect data. Since anomalies are rare events which violate the frequent relationships among data, we propose a method, called RADAR (Research of Anomalous Data through Association Rules), which is based on data mining techniques, to first detect frequent “rules” from datasets, in the form of *quasi-functional dependencies* and then extract anomalies. We can discover the anomalies with respect to the quasi-functional dependencies by querying either the original database or the association rules previously mined. The analysis on this kind of anomalies can either derive the presence of erroneous data or highlight novel information which represents significant outliers of frequent rules. Experiments performed on real XML databases are reported to show the applicability and effectiveness of the proposed approach.

1 Introduction

Huge amounts of data usually hide interesting, but not a-priori known, information; moreover, data intensive sources present the problem of analyzing data, because of their size. There are two kinds of interesting knowledge to discover from these sources: frequent trends and anomalies with respect to frequent trends. Many techniques have been exploited to discover frequent trends in data (e.g., mined association rules), but also infrequent behaviors can augment the knowledge about a data source. We define such infrequent situations as *anomalies*. They are of interest since they represent errors or semantically correct, albeit infrequent, situations. In both cases detecting anomalies is a challenging task, either to allow one to correct erroneous data or to investigate the meaning of interesting exceptions. These topics are of great importance in a variety of application fields, especially for biological and clinical data, in which it is important to detect anomalies in order to clean errors or discover interesting situations needing a further investigation by specialists.

In this work we propose a method, called RADAR (Research of Anomalous Data through Association Rules), to perform anomaly detection in databases by means of data mining techniques, which allow us to discover *quasi-functional dependencies* and association rules from data. A quasi-functional dependency (i.e., approximate functional dependency derived from data [8]), represents an implication among attributes (or elements, with respect to the XML context), which frequently holds in the analyzed

datasets. For example, in a dataset describing research publications, we expect that the title of a publication always, or very often, determines the year of the publication. In particular, a quasi-functional dependency constraint is violated only by few cases (or tuples, when working on relational datasets). The few cases that invalidate the exact functional dependency constraint highlight either errors or interesting situations, which represent anomalies. In our example, an anomaly to the functional dependency we expect to extract represents two articles, with the same title, published in different years.

The main contributions of our proposal are listed in the following.

- The RADAR method allows one to extract functional and quasi-functional dependencies, not a-priori known, by directly inferring association rules from the considered data sources. The proposed approach is independent from the type of the data sources; in this work we consider both relational and XML data sources. Moreover, if the size of the considered databases is statistically significant, the discovered relations can be considered “independent” of the used instances and represent frequent rules holding on datasets having the same schema; i.e., they can suggest dependencies on the considered application domain.
- Quasi-functional dependencies are analyzed to single out anomalies in the datasets; this means that, in our approach, anomalies detection is guided by rules that frequently hold in the analyzed datasets.
- Erroneous data can be distinguished from interesting exceptions by further investigating the association rules with confidence value lower than a fixed threshold.

The paper is organized as follows. Section 2 describes the concept of quasi-functional dependency and the methodology to detect them by means of association rules. In Section 3 we show how to use a quasi-functional dependency to find anomalies. In Section 4 the experiments performed on real databases are presented. Section 5 discusses related work, while Section 6 draws conclusions and presents future developments of the proposed approach.

2 Quasi-Functional Dependencies

Functional dependencies are constraints among sets of attributes of the database. In the relational data context, a functional dependency between two sets of attributes X and Y of a relation R imposes the following constraint on an instance r of R : any two tuples t_1 and t_2 of r that agree on the value of X (i.e., $t_1[X] = t_2[X]$) must agree on the value of Y (i.e., they must also have $t_1[Y] = t_2[Y]$). The functional dependency between the two sets of attributes X and Y of R is denoted by $X \rightarrow Y$ [10].

The notion of functional dependency for XML has been introduced in [7] by using tree tuples, which describe the paths of an XML document whose schema is expressed by a DTD (Document Type Definition).

A tree tuple t in a DTD D is a function that assigns to each path in D a value that represents a node identifier, or a string (for the content of leaf elements), in such a way that t represents a finite tree with paths from D containing at most one occurrence of each path [7]. Given a DTD D , a functional dependency over D is an expression of the form $S_1 \rightarrow S_2$, where S_1, S_2 are finite nonempty subsets of paths in D . Given an

CitizenID	City	Province	...
1	Piobesi	Torino	...
2	Piobesi	Torino	...
3	Piobesi	Torino	...
4	Piobesi	Cuneo	...
5	Piobesi	Cuneo	...
6	Recetto	Novara	...
7	Recetto	Milano	...
8	Recetto	Novara	...
...

Table 1. A subset of citizen database.

Body	Head	Support	Confidence
City=Piobesi	Province=Torino	45.1%	75.2%
City=Piobesi	Province=Cuneo	14.9%	24.8%
City=Recetto	Province=Novara	28.7%	99.7%
City=Recetto	Province=Milano	0.1%	0.3%

Table 2. Examples of association rules found in the database by considering the *City* and *Province* attributes.

XML tree T valid w.r.t. D , T satisfies $S_1 \rightarrow S_2$, if for every tree tuple t_1 and t_2 in T , $t_1[S_1] = t_2[S_1]$ implies $t_1[S_2] = t_2[S_2]$.

In this work we deal with a particular subset of XML functional dependencies, which are those relating paths which reach leaf elements.

Association rules describe the co-occurrence of data items (i.e., couples (*attribute, value*)) in a large amount of collected data [4]. Rules are usually represented as implications in the form $A \rightarrow B$, where A and B are two arbitrary sets of data items, such that $A \cap B = \emptyset$. The quality of an association rule is usually measured by means of support (s) and confidence (c). Support corresponds to the frequency of the set in the dataset, while confidence corresponds to the conditional probability of finding B , having found A and is given by $c = \frac{s(A \cup B)}{s(A)}$. As demonstrated in [8], a functional dependency between two sets of attributes X and Y can be detected from data by analyzing all the association rules involving X and Y , and computing the dependency degree between them. The dependency degree of a mined functional dependency between the sets of attributes X and Y is computed according to the following formula: $p = \sum_{i \in S} s_i \cdot c_i$, where S is the set of all association rules relating attributes X and Y , s_i and c_i are respectively the support and confidence values of each rule. If the dependency degree is equal to 1, then a functional dependency has been mined. If the degree is close to one (does not decrease below a specific threshold), the attributes are defined as *quasi-functional dependent*. We introduce the notion of quasi-functional dependency by using a simple example.

Example 1. Consider a relational database of citizens, which contains information about people of a country. Each person has some attributes, among which there are the city and the province where he/she lives, as shown in Table 1. We expect to find the functional dependency $City \rightarrow Province$. Hence, we expect exclusively to find association rules with confidence 100% between the attributes *City* and *Province*. Instead, we detect also association rules with confidence lower than 100%. Examples of these rules are reported in Table 2. By computing the dependency degree between *City* and *Province*, we obtain a value of 0.95. Supposing that this value is high enough, we can state that there is a quasi-functional dependency between the two attributes; that is $City \rightarrow Province$ with a dependency degree equal to 0.95.

3 Anomaly Detection

Once the quasi-functional dependencies that involve two (or more) attributes have been detected, we must identify the anomalies that violate such dependencies. For this purpose we have two possibilities:

1. we query the original datasets to extract the instances that violate the dependencies;
2. for each quasi-functional dependency relating the sets X and Y , we query all the stored association rules that involve X and Y , with a low confidence (i.e., with a confidence lower than a fixed threshold). Such rules are those that allow us to infer that the dependency is a quasi-functional dependency, and not an exact one (see the formula to compute the degree of a dependency).

Once an anomaly has been detected, we have to investigate its nature. For this purpose we now investigate the rules that involve *City* and *Province* with a low confidence (i.e., lower than 50%).

Example 2. We investigate the rules that involve *City* and *Province* with a low confidence. Examples of such rules are:

$City=Piobesi \rightarrow Province=Cuneo$ [$s=14.9\%$; $c=24.8\%$]

$City=Recetto \rightarrow Province=Milano$ [$s=0.1\%$; $c=0.3\%$]

Both of them represent interesting cases. The first one is a correct, albeit infrequent, relationships, since there are two cities with the same name (Piobesi) in two different provinces (Cuneo and Torino). The second one is an error, since there is only a city named Recetto in the Novara province. We are able to distinguish between the two cases by analyzing the confidence value. The value of the second rule (0.3%) is at least one order of magnitude smaller than the average confidence values of the other rules; hence, we can conclude that this is an error.

3.1 Errors vs. interesting exceptions

In this section, we describe in more detail our proposal to distinguish erroneous situations from interesting anomalies. Once the association rules and the related quasi-functional dependencies have been mined from a dataset, we store them either in a relational database or as XML documents [2] on the basis of the data sources we are analyzing. The relational database of rules and quasi-functional dependencies is composed by the following tables:

RULE(ID, Sup, Conf, NumItemHead, NumItemBody)

ITEMR(IDRule, DataElement, Value, Head_Body)

CONSTR(ID, P_Degree, N_ItemHead, N_ItemBody)

ITEMD(IDConstr, DataElement, Head_Body)

The tables **RULE** and **ITEMR** allow to store association rules; each tuple of **RULE** keeps trace of the identifier (ID), support (Sup), confidence (Conf), the number of items appearing in the head (NumItemHead) and in the body (NumItemBody) of a rule. The table **ITEMR** stores the structure of each extracted rule; in particular, for each rule,

it stores the identifier (IDRule), the name and the value of an attribute/element that appears in the considered rule (DataElement and Value, respectively) and the position (i.e., head or body) of the current attribute/element in the rule itself (Head_Body). Analogously, tables **CONSTR** and **ITEMD** store quasi-functional dependencies and their structures.

Given a quasi-functional dependency $X \rightarrow Y$ (with X and Y two attributes or elements) with a dependency degree p , we can discover the related anomalies by applying one of the following two queries:

1. we use the original database where the rule has been mined and extract the tuples violating the functional dependency. By supposing to have a super table R representing the database, the SQL query we have to apply is the following:

```
SELECT X, Y FROM R WHERE X IN
(SELECT X FROM R GROUP BY X HAVING COUNT(DISTINCT (Y))>1)
```

2. we use the tables storing the association rules and apply the following query to retrieve the list of attributes and the respective values of association rules with a low confidence:

```
SELECT IDRule, DataElement, Value, Head_Body, FROM ITEMR
WHERE IDRule IN
( SELECT ID
  FROM RULE JOIN ITEMR ON ID=IDRule
  WHERE Conf<Threshold AND NumItemHead =1 AND NumItemBody =1 AND
    DataElement='X' AND HEAD_BODY='body'
  INTERSECT
  SELECT ID
  FROM RULE JOIN ITEMR ON ID=IDRule
  WHERE Conf<Threshold AND NumItemHead =1 AND NumItemBody =1 AND
    DataElement='Y' AND HEAD_BODY='head'
)
ORDER BY IDRule
```

When dealing with XML-based data sources, in order to have a homogeneous representation of data and queries, the set of association rules can be represented as an XML document with the DTD reported in Listing 1.1, while the quasi-functional dependencies are represented by an XML document with the DTD reported in Listing 1.2.

```
<?xml version="1.0" encoding="UTF-8"?>
<!ELEMENT RuleSet(AssRule+)>
<!ELEMENT AssRule (RuleBody, RuleHead)>
<!ATTLIST AssRuleNumberItemHead CDATA #REQUIRED
  NumberItemBody CDATA #REQUIRED
  support CDATA #REQUIRED
  confidence CDATA #REQUIRED>
<!ELEMENT RuleBody (item+)>
<!ELEMENT RuleHead (item+)>
<!ELEMENT Item (#PCDATA)>
<!ATTLIST Item DataElement CDATA #REQUIRED>
```

Listing 1.1. DTD associated with the XML document describing association rules.

The XQuery [3] expression to retrieve anomalies from the document of association rules, i.e., the rules with a low confidence, is the following:

```
FOR $r in doc("AssociationRule.xml")//AssRule
where $r[@confidence<Threshold] AND
$r[@NumberItemHead=1] AND $r[@NumberItemBody=1] AND
$r/RuleBody/item/@DataElement='X'AND $r/RuleHead/item/@DataElement='Y'
return $r
```

In order to discover an anomaly from the XML documents used to mine the association rules relating the elements X and Y , we should write an XQuery expression to find out X elements which are related to more than one Y element. We do not explicitly indicate here this possibility because the syntax of the XQuery expression is strictly related to the DTD of the data source.

```
<?xml version="1.0" encoding="UTF-8"?> <!ELEMENT DepSet (Constraint+)>
<!ELEMENT Constraint (Body, Head)> <!ATTLIST Constraint NumItemHead CDATA
#REQUIRED NumItemBody CDATA #REQUIRED Prec_degree CDATA #REQUIRED>
<!ELEMENT Body (item+)> <!ELEMENT Head (item+)> <!ELEMENT Item(#PCDATA)>
```

Listing 1.2. DTD associated with the XML document describing quasi-functional dependencies.

The described SQL and XML queries can be generalized when dependencies, with more than two items, are mined. These queries allow domain experts to retrieve anomalies w.r.t. quasi-functional dependencies and consequently to concentrate on a small portion of data (the anomalies) to find out interesting outlier situations, by manually discarding errors. A possibility to distinguish semantic correct anomalies from errors with a semi-automatic procedure is to apply the query to detect anomalies to other databases in the same application domain. By comparing the results of the distributed query, we can probably infer that an anomaly is an error if it does not occur in more than one data source, otherwise, if we discover the same anomaly more times we can assume it is an admissible situation that can be further analyzed. The query to discover anomalies must be submitted to a rewriting procedure to be applied to data sources represented by different models and with different schemata.

4 Preliminary Experiments

The experiments have been performed on a 3.2GHz Pentium IV system with 2GB RAM, running Kubuntu 6.10. For the itemset and association rule extraction we use a publicly available version of Apriori[4]. We validated our approach by means of experiments on the TPC-H database [1] and the DBLP database [11]. TPC-H is a suite of synthetic datasets generated from the TPC-H relational tables saved in XML format. DBLP is a real-life dataset where each transaction includes elements that characterize an article (authors, title, year, conference, etc.). Quasi-functional dependencies with a high value of p ($1 > p > 0.95$) obtained by considering attributes of the LINEITEM table (TPC-H) are reported in Table 3. Moreover, by considering the ORDERS table (TPC-H), the quasi-functional dependency $Order\text{-}date \rightarrow Order\text{-}status$ has been extracted and its dependency degree is equal to 0.978. With respect to the DBLP dataset, by means of our approach, the quasi-functional dependency $Title \rightarrow Year$ has been mined; its dependency degree is equal to 0.981. In order to investigate in details this quasi-functional dependency, we have to retrieve the association rules with a low confidence that involve the attributes *Title* and *Year*. Some of the rules with confidence lower than 100% are reported in Table 4; to preserve authors' privacy, we do not report real values of the elements *Title*, *Author*, and *Booktitle*.

The rules with a confidence lower than 100% highlight articles with the same title but different years. It is a rare situation, since different articles usually have different

Quasi-functional dependency	p
Lineitem-receipt-date \rightarrow Linestatus	0.996
Lineitem-returnflag \rightarrow Linestatus	0.987
Lineitem-commit-date \rightarrow Linestatus	0.981
Orderkey \rightarrow Linestatus	0.987

Table 3. TPC-H: dependency degree between attributes in the LINEITEM table.

Body	Head	Support	Confidence
Title='Title 1'	Year=1992	1/N	50%
Title='Title 1'	Year=1994	1/N	50%
Title='Title 2'	Year=1995	1/N	50%
Title='Title 2'	Year=1996	1/N	50%

Table 4. Examples of association rules found in the DBLP database by considering the *Title* and *Year* attributes. N is the number of articles (N=618145).

titles. We find two causes of such anomaly. The first one is when the same article is published with the same title on a conference and on a journal in different years. The second one is when the same article is published with the same title on two different conferences in different years. Both cases are interesting anomalies.

The execution times of RADAR are: 270s for the ORDER table, 4740s for the LINEITEM table, and 3729s for the DBLP database. We remember that, for each database, the proposed approach analyzes all the possible pairs of elements. Hence, the execution time increases linearly with the number of possible pairs of elements.

5 Related Work

The inference of functional dependencies from data to detect anomalies has already been applied in [6]. The authors present results on protein structure databases, without formalizing the concepts of quasi-functional dependency and without defining a general method to retrieve the anomalies. In this paper we formalize the notion of quasi-functional dependency and propose a method to retrieve anomalies and to discover their nature. We mainly focalize on the application of the proposed method to XML data.

The problem of outlier detection has been well formalized in [5] in the context of logic programming-based knowledge systems. The authors propose a basic framework where observations (outliers) are described by means of a set of facts encoding some aspects of the current status of the world, while the background knowledge of the system is described by means of a logic program. Outliers are identified on the basis of some disagreement with the background knowledge and supported by some evidence in the observed data, called witness sets. Moreover, the basic framework is extended to the case the observations of the current status of the world has to be captured by means of more complex form, i.e., no ground facts but logical rules. We mainly differ from [5] because we do infer functional dependencies automatically from datasets and do not consider such rules as part of the knowledge base. We plan to extend our approach to more general rules, by mining graph-based patterns from nested XML documents.

In [9] the authors introduce the notion of pseudo-constraints, that are predicates which have significantly few violations. The authors use this pattern to identify rare events in databases. The aim of the work is similar to our main purpose: they define this data mining pattern to detect interesting anomalies. However, our approach differs from [9] for these reasons: (i) they define the notion of pseudo-constraint on the Entity-Relationship (ER) model, whereas we use association rules to define a quasi-functional

constraint and (ii) they focus on cyclic pseudo-constraints and propose an algorithm for extracting this kind of cyclic pattern. On the contrary, our notion of dependency is an implication between sets of elements and is not related to the structure of the data source used to mine the pattern. Moreover, we do not apply our notion only to relational databases: we have generalized the method to XML datasets and experimental results confirm the generality of our proposal.

6 Conclusions and Future Work

In this work we have described an approach to discover anomalies to quasi-functional dependencies mined by using association rules. The proposal is general and can be applied in any context where the process of data cleaning is a crucial task. As an ongoing work we are applying this technique to biological datasets where protein structure sequences can be mined and anomalies must be detected. In particular, we are developing a tool that first mines functional dependencies from a database and then automatically rewrites the query in order to apply it to other relational and XML datasets.

Acknowledgment. We like to thank Elena Baralis and Letizia Tanca for the precious collaboration in the setting foundations of our research.

References

1. The TPC benchmark H. Transaction Processing Performance Council. Retrieved from <http://www.tpc.org/tpch/default>.
2. World Wide Web Consortium (1998). eXtensible Markup Language (XML). Retrieved from <http://www.w3C.org/TR/REC-xml/>.
3. World Wide Web Consortium (2002). XQuery: An XML Query Language. Retrieved from <http://www.w3C.org/TR/REC-xml/>.
4. R. Agrawal and R. Srikant. Fast algorithms for mining association rules in large data-bases. In *Proceedings of VLDB'94*, pages 478–499. Morgan Kaufmann, 1994.
5. F. Angiulli, G. Greco, and L. Palopoli. Outlier detection by logic programming. *ACM Transaction on Computational Logic*, 2006. To appear.
6. D. Apiletti, G. Bruno, E. Ficara, and E. Baralis. Data Cleaning and Semantic Improvement in Biological Databases. *Journal of Integrative Bioinformatics*, 3(2), 2006.
7. M. Arenas and L. Libkin. A Normal Form for XML Documents. *ACM Transactions on Database Systems*, 29(1):195–232, 2004.
8. E. Baralis, P. Garza, E. Quintarelli, and L. Tanca. Answering Queries on XML Data by means of Association Rules. In *Current Trends in Database Technology*, volume 3268. Springer-Verlag, 2004.
9. S. Ceri, F. Di Giunta, and P. Lanzi. Mining Constraint Violations. *ACM Transactions on Database Systems*. To appear.
10. R. Elmasri and S.B. Navathe. *Foundamentals of Database Systems*. Pearson, Addison Wesley, 2004.
11. M. Lay. The DBLP bibliography server. Retrieved from <http://dblp.uni-trier.de/xml>, 2005.

Modellazione della QoS in ambienti Web-Service con applicazioni di Video Streaming

F. Buccafurri³, L. Console⁴, P. De Meo³, M.G. Fugini², A. Goy⁴, G. Lax³, P. Lops¹, S. Modafferi², B. Pernici², C. Picardi⁴, D. Redavid¹, G. Semeraro¹, and D. Ursino³

¹ Università degli Studi di Bari, Dipartimento di Informatica

² Politecnico di Milano, Dipartimento di Elettronica e Informazione

³ Università degli Studi Mediterranea di Reggio Calabria, Dipartimento di Informatica Matematica Elettronica e Trasporti

⁴ Università degli Studi di Torino, Dipartimento di Informatica

Sommario Il Progetto Quadrantis (Quality of service, Diagnosis, Reaction, Assessment Techniques in Information Systems) si propone di modellare gli aspetti legati alla cooperazione degli e-Service concentrandosi sulle problematiche legate alla QoS e sulle possibili azioni di recovery che, a valle di una diagnosi per determinare la causa del problema, possano implementare politiche di adattività. La modellazione degli e-service è basata sull'estensione dei linguaggi disponibili in letteratura con la rappresentazione dei parametri di qualità associati ai servizi. Il contributo fornito è innovativo in quanto considera la rappresentazione semantica dei parametri di qualità sia a livello di Web Service sia a livello di "oggetto erogato". In questa accezione l'oggetto dell'e-service diventa particolarmente interessante se esso stesso è dotato di parametri di qualità e se, al di fuori della sua richiesta che viene gestita con il protocollo dei Web-Service, è possibile realizzare politiche adattive che consentano il monitoraggio e la gestione avanzata della QoS durante la sua erogazione. E' stato proposto un framework a due livelli in grado di gestire in maniera adattiva la QoS in ambienti Web-Service cooperativi, considerando in particolare il caso in cui l'oggetto di uno o più servizi nel processo sia un video streaming.

1 Introduzione

Recentemente un'interessante direzione di ricerca è quella della modellazione e gestione della Quality of Service (QoS) in ambiente Web-Service. Accanto al criterio funzionale, la QoS rappresenta infatti la guida nella scelta e nella gestione dei servizi consentendo di sfruttare al meglio le caratteristiche di dinamicità, flessibilità e possibilità di composizione, che sono gli aspetti più interessanti della Service Oriented Architecture (SOA). Il progetto Quadrantis si propone di modellare la QoS dei Web-Service introducendo l'innovativo concetto di "oggetto erogato" dal Web-Service distinguendone i parametri di qualità da quelli propri del componente erogante e, al tempo stesso, di fornire una piattaforma per la

gestione adattiva della cooperazione fra diversi Web-Service. Per adattività, nel contesto del progetto, si intende la capacità del sistema di monitorare un insieme predefinito di parametri di qualità reagendo ad eventuali violazioni di questi parametri attraverso opportune politiche di recovery. La scelta dell'insieme di operazioni di recovery da implementare è fatta a valle di una fase di diagnosi in grado di individuare la sorgente del fault che non necessariamente è nello stesso attore/componente che intercetta il sintomo.

Particolarmente interessante è il caso in cui l'oggetto dell'operazione del Web-Service è l'erogazione di un video streaming. In questo caso infatti esiste una diffusa letteratura sulla relativa QoS ed esistono sistemi ad agenti in grado di implementare politiche adattive in caso di degrado della QoS. E' quindi possibile descrivere come la QoS del Web-Service e quella dell'oggetto si influenzano. Lo scenario scelto per testare l'approccio proposto prevede quindi un processo di business all'interno del quale siano presenti dei Web-Service che erogano un video streaming.

L'articolo è strutturato nel seguente modo: nella Sezione 2 viene discusso il problema della qualità nei Web-Service, la Sezione 3 è dedicata alla presentazione dell'ontologia sviluppata mentre nella Sezione 4 viene presentato il framework per la gestione contestuale della QoS nei Web-Service e nello streaming. La Sezione 5 è dedicata alle conclusioni e agli sviluppi futuri.

2 La qualità nei Web-Service

Una delle sfide più significative per rendere la SOA realmente efficace nei moderni sistemi di business è la gestione della QoS. Come già accennato, infatti, la QoS rappresenta, accanto al criterio funzionale, una guida nella scelta e nella gestione dei Web-Service, e consente di sfruttare al meglio le caratteristiche di dinamicità, flessibilità e possibilità di composizione della SOA. In generale la QoS viene gestita attraverso contratti stipulati fra un provider ed un client. Attraverso un contratto il provider si impegna a garantire una certa QoS ed il client a pagare un certo prezzo P.

All'interno di un processo di business una violazione della QoS può avere svariate origini e sistemi adattivi che supportano il *Self-Healing* sono in grado di capire come e perchè si sia generata tale violazione e di reagire conseguentemente. In questa accezione le violazioni di QoS possono essere interpretate come fault di una specifica istanza di un processo; è quindi parte integrante delle capacità adattive del sistema la possibilità di reagire correttamente a queste situazioni.

In letteratura esistono diversi approcci per la modellazione e per la gestione della QoS. Un interessante confronto fra alcuni modelli di qualità è presentato in [16]. In generale si sta sempre di più affermando l'uso di ontologie per la rappresentazione della QoS perchè supportano: i) un vocabolario condiviso utilizzabile dalle macchine; ii) una rappresentazione della conoscenza interpretabile dalle macchine; iii) la possibilità di fare reasoning automatico sulle descrizioni della QoS.

Esistono dei sistemi il cui focus è l'utilizzo di ontologie per la gestione della QoS, ad esempio [5]. Questa attenzione alle ontologie porta ad avere modelli ricchi, ma sistemi reali piuttosto semplici. E' il caso del framework presentato in [11] che è basato su agenti che utilizzano l'ontologia per selezionare i servizi richiesti dall'utente; un interessante tentativo di integrare una rappresentazione ontologica all'interno di un sistema per la gestione della QoS è presentato in [14].

Altri modelli si focalizzano invece sulla gestione della QoS assumendone una rappresentazione piuttosto semplice ed essenzialmente basata su modelli scritti in XML, ma non necessariamente ontologici. In [2] il focus è la selezione dinamica delle risorse per la massimizzazione della QoS per l'utente finale. Le dimensioni di qualità considerate sono poche e semplici e il modello per la loro rappresentazione è un semplice file XML. In [6] vengono rappresentate la QoS offerta da un servizio e quella richiesta dall'utente e, attraverso l'uso del linguaggio Ws-Policy, vengono implementate politiche di matching che tendono a massimizzare la qualità come percepita dall'utente. In [10] gli autori propongono un framework per supportare l'esecuzione di servizi composti con una logica goal-oriented. Il lavoro [17] presenta una piattaforma middleware per la selezione e l'esecuzione di Web-Service composti in grado di supportare l'ottimizzazione sia a livello locale che a livello globale.

I modelli basati su ontologie, sopra presentati, sono generici ed hanno una naturale estensibilità che li rende capaci di supportare l'approccio proposto in questo articolo. Non esistono però modelli di gestione che esplicitamente distinguano tra qualità del servizio e qualità dell'oggetto, che è invece molto importante sia oggettivamente, che per la percezione globale della qualità che ha l'utente.

3 L'ontologia per la descrizione della QoS

L'evoluzione del Web è legata in modo imprescindibile alla rappresentazione della semantica delle informazioni in esso presenti. Tale rappresentazione permetterà lo scambio di informazioni tra applicazioni eterogenee allo scopo di abilitare le applicazioni ad automatizzare le operazioni che oggi possono essere eseguite solo manualmente [4]. Tale evoluzione deve avvenire necessariamente in modo graduale, per cui è auspicabile che le comunità di ricerca che lavorano nell'ambito del Web si confrontino con le metodologie e tecnologie sviluppate per il Semantic Web. In quest'ottica nell'ambito del progetto Quadrantis sono stati adottati costrutti ontologici per la rappresentazione della semantica associata ai parametri di qualità descritti in Tabella 1. In generale, l'ontologia costruita descrive attributi di qualità arbitrari, la natura delle associazioni che esistono tra di loro e le modalità con cui è possibile misurarle.

Le principali classi ontologiche, riportate anche in Figura 1¹, utilizzate per descrivere i parametri in esame sono le seguenti:

¹ Realizzata con il plugin per protégè Jambalaya, disponibile al sito: <http://www.thechiselgroup.org/jambalaya>

Nome	Definizione
Tempo di risposta	Ritardo tra l'istante di tempo in cui una richiesta è mandata e quello quando il risultato è ottenuto.
Qualità dei dati	Parametro relativo ai dati e ai database a cui il servizio fa riferimento, è tipicamente suddiviso in tre sottocategorie:
Timeliness	Grado di aggiornamento dei dati.
Accuracy	Valutazione della corrispondenza del dato rispetto a un dato di riferimento considerato corretto.
Completeness	Copertura dei dati scambiati rispetto alla totalità delle informazioni.
Prezzo	Importo che chi richiede un servizio deve pagare al Provider dello stesso per l'invocazione.
Availability	Probabilità che una data operazione sia realmente accessibile al momento della richiesta.
Reputation	Grandezza definita come il rapporto fra il numero di invocazioni del servizio che rispettano la QoS e il totale delle invocazioni del servizio.

Tabella 1. Parametri di qualità del livello Web-Service

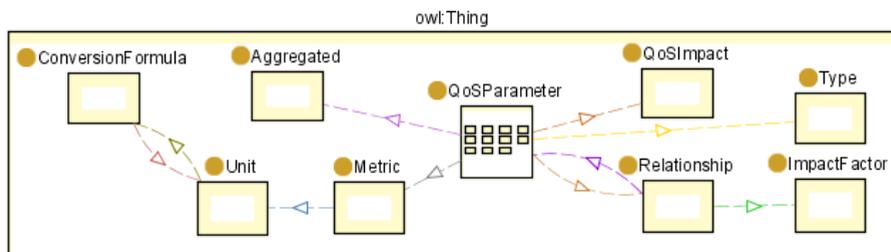


Figura 1. Le classi dell'ontologia per la rappresentazione dei parametri di qualità

- QoSParameter - tutti i parametri in esame sono sottoclassi di QoSParameter. Le relazioni di proprietà di questa classe possono essere misurabili o non misurabili e possono avere relazioni tra di loro.
- Metric - questa classe definisce la modalità con cui è possibile assegnare un valore al parametro di qualità, ed è associata alla classe QoSParameter attraverso la object property hasMetric. Ogni individual di questa classe ha due datatype property, MetricType e Value. La relazione di proprietà hasUnit lega la classe Metric alla descrizione dell'unità di misura associata per misurare le quantità del QoSParameter.
- QoSImpact - questa classe abilita alla rappresentazione di indicatori sulla qualità percepita dall'utilizzatore del servizio sulla base del valore del QoSParameter. La proprietà ad esso correlata (hasQoSImpact) abiliterebbe un sistema alla stima automatica del grado di soddisfazione dell'utente inerente un dato valore di QoSParameter.

Le relazioni di proprietà più significative per queste classi sono le seguenti:

- hasNature - indica la natura del parametro, statica o dinamica. Parametri di qualità definiti a priori che non cambiano durante l'intera sessione del servizio sono detti statici, quelli che possono variare durante l'esecuzione del servizio sono detti dinamici.
- isAggregated - esprime il legame tra due o più parametri di qualità.
- hasRelationship - lega il parametro di qualità alla classe Relationship, che descrive il modo in cui il parametro è correlato con gli altri parametri.

L'analisi eseguita sui parametri di qualità selezionati non ha evidenziato relazioni di proprietà caratterizzanti i singoli parametri, ma la rappresentazione ontologica, descritta utilizzando OWL [12], permette di specializzare le loro definizioni

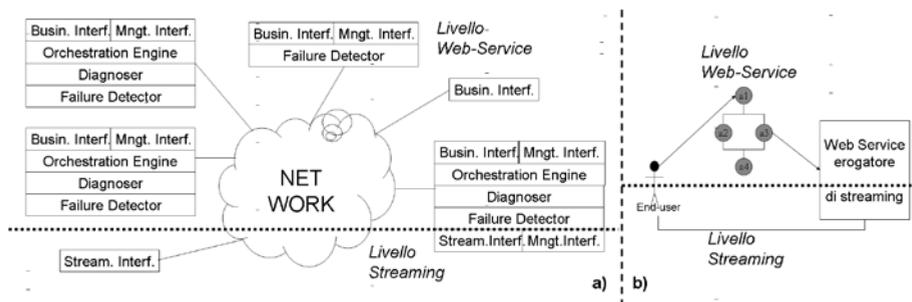


Figura 2. Architettura generale ed esempio di istanza

non appena se ne evidenzia il bisogno. L'utilizzo delle ontologie nel contesto del progetto Quadrantis è previsto solo per la rappresentazione della semantica associata ai parametri di qualità scelti. Nonostante non sia previsto l'impiego di strumenti di inferenza automatica che sfruttino il potere espressivo dei formalismi ontologici, l'utilizzo di formalismi propri del Semantic Web predispone i risultati della ricerca corrente a sviluppi futuri nella direzione proposta per l'evoluzione del Web. Nel proseguimento del progetto le ontologie saranno sfruttate anche per la rappresentazione della semantica dell'oggetto erogato dal servizio che, nel caso particolare, è lo streaming video.

4 Il framework

L'architettura proposta è formata da una serie di nodi che cooperano fra di loro per la realizzazione di un business process. Per supportare strumenti di diagnosi e recovery è necessario che i nodi abbiano determinate caratteristiche: quelle minime richieste perché siano considerati nodi attivi nelle azioni di diagnosi e recovery è che supportino il monitoring ed esportino un'interfaccia di management. In realtà non è richiesto che tutti i nodi abbiano tutte le caratteristiche: pur partecipando a tutti gli effetti alla realizzazione della parte di business, alcuni nodi possono essere coinvolti marginalmente o del tutto non coinvolti nella parte di diagnosi e recovery. Il modello generale di riferimento proposto considera uno scenario di business in cui alcuni servizi sono provider di video streaming. Per la gestione il modello prevede due livelli di comunicazione distinti: uno per l'interazione fra Web-Service ed uno per l'erogazione dei contenuti di streaming. Tale scelta è dettata dalla volontà di integrare in un unico framework le potenzialità offerte dall'approccio Service Oriented con l'erogazione di servizi Real Time. L'eterogeneità dei protocolli utilizzati e l'inadeguatezza del protocollo SOAP che sta alla base delle architetture SOA per l'erogazione di servizi real time, ha portato a separare i due canali. Tale separazione è netta nei protocolli di comunicazione, ma restano interessanti interazioni fra i due livelli soprattutto per la gestione della QoS e quindi per la reazione ad eventuali fault.

La Figura 2.a mostra un esempio di framework dove viene evidenziato che esistono due distinti livelli e che i nodi hanno caratteristiche diverse. Si ipotizza che possano esistere nodi passivi, che cioè non mostrano nulla della propria struttura interna e/o forniscono informazioni sulla QoS. Tutti gli altri nodi devono come minimo avere una struttura che supporti il monitoring e un'interfaccia di management che consenta di chiedere/attivare specifiche azioni di recovery su quel nodo. Particolarmente interessanti sono i cosiddetti *Complete Streaming Node* che sono nodi coinvolti sia al livello Web-Service che al livello Streaming. Hanno infatti l'interfaccia di business e di management ad entrambi i livelli ed in un certo senso costituiscono, da un punto di vista logico, il gateway fra i due livelli. In Figura 2.b viene mostrata una possibile istanza dell'architettura. Su input dell'utente, a livello Web-Service viene costruita e gestita l'orchestrazione e/o la coreografia dei servizi all'interno della quale trovano posto anche servizi di streaming. A livello streaming è gestita l'erogazione dei contenuti di streaming.

Si crea un canale diretto fra l'erogatore e l'utente finale. Eventuali fault o modifiche significative della QoS che vengono registrati a questo livello possono essere gestiti in prima istanza sempre a tale livello, ma possono anche influenzare e quindi essere gestiti al livello Web-Service. Il server gestore dello streaming funge quindi da gateway fra i due livelli. Supporta l'infrastruttura per il monitoring della QoS ad entrambi i livelli ed è legata alla sua struttura interna la propagazione di sintomi di fault da un livello ad un altro.

4.1 Livello Web-Service

A questo livello viene gestito il business process all'interno del quale vengono erogati contenuti di streaming. Il livello Web-Service è costituito da più nodi che collaborano all'interno di una architettura SOA. Nel caso più semplice esiste un solo nodo completo e quindi il processo è totalmente orchestrato. In casi più interessanti esistono più nodi completi e quindi all'interno del framework si realizza una coreografia.

Il monitoring. Se il servizio composto è descritto mediante una coreografia, i Web-Service che cooperano hanno solo viste parziali sul servizio complessivo e devono coordinarsi tra loro attraverso una conversazione complessa che coinvolge più partecipanti. In questa prospettiva, è possibile dotare il servizio coreografato di una funzionalità di monitoraggio del progredire della sua esecuzione, che permetta l'individuazione precoce degli errori e la notifica dei problemi ai Web-Service coinvolti, attraverso il tracciamento del comportamento conversazionale dei Web-Service che cooperano [3].

Durante l'esecuzione del servizio coreografato è presente un Monitor che viene informato relativamente ai messaggi spediti o ricevuti dai Web-Service coinvolti e al loro stato di esecuzione, attraverso l'interfaccia di management. Il Monitor utilizza queste informazioni per monitorare i parametri della QoS del livello Web-Service (per esempio, se il Monitor, interagendo con l'interfaccia di management del servizio che - secondo la coreografia - dovrebbe inviare un messaggio, si accorge che si trova in uno stato tale da non poter effettuare l'invio, può rilevare - prima del timeout del ricevente - un ritardo che può causare una violazione di

QoS) e per controllare se il servizio progredisce correttamente, cioè se il flusso dei messaggi tra i Web-Service rispetta la coreografia (accorgendosi, per esempio, se i messaggi non vengono inviati nell'ordine previsto). Se si verifica un errore, il Monitor valuta se il servizio coreografato è in grado di proseguire e informa i Web-Service partecipanti, al fine di metterli in grado di reagire al problema che si è verificato. Per mantenere il framework il più generale possibile, assumiamo che il Monitor non abbia alcuna informazione relativa all'implementazione interna dei Web-Service. I log dei messaggi scambiati durante l'esecuzione del servizio composto vengono anche utilizzati per popolare l'ontologia che rappresenta la QoS. I valori monitorati costituiscono inoltre l'input del predittore che sarà in grado di prevedere il verificarsi di fault. Poichè attualmente le informazioni individuate inerenti i parametri di qualità sono numeriche saranno impiegate tecniche classiche di machine learning supervisionato che, sulla base delle osservazioni provenienti dai file di log, forniranno indicazioni su possibili fault di qualità.

La comunicazione fra i nodi. L'interfaccia che i nodi espongono a questo livello è costituita dal WSDL per la parte di business e da WSDM per la parte di management. Attraverso tale interfaccia è possibile ottenere informazioni sullo stato del processo e specificamente delle sue variabili pubbliche, ma è anche possibile chiedere lo stop momentaneo dell'esecuzione di una specifica istanza o ancora specifiche azioni di recovery come la sostituzione di un servizio invocato. Ciascun nodo dichiara quindi quali azioni di recovery supporta che possono quindi essergli richieste da altri nodi. Se un nodo non espone un'interfaccia di management o le operazioni che offre sono inadeguate a risolvere il problema verificatosi nel sistema, esso può essere totalmente sostituito da un altro nodo che abbia caratteristiche compatibili.

La diagnosi. In sistemi complessi e composti da diversi attori, per individuare la causa di un fault è necessaria una attività di diagnosi. Nel progetto Quadrantis si è deciso di utilizzare l'approccio *model-based* (MBD, *Model-Based diagnosis*) [8]. La maggior parte degli approcci *model-based* sono basati su un modello orientato ai componenti del sistema da diagnosticare. Il modello del sistema complessivo risultante è in grado di predire, o quanto meno di vincolare, l'effetto del comportamento corretto o scorretto di un componente, anche su variabili che non sono direttamente in relazione con il componente in esame. Il ragionamento diagnostico identifica le diagnosi come assegnamenti di modalità di comportamento ai componenti, che spieghino un certo insieme di osservazioni (valori per le variabili osservabili). Un motore diagnostico, in generale, esplora lo spazio delle diagnosi candidate ed effettua una discriminazione tra i candidati, eventualmente suggerendo l'acquisizione di ulteriori informazioni utili allo scopo. La discriminazione può essere effettuata esclusivamente in vista di un obiettivo diagnostico, per esempio la selezione di un'azione di riparazione appropriata.

Nella diagnosi *consistency-based* [15] utilizzata nel nostro approccio, una diagnosi è un assegnamento di modalità di comportamento ai componenti tale da essere consistente con le osservazioni. Per i modelli statici, questo significa che la diagnosi candidata predice, per le variabili osservabili, un insieme di possibili va-

lori che includono quello osservato. L'approccio alla diagnosi che viene proposto è *decentralizzato*. In particolare: i) ad ogni servizio è associato un *diagnostizzatore locale* che deve fornire al diagnosticatore globale (vedi punto seguente) le informazioni necessarie per identificare le cause del malfunzionamento (del servizio globale); ii) esiste un *diagnostizzatore globale* in grado di invocare i diagnosticatori locali e di mettere in relazione le informazioni ricevute da questi.

Si è scelto di adottare questo approccio in quanto esso consente di raggruppare ricorsivamente i Web-Service in aggregati di sotto-servizi, nascondendo i dettagli dell'aggregazione ai servizi dei livelli più alti. Il diagnosticatore globale deve conoscere esclusivamente le interfacce dei diagnosticatori locali e quindi condividere un protocollo di comunicazione con essi. Questo significa che il modello di un servizio resta privato al suo diagnosticatore locale e non deve essere reso visibile agli altri diagnosticatori locali o al diagnosticatore globale stesso.

Le politiche adattive. Sono al momento in corso di definizione le politiche di recovery che vengono messe in atto a fronte di un fault di QoS. Bisogna ricordare che il monitoring delle QoS del livello Web-Service e del livello streaming è totalmente disaccoppiato mentre le azioni di recovery possono influenzarsi.

Se la QoS a livello Web-Service si degrada in relazione ad operazioni non legate allo streaming, verranno attuate politiche di recovery per provare a risolvere il problema. Tali politiche possono o meno coinvolgere il task del processo che prevede l'erogazione dello streaming. Nel secondo caso l'erogazione continua perchè vuol dire che nulla osta a che prosegua lo streaming; nel primo caso invece l'operazione di erogazione dello streaming viene considerata a tutti gli effetti una operazione di un servizio composto e può quindi subire operazioni di recovery come ad esempio quelle definite in [7,13]. I Complete Streaming nodes offrono a livello Web-Service anche azioni specifiche legate allo streaming.

4.2 Livello Streaming

Il server gestore dello streaming funge da gateway fra i due livelli. A livello di streaming viene creato un canale apposito che utilizza i protocolli propri del video streaming. La comunicazione fra il lato client e quello server avviene attraverso una struttura ad agenti il cui compito è anche quello di implementare azioni di adattamento dinamico della QoS basate sulle preferenze dell'utente.

Il monitoring. Il problema di monitorare la banda gioca un ruolo chiave nei processi di gestione e regolazione della Qualità del Servizio di un'applicazione di streaming. Una metodologia per la stima della banda in un ambiente multimediale si basa sul concetto di *probing packet*. In particolare, si consideri una rete di calcolatori e si supponga di avere un sender S e un receiver R ; per misurare la banda disponibile per trasmettere dati da S a R supponiamo che il sender invii dei pacchetti (detti *probing packet*) al receiver. In particolare, diciamo R_p la velocità con cui il sender emette dei pacchetti e con B la banda da stimare. Supponiamo di indicare con ΔD il ritardo associato alla ricezione di due pacchetti consecutivi. Si può dimostrare [9] che se il probing rate è inferiore alla banda da stimare, allora il ritardo ΔD è nullo. Questo risultato permette di progettare un algoritmo iterativo per la stima della banda B [1,9].

La comunicazione fra i nodi. Le interfacce dei nodi interessati allo streaming, sia dal lato server che da quello client sono gestite tramite un'architettura ad agenti ed il linguaggio utilizzato per le comunicazioni è basato sullo standard ACML. I Complete Streaming Nodes forniscono inoltre all'interno dell'interfaccia di management del Web-Service primitive specifiche per l'interazione con lo streaming. Questa caratteristica garantisce agli attori del livello Web-Service un ulteriore strumento per le loro politiche adattative e di recovery. Ad esempio semplicemente interrompere momentaneamente lo streaming perchè è necessaria un'interazione con l'utente e poi riprenderla subito dopo è una situazione realizzabile solo con un'interfaccia come quella appena proposta. .

Le politiche adattive. Se la QoS a livello di streaming si degrada, gli agenti intelligenti preposti alle politiche adattive proveranno a modificare i parametri per provare a fare rientrare i valori in un range accettabile. Se tali politiche falliscono il fault verrà inoltrato dal Server di streaming al livello Web-Service. Il fault diventa quindi il fault di una operazione definita nell'interfaccia WSDL del server di streaming. Verranno applicate le consuete tecniche per le recovery a livello Web-Service come ad esempio quelle definite in [13].

5 Risultati preliminari e ricerca in corso

Nel primo anno di attività del progetto si è definita l'ontologia per i parametri di qualità di un Web-Service ed è in corso di definizione quella per i parametri di qualità di un video streaming. Si è definito il framework generale presentato nella Sezione 4 e si sta studiando il problema di learning che sarà alla base del funzionamento del classificatore utilizzato per la predizioni di possibili fault di QoS. E' infine allo studio una definizione precisa dell'interazione fra le QoS dei due livelli. I punti dove avviene il contatto fra le due QoS sono l'utente e il Server di streaming. La QoS dello streaming viene rappresentata nel sistema come QoS dell'oggetto del Web-Service. E' possibile che dal livello Web-Service vengano richieste misurazioni della QoS dell'altro livello, ma in generale ciascun livello in fase di analisi considera la QoS dell'altro come un valore binario. E' possibile invece ipotizzare un'interazione più sofisticata durante l'attuazione di politiche adattive. E' importante notare che il rapporto fra violazioni di QoS non è simmetrico: è infatti presumibile che anche se il video streaming fallisce l'intero processo di business possa ripararsi, al limite sostituendo il Web-Service erogatore; non è invece plausibile che, a fronte di un fallimento dell'intero business process a causa di un errore critico, il Server di streaming continui ad erogare i suoi contenuti.

Ringraziamenti

Questo lavoro è parzialmente supportato dal Ministero dell'Università e della Ricerca nell'ambito del progetto PRIN 2005 Quadrantis.

Riferimenti bibliografici

1. M. Allman. Measuring end-to-end bulk transfer capacity. In *Proc. ACM SIGCOMM Workshop on Internet Measurement*, pages 139–143. ACM Press, 2001.
2. D. Ardagna and B. Pernici. Dynamic web service composition with QoS constraints. *Int. J. Business Process Integration and Management*, 1(4), 2006.
3. L. Ardissono, R. Furnari, A. Goy, G. Petrone, and M. Segnan. Monitoring choreographed services. In *Proc. Intl. Joint Conf. on Computer, Information, System Sciences, and Engineering CISSE 2006*, 2006.
4. T. Berners-Lee, J. Hendler, and O. Lassila. The semantic web. *Scientific American.*, 284(5), 2001.
5. C. Cappiello, B. Pernici, and P. Plebani. Quality-agnostic or quality-aware semantic service descriptions? In *W3C Workshp on Semantic Web Service Framework*, Innsbruck, Austria, 2005.
6. M.G. Fugini, P. Plebani, and F. Ramoni. A user driven policy selection model. In *Proc. Intl. Conference on Service Oriented Computing ICSOC*, Chicago, IL, 2006.
7. R. Hamadi and B. Benatallah. Recovery nets: Towards self-adaptive workflow systems. In *Proc. Intl. Conf. on Web Information Systems Engineering (WISE)*, pages 439–453, Brisbane, Australia, 2004.
8. W. Hamscher, L. Console, and J. de Kleer, editors. *Readings in Model-Based Diagnosis*. Morgan Kaufmann, 1992.
9. M. Jain and C. Dovrolis. End-to-end available bandwidth: measurement methodology, dynamics, and relation with tcp throughput. *IEEE/ACM Trans. on Networking*, 11(4):537–549, 2003.
10. A. Lazovik, M. Aiello, and M.P. Papazoglou. Planning and monitoring the execution of web service requests. In *Proc. Intl. Conf. on Service-Oriented Computing (ICSOC)*, pages 335–350, Trento, Italy, 2003.
11. E. M. Maximilien and M. P. Singh. A Framework and Ontology for Dynamic Web Services Selection. *IEEE Internet Computing*, 08(5):84–93, 2004.
12. D.L. McGuinness and F. van Harmelen. OWL Web Ontology Language - Overview. W3C Recommendation, February 2004. <http://www.w3.org/TR/owl-features/>.
13. S. Modafferi and E. Conforti. Methods for enabling recovery actions in WS-BPEL. In *Proc. Intl. Conf. on Cooperative Information Systems (COOPIS)*, Montpellier, France, 2006.
14. I.V. Papaioannou, D.T. Tsesmetzis, I.G. Roussaki, and M.E. Anagnostou. A QoS ontology language for web-services. In *Proc. of IEEE Int. Conf. on Advanced Information Networking and Applications AINA*, pages 101–106, Los Alamitos, CA, USA, 2006. IEEE Computer Society.
15. R. Reiter. A theory of diagnosis from first principles. *Artificial Intelligence*, 32(1):57–95, 1987.
16. J. Ruckert and B. Paech. Web service quality descriptions for web service consumers. In *Proc. Intl. Conf. on Quality Engineering in Software Technology CONQUEST*, 2006.
17. L. Zeng, B. Benatallah, A.H.H. Ngu, M. Dumas, J. Kalagnanam, and H. Chang. Qos-aware middleware for web services composition. *IEEE Trans. Software Eng.*, 30(5):311–327, 2004.

Containment of Conjunctive Object Meta-Queries

Andrea Cali and Michael Kifer

¹Faculty of Computer Science, Free University of Bozen-Bolzano, piazza Domenicani
3 I-39100 Bolzano, Italy

²Department of Computer Science, State Univ. of New York at Stony Brook, Stony
Brook, NY 11794-4400, U.S.A.

¹ac@andreacali.com, ²kifer@cs.sunysb.edu

Abstract. We consider¹ the problem of query containment over an object data model derived from F-logic, a formalism that has generated considerable interest as a means for building ontologies and for reasoning on the Semantic Web; F-logic queries, differently from relational database queries, can mix the data-level and the meta-level in simple and useful ways. We show that, even for queries over meta-information together with data, this problem is decidable for non-recursive conjunctive queries, and we provide relevant complexity results.

1 Introduction

The query containment problem is a question of whether the result of one query is always contained in the result of another. This problem has attracted considerable interest in the database and knowledge representation communities. In databases, query containment is key to query optimization and schema integration [2, 12, 18], and in knowledge representation it has been widely used in Description Logic [3] for object classification, schema integration, service discovery, and more [7, 16].

The most interesting and practically significant instance of the problem arises when queries are posed against databases that satisfy constraints. In this case, query results that are otherwise not contained within each other might become contained if we restrict the attention to databases that satisfy a given set of constraints. A study of this problem was pioneered by Johnson and Klug in [12] for functional and inclusion dependencies, and then further studied in other works [6, 4]. In dealing with this problem, an important issue is the form of the constraints and where they are coming from, since expressive constraints may easily make the problem undecidable. In practice, constraints typically come from design tools that follow certain methodology, such as the Entity-Relationship Model [8, 4]. The present paper takes the same approach and considers the constraints that typically arise from object-oriented design. The specific data model that we use comes from F-logic [14]—a knowledge representation formalism that has

¹ The material presented in this paper has been published as [5] in the 32nd Int. Conference on Very Large Data Bases (VLDB 2006)

generated considerable interest in the academia, within various standardization efforts, and commercially as a means for building ontologies and for reasoning on the Semantic Web. Though query containment over object databases has been studied before [15], F-logic queries have *meta-querying* capability, i.e., it can query data *and schema* in a uniform way. This property is considered important in knowledge integration and service discovery on the Semantic Web.

Here, we show that query containment is decidable for a subset of conjunctive F-logic meta-queries, and is in NP. This subset, which we call *F-logic lite*, excludes negation and default inheritance, and allows only a limited form of cardinality constraints. This result complements the earlier works on the subject, such as [6, 4], because F-logic queries and the associated constraints are different. For instance, decidability does not follow from [6] because conjunctive F-logic queries involve certain recursion, unions, and joins of ternary predicates, which are not allowed in [6].

The practical upshot of these results is that they pave the way to query optimisation in ontology integration, Web service modeling and discovery, and the Semantic Web.

2 Preliminaries

F-logic was introduced in [13, 14] as a formalism for object-oriented deductive databases, and was implemented in the FLORA-2 and FLORID knowledge representation systems [20, 10, 11] as well as commercially [19].

Basic F-logic notation. Unlike classical predicate calculus, F-logic has special atomic formulas to represent the various object-oriented concepts that are common to object-oriented and frame-based systems. For instance, `john:student` states that object `john` is a member of class `student`; `freshman::student` and `student::person` state that class `freshman` is a subclass of the class `student` and `student` is a subclass of `person`. These statements imply, for instance, that `john:person` (`john` is a student) and `freshman::person` (class `freshman` is a subclass of `person`) are true statements.

A statement of the form `john[age->33]` means that object `john` has an attribute, `age`, whose value is `33`. Actually, this really means that `33` is just *one of the values* of the attribute `age` — to say that `33` is the *only* value, one would need a cardinality constraint, as explained later.

Common constraints such as *type constraints* and *cardinality constraints* are specified via so called *signature* statements. A typical signature has the form `person[age*=>number]`. It says that the attribute `age` of class `student` has the *type* `number` and that this type is inherited by subclasses and class instances of `person`. This acts as a constrain on the statements of the form `john[age->33]`. That is, for every object in class `person` the value of the attribute `age` must be an object of class `number`.

Cardinality constraints can also be specified using signature statements. For instance, to say that the attribute `age` has at most one value, one would write `person[age {0:1} *=> number]`. Another frequently used cardinality

constraint states that a certain attribute in a class is **mandatory**, i.e., it must have at least one value on any object in the class. For instance, to say that the **name** attribute is mandatory in class **person**, we write `person[name {1:*} *=>string]`.

As mentioned in the introduction, F-logic treats object data and meta-data in a uniform way. This is primarily manifested in the following two ways: (i) classes are also objects; (ii) variables can occur anywhere an object, an attribute, or a class is allowed.

Examples of meta-queries. Meta-querying is a commonly acknowledged need in knowledge representation, especially in the Semantic Web. Consider the following rule:

$$q(A,B) \text{ :- } T1[A*=>T2], T2::T3, T3[B*=>_].$$

As usual, `:-` here denotes Datalog implication. It defines a set of pairs (A,B) of attributes that are *joinable* in a path expression of the form $A.B$ (that is, the range of A is contained in the domain of B). If we now examine the rule

$$qq(A,B) \text{ :- } T1[A*=>T2], T2[B*=>_].$$

we will see that the query containment $q \subseteq qq$ holds.

Low-level encoding of F-logic primitives. Our actual theoretical development will use an encoding, relying on the equivalence result of [14], of the semantics of a subset of F-logic using the standard logic programming notation. The subset of F-logic which we will consider in this paper will be referred to as *F-logic Lite*. This subset is characterized by the absence of negation and nonmonotonic features of F-logic (such as default inheritance) and by allowing only the cardinality constraints of the form $\{1:*\}$ — a constraint that says that the corresponding attribute in mandatory — and of the form $\{0:1\}$ — a constraint that marks functional (single-valued) attributes.

The encoding, uses the following predicates, whose set we will denote by P_{FL} :

- `member(O,C)`: object O is a member of class C .
This is the encoding for $O : C$.
- `sub(C1,C2)`: class C_1 is a subclass of class C_2 .
This encodes the statement $C_1 :: C_2$.
- `data(O,A,V)`: attribute A has value V on object O . This is the encoding for $O[A \rightarrow V]$.
- `type(O,A,T)`: attribute A has type T for object O (recall that in F-logic classes are also objects). This encodes the statements of the form $O[A*=>T]$.
- `mandatory(A,O)`: attribute A is mandatory for object (class) O , i.e., it must have at least one value for O . This is an encoding of statements of the form $O[A\{1:*\}*>_]$.
- `funct(A,O)`: A is a functional attribute for the object (class) O , i.e., it can have at most one value for O . This statement encodes $O[A\{0:1\}*>_]$.

Note that this encoding places meta-data (classes and attributes) and object data at the same level, which is needed for supporting F-logic meta-queries. This encoding is also related to, but is slightly different from, the usual encoding of RDF in first-order logic.

We can now formulate the axioms that represent the low-level encoding of the F-logic primitives discussed above in standard predicate notation. We annotate each rule in the encoding to make it easier to follow.

(1) *Type correctness:*

$\text{member}(V, T) :- \text{type}(O, A, T), \text{data}(O, A, V).$

This encodes the semantics of the constraint that an F-logic signature like $O[A \text{ *}\Rightarrow T]$ imposes on a statement like $O[A \rightarrow V]$; that is, that V must be of type T .

(2) *Subclass transitivity:*

$\text{sub}(C_1, C_2) :- \text{sub}(C_1, C_3), \text{sub}(C_3, C_2).$

This rule encodes the fact that the subclass relationship is transitive.

(3) *Membership property:*

$\text{member}(O, C_1) :- \text{member}(O, C), \text{sub}(C, C_1).$

This is the usual property that relates class membership and subclass relationship: $O:C$ and $C::C_1$ imply $O:C_1$.

(4) *Functional attribute property:*

$V = W :- \text{data}(O, A, V), \text{data}(O, A, W), \text{funct}(A, O).$

This rule states that if we have $O[A \rightarrow V]$, $O[A \rightarrow W]$, and the attribute A is single-valued then V must equal W .

(5) *Mandatory attributes definition:*

$\forall O, A \exists V \text{data}(O, A, V) :- \text{mandatory}(A, O).$ This states that mandatory attributes must have at least one value. Note that this is *not* a Datalog rule (not even a Horn rule) because it has an existentially quantified variable in the head.

(6) *Inheritance of types from classes to members:*

$\text{type}(O, A, T) :- \text{member}(O, C), \text{type}(C, A, T).$

This rule expresses the usual property of type inheritance: the type of an attribute is inherited from superclasses to class members.

(7) *Inheritance of types from classes to subclasses:*

$\text{type}(C, A, T) :- \text{sub}(C, C_1), \text{type}(C_1, A, T).$

This states that subclasses inherit types from superclasses.

(8) *Supertyping:*

$\text{type}(C, A, T) :- \text{type}(C, A, T_1), \text{sub}(T_1, T).$

This states that $T_1::T$ and $C[A \text{ *}\Rightarrow T_1]$ entails $C[A \text{ *}\Rightarrow T]$. This is also one of the usual properties of typing: if an attribute has certain type then any supertype of that type will also do.

(9) *Inheritance of mandatory attributes to subclasses:*

$\text{mandatory}(A, C) :- \text{sub}(C, C_1), \text{mandatory}(A, C_1).$

This states that a mandatory attribute of a class is also a mandatory attribute of its subclasses.

(10) *Inheritance of mandatory attributes from classes to their members:*

$\text{mandatory}(A, O) :- \text{member}(O, C), \text{mandatory}(A, C).$

Like in (9), but this time inheritance of the mandatory property is to class members rather than subclasses.

(11) *Inheritance of functional property to subclasses:*

$\text{funct}(A, C) \text{ :- sub}(C, C_1), \text{funct}(A, C_1).$

If A is a single-valued attribute in a class then it must be single-valued in the subclasses of that class.

(12) *Inheritance of functional property to members:*

$\text{funct}(A, O) \text{ :- member}(O, C), \text{funct}(A, C).$

Like (11), but this time inheritance of the single-valued property is to class members.

In the following, we will denote the above set of rules by Σ_{FL} . We will also refer to the i -th rule as ρ_i . The above statements are all Datalog rules except ρ_4 and ρ_5 . Rule ρ_4 uses equality in the head and ρ_5 has an existential quantifier in the rule head and thus invents new (fresh) values. Also, most of the rules are recursive. Additional properties of this encoding are studied in the next section.

Thanks to the above encoding, we can express any F-logic Lite database and its schema as a relational database augmented with a set of rules for deriving new information and for expressing constraints. We shall consider *only* the databases that satisfy the above set of rules.

Query containment. The query containment problem for F-logic Lite can be now stated as follows. Given a pair of meta-queries, q_1 and q_2 , over the predicates P_{FL} of the above encoding of F-logic Lite, we say that q_1 is contained in q_2 with respect to Σ_{FL} , denoted $q_1 \subseteq_{\Sigma_{FL}} q_2$, if for every database B that satisfies Σ_{FL} we have $q_1(B) \subseteq q_2(B)$, where $q(B)$ denotes the result of query q on B . We will focus on positive *conjunctive queries* [1].

3 Decidability of Containment

In this section we show that containment of object meta-queries is decidable, and we give a nondeterministic polynomial time algorithm for checking containment between two object meta-queries.

The chase [17, 12] is a tool for representing databases that satisfy certain dependencies, and it is used to check implication of dependencies and containment of queries under dependencies. Given a database, the chase is constructed by a sort of *repair* of the database w.r.t. the rules that are not satisfied. In particular, in our case, violations of all rules except ρ_4 are repaired with the *addition* of suitable tuples, while violations of ρ_4 are repaired by *equating* constants that are not equal. The rules in $\Sigma_{FL} - \{\rho_4\}$ are called *tuple-generating dependencies*, while ρ_4 is an *equality generating dependency*. The query q to be “chased” is treated as a database, and new tuples (or conjuncts) are added according to the rules. The chase of a query q , denoted $\text{chase}_{\Sigma_{FL}}(q)$, is a database constructed starting from $\text{body}(q)$.

As an example, consider the following meta-query:

$$q(V_1, V_2) \text{ :- data}(O, A, V_1), \text{data}(O, A, V_2), \\ \text{funct}(A, C), \text{member}(O, C)$$

In the construction of $chase_{\Sigma_{FL}}(q)$, rule ρ_{12} will add the conjunct $\mathbf{funct}(A, O)$ and then, by rule ρ_4 , we will replace V_2 with V_1 and obtain

$$q(V_1, V_1) \quad :- \quad \mathbf{data}(O, A, V_1), \mathbf{data}(O, A, V_1), \\ \mathbf{funct}(A, O), \\ \mathbf{funct}(A, C), \mathbf{member}(O, C)$$

We refer the reader to [5] for the detailed construction of the chase. Here we just briefly introduce the notion of *level*, that allows us to measure the size of a partially constructed chase: the conjuncts in the initial query are assumed to be at level 0, and a tuple added because of the presence of some conjuncts c_1, \dots, c_n will have level $k+1$, where k is the maximum level of a conjunct among c_1, \dots, c_n .

For technical reasons, we will do the chase in a special way, to allow us to concentrate on more important properties. We shall first proceed with the chase with respect to all the rules except for ρ_5 ; such a preliminary chase always terminates, since no new constant is generated. To simplify matters, we will view all tuples in $chase_{\Sigma_{FL}^-}(q)$ as being at level 0, where $\Sigma_{FL}^- = \Sigma_{FL} - \{\rho_5\}$. This will allow us to isolate the initial part of the chase.

We now informally illustrate a property of the chase that will lead us to the main result. It is not difficult to notice that, in the construction of the chase for a query q with respect to the set Σ_{FL} , the only way to have an infinite chase is the iterative application of rules ρ_5 - ρ_1 - ρ_6 - ρ_{10} . This happens when q contains at least a set of atoms specifying a cycle of mandatory attributes A_1, \dots, A_k belonging to classes T_1, \dots, T_k , respectively, where A_i is of type T_{i+1} for all $i \in \{1, \dots, k-1\}$ and A_k is of type T_1 . More precisely, we need q to have conjuncts of the following form:

$$\mathbf{mandatory}(A_1, T_1) \\ \mathbf{type}(T_1, A_1, T_2) \\ \dots \\ \mathbf{mandatory}(A_{k-1}, T_{k-1}) \\ \mathbf{type}(T_{k-1}, A_{k-1}, T_k) \\ \mathbf{mandatory}(A_k, T_k) \\ \mathbf{type}(T_k, A_k, T_1)$$

In such a case, if there is no atom in q of the form $\mathbf{data}(T_1, A_1, v)$, where v is a constant or variable, the chase process yields the following series of conjuncts:

$$\begin{array}{ll} \text{cycle 1 : } \mathbf{data}(T_1, A_1, v_1) & \text{cycle 2 : } \mathbf{data}(v_1, A_2, v_2) \\ \mathbf{member}(v_1, T_2) & \mathbf{member}(v_2, T_3) \\ \mathbf{type}(v_1, A_2, T_3) & \mathbf{type}(v_2, A_3, T_4) \\ \mathbf{mandatory}(A_2, v_1) & \mathbf{mandatory}(A_3, v_2) \\ & \dots \\ \text{cycle } k-1 : \mathbf{data}(v_{k-2}, A_{k-1}, v_{k-1}) & \text{cycle } k : \mathbf{data}(v_{k-1}, A_k, v_k) \\ \mathbf{member}(v_{k-1}, T_k) & \mathbf{member}(v_k, T_1) \\ \mathbf{type}(v_{k-1}, A_k, T_1) & \mathbf{type}(v_k, A_1, T_2) \\ \mathbf{mandatory}(A_k, v_{k-1}) & \mathbf{mandatory}(A_1, v_k) \end{array}$$

The above observation shows that the chase behaves somehow in a “periodic” way; this regularity is the key property for decidability. From [9] we know that, in order to check $q_1 \subseteq_{\Sigma_{FL}} q_2$ we need to find a homomorphism from q_2 to $\text{chase}_{\Sigma_{FL}}(q_1)$: as a consequence of the regularity of the chase, to check $q_1 \subseteq_{\Sigma_{FL}} q_2$ we only need to find a homomorphism from q_2 to to consider an initial, a priori bounded finite segment of $\text{chase}_{\Sigma_{FL}}(q_1)$. This is proved through a rather complicated series of lemmata, for which we refer the reader to [5].

Theorem 1. *Let q_1 and q_2 be two conjunctive queries over P_{FL} with the same arity. Then $q_1 \subseteq_{\Sigma_{FL}} q_2$ if and only if there exists a homomorphism that sends the conjuncts of $\text{body}(q_2)$ to conjuncts in the first $|q_2| \cdot \delta$ levels of $\text{chase}_{\Sigma_{FL}}(q_1)$, and $\text{head}(q_2)$ to $\text{head}(\text{chase}_{\Sigma_{FL}}(q_1))$, where $\delta = 2 \cdot |q_1|$.*

Finally, we characterize the computational complexity of the problem by giving an upper bound to it.

Theorem 2. *Consider two conjunctive queries q_1, q_2 on P_{FL} . Containment $q_1 \subseteq_{\Sigma_{FL}} q_2$ of q_1 in q_2 can be decided by a nondeterministic algorithm running in time polynomial in $|q_1|$ and $|q_2|$.*

4 Conclusions

In recent years, F-logic [14] has become a popular tool for building ontologies, information integration, and semantic Web services and a number of systems—both academic and commercial—have become available (see, e.g., [10]). In this paper, we considered the problem of query containment for conjunctive meta-queries over F-logic knowledge bases and have shown that the problem is decidable and is in NP. This important class of queries has not been covered yet and, we believe, a solution to this problem will open the door to new F-logic based applications in the areas of ontology modeling, information integration, and semantic Web services.

Acknowledgments. The work of Michael Kifer was supported in part by NSF grants CCR-0311512, IIS-0534419, and by U.S. Army Medical Research Institute under a subcontract through BNL. Andrea Cali’s work was supported by the EU FET Project IST-2005-7603 TONES (Thinking ONtologiES), and by the Italian PRIN project NGS (New Generation Search engine).

References

1. S. Abiteboul, R. Hull, and V. Vianu. *Foundations of Databases*. Addison-Wesley, 1995.
2. A. Aho, Y. Sagiv, and J. D. Ullman. Equivalence of relational expressions. *SIAM Journal of Computing*, 8(2):218–246, 1979.
3. F. Baader, D. Calvanese, D. McGuinness, D. Nardi, and P. Patel-Schneider. *The Description Logic Handbook*. Cambridge Univ. Press, Cambridge, UK, 2003.
4. A. Cali. Containment of conjunctive queries over conceptual schemata. In *The 11th International Conference on Database Systems for Advanced Applications (DAS-FAA 2006)*, pages 628–643, April 2006.

5. A. Cali and M. Kifer. Containment of conjunctive object meta-queries. In *32nd Int. Conference on Very Large Data Bases (VLDB 2006)*, pages 942–952, 2006.
6. D. Calvanese, G. De Giacomo, and M. Lenzerini. On the decidability of query containment under constraints. In *ACM Symposium on Principles of Database Systems*, pages 149–158, 1998.
7. D. Calvanese, G. De Giacomo, and M. Lenzerini. Description logics for information integration. In A. Kakas and F. Sadri, editors, *Computational Logic: Logic Programming and Beyond, Essays in Honour of Robert A. Kowalski*, volume 2408 of *Lecture Notes in Computer Science*, pages 41–60. Springer, 2002.
8. P. Chen. The entity-relationship model - toward a unified view of data. *ACM Transactions on Database Systems*, 1(1):9–36, 1976.
9. R. Fagin, P. G. Kolaitis, R. J. Miller, and L. Popa. Data exchange: Semantics and query answering. In *International Conference on Database Theory (ICDT)*, pages 207–224, 2003.
10. FLORA-2. The FLORA-2 Web site. <http://flora.sourceforge.net>.
11. J. Frohn, R. Himmerder, G. Lausen, W. May, and C. Schleppehorst. Managing semistructured data with FLORID: A deductive object-oriented perspective. *Information Systems*, 23(8):589–613, 1998.
12. D. Johnson and A. Klug. Testing containment of conjunctive queries under functional and inclusion dependencies. *Journal of Computer and System Sciences*, 28:167–189, 1984.
13. M. Kifer and G. Lausen. F-Logic: A higher-order language for reasoning about objects, inheritance and schema. In *ACM SIGMOD Conference on Management of Data*, pages 134–146, New York, 1989. ACM.
14. M. Kifer, G. Lausen, and J. Wu. Logical foundations of object-oriented and frame-based languages. *Journal of ACM*, 42:741–843, July 1995.
15. A. Levy and D. Suciu. Deciding containment for queries with complex objects (extended abstract). In *ACM Symposium on Principles of Database Systems*, pages 20–31, 1997.
16. L. Li and I. Horrocks. A software framework for matchmaking based on semantic web technology. In *Twelfth International World Wide Web Conference (WWW 2003)*, 2003.
17. D. Maier, A. O. Mendelzon, and Y. Sagiv. Testing implications of data dependencies. *ACM Transactions on Database Systems*, 4(4):455–469, 1979.
18. T. Millstein, A. Levy, and M. Friedman. Query containment for data integration systems. In *ACM Symposium on Principles of Database Systems*, pages 67–75, New York, NY, USA, 2000. ACM Press.
19. Ontoprise, GmbH. Ontobroker. <http://www.ontoprise.com/>.
20. G. Yang, M. Kifer, and C. Zhao. FLORA-2: A rule-based knowledge representation and inference infrastructure for the Semantic Web. In *International Conference on Ontologies, Databases and Applications of Semantics (ODBASE-2003)*, November 2003.

Query optimisation for web data sources: minimisation of the number of accesses

Andrea Cali, Davide Martinenghi, and Domenico Carbotta

Free University of Bozen-Bolzano - Faculty of Computer Science
Piazza Domenicani, 3 - I-39100 Bolzano, Italy
ac@andreacali.com, martinenghi@inf.unibz.it,
Domenico.Carbotta@stud-inf.unibz.it

Abstract. When relational data have access constraints that require certain attributes to be selected in queries, as in the case of (wrapped) Web sources accessible via forms, a recursive query plan is needed to answer queries at best. We present a query plan optimisation technique for several classes of queries that minimises the number of accesses according to a novel, strong notion of minimality. We provide experimental evidence of the effectiveness of our technique.

1 Introduction

In the context of integration of Web data [3], data are often accessible only via forms, where typically certain fields are required to be filled in by the user in order to obtain a result. For example, no online shop would return a list of all items to a request posed by a user who leaves all fields of a form empty, while searching for products. Analogously, in legacy systems where data are scattered over several files, data may be wrapped and masked as relational tables that have similar access limitations, due to the way the data are organized in the files.

Limitations on how sources can be accessed significantly complicate query processing [9, 5, 4, 2]. As shown in [9, 5, 7, 6], query answering in the presence of access limitations in general requires the evaluation of a recursive query plan. The problem of query containment under access limitations is addressed in [8].

Since accessing data sources is costly, especially in the case of Web data, a fundamental issue is how to reduce the *number* of accesses to the sources while still returning all obtainable answers to a query. Several optimisations that can be made during the query plan generation are discussed in [5, 7, 6]. However, these works have serious limitations: in particular, in [6], the optimisation is applicable only to a class of queries that is a proper subset of the class of union of conjunctive queries, and that is unfit to express most reasonable real-world queries. Since this class of queries is not expressive enough to capture conjunctive queries, the problem of optimising query answering when the query over the sources is an arbitrary conjunctive query is left open. In [1], a run-time optimisation technique, that exploits the information encoded in database dependencies, is presented.

In this paper we address the problem of query plan optimisation under access limitations, considering a strong notion of minimality that is not found in the literature and captures, we believe, a correct notion of the “best” plan. We first present an optimisation for *conjunctive queries*, and then we extend it to *unions of conjunctive queries with negation*. Our technique makes use heavily of the structure of the query to ensure that the optimised query plan, once executed on arbitrary data, avoids unnecessary accesses. We provide experimental results that confirm the effectiveness of our solution.

2 Preliminaries

We consider relations with attributes over *abstract domains*, which have an underlying concrete domain, but represent information at a higher level of abstraction, which distinguishes, e.g., strings representing person names from strings representing plate numbers. Relation schemata are adorned with an *access pattern*, i.e., a sequence of *i* and *o* symbols, indicating that the corresponding attribute's mode is *input* or *output*. If the access pattern does not contain any *i* then the relation is said to be *free*. The input attributes indicated in the access pattern are those that must be bound by a value in order to query the relation. For example, in relation r with schema $r^{ooi}(A_1, A_2, A_3)$, the first two attributes, of abstract domains A_1 and A_2 , are output attributes, while the third attribute is an input attribute and thus has to be selected with a value for r to be queried.

Data extraction is characterized by the notion of *access* to a relation r , i.e., a selection query over r selecting all input attributes of r ; the answer to an access is called *extraction*. A relation can be accessed either if it has no input attributes or if some constants that can bind its input attributes are known. Then, as soon as new constants are extracted with an access, these can be used to make more accesses. Such a sequence of accesses is called *access plan*. A query plan Π for a query q is *sound* whenever, for all database D , all tuples obtained for D by Π are in q^D . We are interested in sound and complete query plans, i.e., those that return all correct answers that can be obtained despite the access constraints.

In order to retrieve all obtainable tuples, relations may need to be accessed in a particular order, and in [5] an algorithm is presented that, given a query over the relations, retrieves all the obtainable tuples in the answer to the query. Such an algorithm encodes in Datalog the limitations on the relations that must be respected during query evaluation and populates *cache* relations with all obtainable values for any given abstract domain.

Example 1. Consider the schema $\mathcal{R} = \{r_1^{io}(A, C), r_2^{oi}(C, B), r_3^{oi}(B, C)\}$ and the query $q(B) \leftarrow r_1(a_1, C), r_2(C, B)$. Assume a DB with $r_1 = \{\langle a_1, c_1 \rangle, \langle a_1, c_3 \rangle, \langle a_3, c_3 \rangle\}$, $r_2 = \{\langle c_1, b_1 \rangle, \langle c_2, b_2 \rangle, \langle c_3, b_3 \rangle\}$, $r_3 = \{\langle b_1, c_1 \rangle, \langle b_1, c_2 \rangle\}$. Starting from a_1 and making all possible accesses, we obtain the answer $\langle b_1 \rangle$; instead, $\langle a_3, c_3 \rangle$ and $\langle c_3, b_3 \rangle$ could not be extracted, so answer $\langle b_3 \rangle$ is not obtainable. ■

3 Dependency graphs and relevant sources

The dependencies between input and output arguments are represented in *d-graphs*. With this we will be able to only access sources that do contribute to the answer, i.e., are *relevant*. Irrelevant sources can then be definitively excluded from the access plan, independently of the database instance, thus reducing the number of accesses that need to be made to answer a query. We base the construction of the d-graph on constant-free queries; constants can be easily eliminated by creating for each constant a new source that only contains that constant as the only tuple.

The set of nodes of a d-graph $G_q^{\mathcal{R}}$ for a query q over \mathcal{R} is determined as follows. For each atom in q , we have a set of nodes (called a *source*) in $G_q^{\mathcal{R}}$, one for each argument of the corresponding source; we call such nodes *black*. Moreover, for each source in \mathcal{R}

not appearing in q , we have a set of nodes (also called a *source*), one for each argument of the source; we call such nodes *white*. Both types of nodes have two labels, the access mode (“*i*” or “*o*”) and the abstract domain. $G_q^{\mathcal{R}}$ has an arc, denoting dependencies between source attributes from u to v whenever they have the same abstract domain, u is output and v is input. A source is *free* if none of its nodes has input access mode. Clearly, free sources can be accessed with no restriction.

We indicate with $\text{outArcs}(u, G)$ the set of outgoing arcs from any node in the same source as node u , for a d-graph G . We can have paths in G , indicating what chains of accesses need to be made to access sources that are not free. A sequence $u_1 \hat{\curvearrowright} v_1, \dots, u_n \hat{\curvearrowright} v_n$ of arcs in G is called a *d-path* for G if, for $2 \leq i \leq n$, $u_i \hat{\curvearrowright} v_i \in \text{outArcs}(v_{i-1}, G)$; the d-path is *cyclic* if $u_1 \hat{\curvearrowright} v_1 \in \text{outArcs}(v_n, G)$. We want to prune unneeded arcs from the d-graph and from it we generate a query plan that is guaranteed to make only accesses that are necessary for extracting all obtainable answers.

The problem of efficiently answering queries under access limitations has been addressed in [5, 6] for the class of *connection queries*, i.e., those unions of conjunctive queries (UCQs) that mandatorily have a join between any two attributes with the same domain. Unfortunately, connection queries do not cover conjunctive queries (CQs), and many interesting CQs are not connection queries. We fully cover CQs and, in Section 5, extend our results to UCQ^- queries.

We may eliminate some arcs thanks to the presence of joins in the query. We say that an arc $u \hat{\curvearrowright} v$ is *strong* whenever (i) both u and v are black, (ii) u and v correspond to two variables which are joined in the query, and (iii) v 's source is not needed to provide arbitrary values to other relations used in the query; all other arcs are called *weak* arcs. Now, in the presence of a strong arc, the join indicates that *all* the useful tuples that can be retrieved from v 's relation are extracted using *only* values coming from u . Therefore, whenever a node has an incoming strong arc, all the incoming weak arcs can be deleted. We say that such weak arcs are *dominated* by the strong arc(s).

Let us call *candidate strong* arc any arc whose nodes are black and whose corresponding variables are joined in the query; let us indicate with $\text{arcs}(G_q^{\mathcal{R}})$ the set of all arcs in d-graph $G_q^{\mathcal{R}}$, and with $\text{cand}(G_q^{\mathcal{R}})$ the set of all candidate strong arcs in $G_q^{\mathcal{R}}$. The set $\text{circ}(G_q^{\mathcal{R}})$ of candidate strong arcs in a cyclic d-path can be neither strong nor deleted. The maximal pair $(\mathcal{S}, \mathcal{D})$ of strong and, resp., deleted arcs for a d-graph is guaranteed to be unique and is calculated by the algorithm of Figure 1. We indicate with $G_q^{\mathcal{R}(\mathcal{S}, \mathcal{D})}$ the d-graph, called *optimized d-graph*, obtained from $G_q^{\mathcal{R}}$ by marking all arcs in \mathcal{D} as “deleted”, all arcs in \mathcal{S} as “strong”, and all remaining arcs as “weak”. Visually, we will remove all arcs labeled as “deleted”, as well as all white nodes with no incoming or outgoing arcs and all sources with no nodes.

Theorem 1. *The function $\text{calculateGFP}(G)$ shown in Figure 1 computes, in PTIME in the size of G , the maximal pair $(\mathcal{S}, \mathcal{D})$ of strong and, resp., deleted arcs with respect to d-graph G .*

The optimized d-graph only makes use of relevant relations, i.e., those relations that may contribute to determining obtainable answers, and thus gives us a procedure for deciding relevance of a relation in the context of CQs.

Theorem 2. *Let G be the optimized d-graph for a CQ q over a schema \mathcal{R} . A relation $r \in \mathcal{R}$ is relevant iff (i) r is nullary and occurs in q , or (ii) r occurs in G .*

```

                                unmarkDeleted( $S : \text{arc\_set}, \mathcal{D} : \text{arc\_set}, G : \text{d-graph}$ ) : arc_set
calculateGFP( $G : \text{d-graph}$ ) : arc_set  $\times$  arc_set   $\mathcal{D}' := \mathcal{D}$ 
   $S := \text{cand}(G) \setminus \text{circ}(G)$ 
   $\mathcal{D} := \text{arcs}(G) \setminus \text{cand}(G)$ 
  do {
    ( $S', \mathcal{D}'$ ) := ( $S, \mathcal{D}$ )
     $S := \text{unmarkStrong}(S', \mathcal{D}', G)$ 
     $\mathcal{D} := \text{unmarkDeleted}(S', \mathcal{D}', G)$ 
  } while ( $S, \mathcal{D}$ )  $\neq$  ( $S', \mathcal{D}'$ )
  return ( $S', \mathcal{D}'$ )

                                 $\mathcal{D}' := \mathcal{D}$ 
                                for each arc  $u \hat{\sim} v \in \mathcal{D}$ 
                                if ( $\text{black}(v)$ ) then
                                  bool strongExists := false
                                  for each arc  $u' \hat{\sim} v' \in \mathcal{S}$ 
                                    if ( $v = v'$ ) then strongExists := true ; break
                                  if (not strongExists) then  $\mathcal{D}' := \mathcal{D}' \setminus \{u \hat{\sim} v\}$ 
                                  else ( $v$  is white)
                                    if ( $\text{outArcs}(v, G) \setminus \mathcal{D} \neq \emptyset$ ) then  $\mathcal{D}' := \mathcal{D}' \setminus \{u \hat{\sim} v\}$ 
                                return  $\mathcal{D}'$ 

unmarkStrong( $S : \text{arc\_set}, \mathcal{D} : \text{arc\_set}, G : \text{d-graph}$ ) : arc set
   $S' := S$ 
  for each arc  $u \hat{\sim} v \in S$ 
    for each arc  $\gamma \in \text{outArcs}(v, G)$ 
      if ( $\gamma \notin S \cup \mathcal{D}$ )  $S' := S' \setminus \{u \hat{\sim} v\}$ ; break
  return  $S'$ 

```

Fig. 1. Algorithm determining the maximal sets of strong arcs and deleted arcs

4 Generating a minimal query plan

In this section, we first define what it means for a query plan to execute a minimal amount of accesses to answer a query. Then, we show how to generate, from an optimized d-graph, a query plan that is guaranteed to be minimal.

We indicate by $\text{Acc}(D, \Pi)$ the set of accesses made by a query plan Π on a database D . Ideally, one would like to have a query plan that makes as few accesses as possible. It turns out that such query plans do not always exist; however we can require query plans to be subset-minimal, as defined next.

Definition 1. Let Π and Π' be two query plans for a CQ q over a schema \mathcal{R} . We write $\Pi' \prec \Pi$ whenever, for every instance D , $\text{Acc}(D, \Pi') \subseteq \text{Acc}(D, \Pi)$ and there is an instance D' such that $\text{Acc}(D, \Pi') \subset \text{Acc}(D, \Pi)$. Query plan Π is subset-minimal iff $\Pi'' \prec \Pi$ for no query plan Π'' of q .

According to the above definition, a query plan Π is subset-minimal if there is no other query plan that is strictly better at making accesses than Π .

We now show how to construct a subset-minimal query plan for a CQ. We derive from the CQ's optimized d-graph a program expressed in Datalog notation. The Datalog program is then evaluated according to a strategy that closely corresponds to Datalog's usual least fixpoint semantics, with a few extra expedients that guarantee access minimality.

Whenever more than one relation is involved in the optimized d-graph, any access plan must necessarily access some relations before others. Deciding the order of the accesses to the different relations may be arbitrary, depending on the database instance, or mandatory, due to the access limitations on the schema. The first step in the generation of a query plan consists then in determining an ordering between groups of sources in the schema.

Let us denote by $\text{src}(u)$ the source corresponding to a node u in a d-graph. Then, for an optimized d-graph G , we establish an ordering among all the sources such that: (i) if

there is a weak arc $u \rightsquigarrow v$ in G , then $src(u) \preceq src(v)$; (ii) if there is a strong arc $u \rightsquigarrow v$ in G , then $src(u) \prec src(v)$; (iii) sources traversed by a cyclic d-path have the same order as the other sources in the cycle; all sources outside the cycle have a different order. We establish an ordering and assign a progressive number $pos(s)$ to each source s such that $pos(s_i) < pos(s_j)$ iff $s_i \prec s_j$.

The algorithm, that we cannot present here due to lack of space, produces a Datalog program based on the optimized d-graph as well as on the ordering. The algorithm rewrites the original query over “new versions” of the relations in the body: for every atom r in the body of the query, we introduce a new predicate with the same arity as r that acts as a sort of *cache* in which we store, during the query answering process, all the tuples extracted from r . Note that different occurrences of the same predicate give rise to different caches. Constants in the query are taken into account by replacing them with “fake” sources.

Finally, the program generated by our algorithm is completed by adding a fact for each artificial relation representing a constant in the query; the fact has the form $r_a(a)$, where r_a is the created relation and a is the removed constant. Intuitively, the generated program ensures that the input values for the input positions of a relation r are obtained from values retrieved from the appropriate relations and stored in the caches.

We now specify the execution strategy, called *fast-failing strategy*, for such a Datalog program that avoids redundant accesses. For each position i from 1 to k in the ordering, we first fully populate the i -caches, as described below, and then evaluate sat_i (a test that checks whether the subquery including only the j -caches, with $j < i$, is satisfiable). Finally we evaluate the query over the caches. In order to populate the i -caches, first the sat_{i-1} predicate is checked: if it fails, we exit and report the empty answer. Else, all their rules are evaluated (they may only refer to j -caches, with $j \leq i$) until a fixpoint is reached, i.e., until no new tuple is extracted for any of the i -caches. Since there may be several sources corresponding to the same relation, we have to make sure to not repeat any access to a relation. For this purpose, we keep track of all access tuples used to access relations; then, before accessing a relation for the evaluation of a cache rule, we check whether the access was already made. If so, we read the extraction from the corresponding cache; else we make the access proper. Note that accessing caches has no cost with respect to our notion of minimality. The fast-failing strategy is guaranteed to always calculate the same answer as the usual fixpoint semantics for the Datalog program, but the former will never repeat an access to a relation and will stop as soon as the answer is known to be empty, thus possibly avoiding further accesses made by the latter.

Theorem 3. *Let q be a query over a schema \mathcal{R} , and let G be the corresponding optimized d-graph. Then, the execution with the fast-failing strategy of the Datalog program constructed from q by our algorithm, based on G , implements a (sound and complete) query plan Π for q ; Π is subset-minimal.*

5 Extension to more expressive query languages

We extend the technique presented in Section 4 to union of conjunctive queries (UCQs) by transforming a UCQ into an appropriate CQ (the two being not equivalent), so that our optimizing technique can be applied on the latter; the output is then adjusted so

as to obtain a query plan for the original UCQ (assumed, w.l.o.g., to be constant-free). This is done as follows. First, we standardize apart all queries in the UCQ q ; then we compose a CQ q' containing the conjunction of all bodies of all CQs in the UCQ q . The obtained CQ q' is processed with the algorithm to produce a Datalog program. The rule corresponding to q' in the program is split in as many rules as there are CQs in q , each with the appropriate bodies. The resulting Datalog program, with minor adjustments to the fast-failing strategy, implements a subset-minimal query plan for q . Relevance can be easily decided from the optimized d -graph of q' in the same way as was done for CQs.

We consider now the class of CQ^\neg queries (CQs with safe negation) and show how to check relevance of relations. Whenever q is a CQ^\neg query, we indicate by q^+ (resp., q^-) the query whose head is the same as q 's and whose body is the conjunction of the negative (resp., positive) literals in q .

Theorem 4. *Let q be a CQ^\neg query over a schema \mathcal{R} and let G be the optimized d -graph for q^+ . A relation $r \in \mathcal{R}$ is relevant iff q is satisfiable, and (i) r occurs in q^- , or (ii) r occurs in G , or (iii) r is nullary and occurs in q .*

Since unsatisfiability for CQ^\neg queries can be checked in PTIME and the optimized d -graph of a CQ is also computed in PTIME, we have the following.

Corollary 1. *Relevance for CQ^\neg queries is decidable in PTIME.*

The argument easily extends to UCQ^\neg queries and a subset-minimal query plan can be obtained similarly to what was done for UCQs.

Unlike UCQ^\neg queries, determining relevance for Datalog queries is impossible, since, if we were able to decide relevance of a relation, we could also decide containment between Datalog queries, which is known to be undecidable. This solves negatively the issue opened in [6].

Theorem 5. *There cannot be an algorithm that determines whether a relation is relevant for a Datalog query.*

6 Experimental evaluation

We now show experimental results to validate our optimisation technique. We have built a prototype system with a query plan optimiser; the relational sources are local, and we measure the number of accesses. The schema we adopted is reported below: pub_1 and pub_2 store published papers and their authors; $conf$ stores information about papers published in conferences, with the year of publication; rev stores data about reviewers of conferences in certain years; sub stores information about papers submitted by persons; rev_vldb stores information about reviewers of VLDB papers, with the associated evaluation.

$$\begin{aligned} &pub_1^{io}(Paper, Person), \\ &pub_2^{oo}(Paper, Person), \\ &conf^{ooo}(Paper, ConfName, Year), \\ &rev^{ooi}(Person, ConfName, Year), \\ &sub^{oi}(Paper, Person), \\ &rev_vldb^{iio}(Person, Paper, Eval) \end{aligned}$$

We populated the above schema with synthetic data, automatically generated. We utilised constants from abstract domains, where every domain has 400 values – this allowed the sources to have some values in common, as it would happen in real-world Web information systems, so as to make the recursive extraction possible; finally, we populated every source with 1000 tuples.

We considered the following queries:

1. q_1 , asking for authors of publications in conferences where they were also reviewers: $q_1(R) \leftarrow pub_1(P, R), conf(P, C, Y), rev(R, C, Y)$
2. q_2 , asking for papers rejected at VLDB by a reviewer and accepted at an event listing the same reviewer: $q_2(R) \leftarrow rev_vldb(R, S, rej), conf(S, C, Y), rev(R, C, Y)$
3. q_3 , asking for reviewers of VLDB 2007 who have accepted at VLDB a submission authored by a VLDB coauthor: $q_3(R) \leftarrow rev_vldb(R, S, acc), sub(S, A), pub_1(P, R), pub_1(P, A), rev(R, vldb, 2007), conf(P, vldb, Y)$

In Figure 2 we report, for every query and for every source relation, the number of accesses and the number of extracted tuples, first for the naive query plan (that of [2], that makes all possible accesses with the constants retrieved so far), and then for the optimised one; when a source is not included in the plan because it is not relevant, we leave the corresponding cells blank in the table. The improvement in the efficiency due to our optimisation has proved to be significant – in the optimised plans many sources are not accessed at all. Notice that it is not surprising that sometimes we get the same values for the same source for different queries, since such values heavily depend on the constants that the sources have in common, that do not depend on the query. Needless to say, the optimised plans return the same answers as the non-optimised ones.

Finally, the optimisation led to significant pruning of the queries’ d-graphs. In Figure 3 we show the d-graph for q_1 before and after the pruning. Attribute nodes in the same source relations are grouped in an oval; they appear in the same order (top to bottom) as the arguments in the schema (left to right). The relation name is written below the source, with the occurrence number superscripted in brackets. Strong arcs are represented by double-lined arrows, and weak arcs by single-lined arrows.

Acknowledgments. This research was supported by Italian PRIN project NGS (New Generation Search engine) and EU FET Project IST-2005-7603 TONES (Thinking ONtologies). The authors would like to thank Diego Calvanese for useful discussion.

q_1					q_2					q_3				
relation	accesses		returned rows		relation	accesses		returned rows		relation	accesses		returned rows	
	naive	min.	naive	min.		naive	min.	naive	min.		naive	min.	naive	min.
pub_1	4		996		pub_1	4		996		pub_1	4		996	
pub_2	399	364	991	884	pub_2	399		991		pub_2	399	364	991	884
$conf$	4	1	1000	1000	$conf$	4	1	1000	1000	$conf$	4	1	1000	1000
rev	20	20	999	999	rev	20	20	999	999	rev	20	1	999	56
sub	400		996		sub	400		996		sub	400	357	996	893
rev_vldb	159,600		997		rev_vldb	159,600	133,588	997	818	rev_vldb	159,600	17,184	997	102

Fig. 2. Experimental results for the test queries

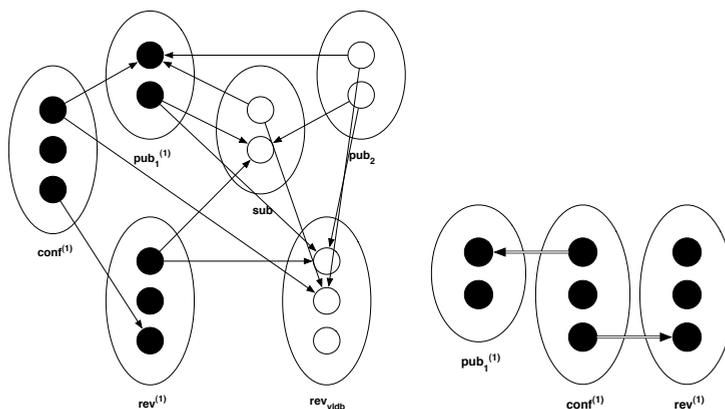


Fig. 3. Non-optimised and optimised d-graphs for query q_1

References

1. Andrea Cali and Diego Calvanese. Optimized querying of integrated data over the Web. In *Proc. of the IFIP WG8.1 Working Conference on Engineering Information Systems in the Internet Context (EISIC 2002)*, pages 285–301. Kluwer Academic Publishers, 2002.
2. Oliver M. Duschka and Alon Y. Levy. Recursive plans for information gathering. In *Proc. of IJCAI'97*, pages 778–784, 1997.
3. Daniela Florescu, Alon Levy, and Alberto Mendelzon. Database techniques for the World-Wide Web: A survey. *SIGMOD Record*, 27(3):59–74, 1998.
4. Daniela Florescu, Alon Y. Levy, Ioana Manolescu, and Dan Suciu. Query optimization in the presence of limited access patterns. In *Proc. of ACM SIGMOD*, pages 311–322, 1999.
5. Chen Li and Edward Chang. Query planning with limited source capabilities. In *Proc. of ICDE 2000*, pages 401–412, 2000.
6. Chen Li and Edward Chang. Answering queries with useful bindings. *ACM Trans. on Database Systems*, 26(3):313–343, 2001.
7. Chen Li and Edward Chang. On answering queries in the presence of limited access patterns. In *Proc. of ICDT 2001*, pages 219–233, 2001.
8. Todd D. Millstein, Alon Y. Levy, and Marc Friedman. Query containment for data integration systems. In *Proc. of PODS 2000*, pages 67–75, 2000.
9. Anand Rajaraman, Yehoshua Sagiv, and Jeffrey D. Ullman. Answering queries using templates with binding patterns. In *Proc. of PODS'95*, 1995.

Ontology-based Database Access

Diego Calvanese¹, Giuseppe De Giacomo², Domenico Lembo²,
Maurizio Lenzerini², Antonella Poggi², Riccardo Rosati²

¹Faculty of Computer Science
Free University of Bozen-Bolzano
Piazza Domenicani 3
I-39100 Bolzano, Italy

²Dip. di Informatica e Sistemistica
Università di Roma “La Sapienza”
Via Salaria 113
I-00198 Roma, Italy

¹calvanese@inf.unibz.it, ²{lastname}@dis.uniroma1.it

Abstract. One of the most interesting usages of shared conceptualizations is *ontology-based data access*. That is, to the usual data layer of an information system we superimpose a conceptual layer to be exported to the client. Such a layer allows the client to have a conceptual view of the information in the system, which abstracts away from how such information is maintained in the data layer of the system itself. While ontologies are the best candidates for realizing the conceptual layer, relational DBMSs are natural candidates for the management of the data layer. The need of efficiently processing large amounts of data requires ontologies to be expressed in a suitable fragment of OWL: the fragment should allow, on the one hand, for modeling the kind of intensional knowledge needed in real-world applications, and, on the other hand, for delegating to a relational DBMS the part of reasoning (in particular query answering) that deals with the data. In this paper, we propose one such a fragment, in fact the largest fragment currently known to satisfy the above requirements. Furthermore, we provide means to access databases that are independent from the ontology, by proposing a novel mapping language that solves the so-called “impedance mismatch” between values in the databases and objects represented in the ontology.

1 Introduction

In several areas, such as Enterprise Application Integration, Data Integration [9], and the Semantic Web [6], clients need to access the services exported by the system, and hence require a representation of the intensional level of the application domain in terms of which they can specify the access to the exported services. One of the most interesting usages of such a shared conceptualization is *ontology-based data access*, where a conceptual layer is exported to the client, abstracting away from how actual data is maintained. While ontologies are the best candidates for realizing the conceptual layer, relational DBMSs are natural candidates for the management of the data layer, since relational database technology is nowadays the best technology for efficient management of very large quantities of data.

Recently, basic research has shown that none of the variants of OWL is suitable to act as the formalism for representing ontologies in this context [4, 11, 8], if not re-

stricted (they all are coNP-hard w.r.t. data complexity). Possible restrictions that guarantee polynomial reasoning (at least, if we concentrate on instance checking only) have been looked at, such as Horn-*SHIQ* [8], \mathcal{EL}^{++} [2], DLP [5]. Among such fragments, of particular interest are those belonging to the DL-Lite family [3, 4]. These logics allow for answering complex queries (namely, conjunctive queries, i.e., SQL select-project-join queries, and unions of conjunctive queries) in LOGSPACE w.r.t. data complexity (i.e., the complexity measured only w.r.t. the size of the data). More importantly, they allow for delegating query processing, after a preprocessing phase which is independent of the data, to the relational DBMS managing the data layer.

In this paper, we propose to use a new DL, called $DL\text{-Lite}_{\mathcal{A}}^+$, which keeps the above mentioned features of the other languages in the *DL-Lite* family, while allowing to distinguish between objects and values, by introducing, besides concepts and roles, also concept-attributes and role-attributes, that describe properties of concepts (resp., roles) represented by values rather than objects. Then, we look at the problem of accessing databases that are independent from the ontology. Observe, however, that such databases, being relational, store only values, not objects. Hence, to deal with the so-called “impedance mismatch”, we propose to relate database values to the ontology by using a novel mapping language [9], such that objects are constructed from such values.

2 The description logic $DL\text{-Lite}_{\mathcal{A}}^+$

In this section we present a new logic of the *DL-Lite* family, called $DL\text{-Lite}_{\mathcal{A}}^+$. As usual in DLs, all logics of the *DL-Lite* family allow one to represent the universe of discourse in terms of concepts, denoting sets of objects, and roles, denoting binary relations between objects. In addition, the DLs discussed in this paper allow one to use (i) value-domains, a.k.a. concrete domains [10], denoting sets of (data) values, (ii) concept attributes, denoting binary relations between objects and values, and (iii) role attributes, denoting binary relations between pairs of objects and values. Obviously, a role attribute can be also seen as a ternary relation relating two objects and a value.

We first introduce the DL $DL\text{-Lite}_{\mathcal{FR}}$, that combines the main features of two DLs presented in [4], called $DL\text{-Lite}_{\mathcal{F}}$ and $DL\text{-Lite}_{\mathcal{R}}$, respectively, and forms the basics of $DL\text{-Lite}_{\mathcal{A}}^+$. The value-domains that we consider in $DL\text{-Lite}_{\mathcal{FR}}$ are those corresponding to the data types adopted by the Resource Description Framework (RDF)¹. Intuitively, these types represent sets of values that are pairwise disjoint. In the following, we denote such value-domains by T_1, \dots, T_n . Furthermore, we denote with Γ the alphabet for constants, which we assume partitioned into two sets, namely, Γ_V (the set of constant symbols for values), and Γ_O (the set of constant symbols for objects). In turn, Γ_V is partitioned into n sets $\Gamma_{V_1}, \dots, \Gamma_{V_n}$, where each Γ_{V_i} is the set of constants for the values in the value-domain T_i .

In providing the specification of our logics, we use the following notation: A denotes an *atomic concept*, B a *basic concept*, C a *general concept*, and \top_C the *universal concept*; E denotes a basic value-domain, i.e., the range of an attribute, F denotes a *general value-domain*, and \top_D the *universal value-domain*; P denotes an *atomic role*, Q a *basic role*, and R a *general role*; U_C denotes an *atomic attribute*, and V_C a *general attribute*; U_R denotes an *atomic role attribute*, and V_R a *general role attribute*.

¹ <http://www.w3.org/RDF/>

Given a concept attribute U_C (resp. a role attribute U_R), we call *domain* of U_C (resp. U_R), denoted by $\delta(U_C)$ (resp. $\delta(U_R)$), the set of objects (resp. of pairs of objects) that U_C (resp. U_R) relates to values, and we call *range* of U_C (resp. U_R), denoted by $\rho(U_C)$ (resp. $\rho(U_R)$), the set of values that U_C (resp. U_R) relates to objects (resp. pairs of objects). Notice that the domain $\delta(U_C)$ of a concept attribute U_C is a concept, whereas the domain $\delta(U_R)$ of a role attribute U_R is a role.

In particular, $DL\text{-Lite}_{\mathcal{FR}}$ expressions are defined as follows.

- Basic and general concept expressions:

$$B ::= A \mid \exists Q \mid \delta(U_C), \quad C ::= \top_C \mid B \mid \neg B$$

- Basic and general value-domain expressions:

$$E ::= \rho(U_C) \mid \rho(U_R), \quad F ::= \top_D \mid T_1 \mid \cdots \mid T_n$$

- General concept and role attribute expressions:

$$V_C ::= U_C \mid \neg U_C, \quad V_R ::= U_R \mid \neg U_R$$

- Basic and general role expressions:

$$Q ::= P \mid P^- \mid \delta(U_R) \mid \delta(U_R)^-, \quad R ::= Q \mid \neg Q$$

A $DL\text{-Lite}_{\mathcal{FR}}$ knowledge base (KB) $\mathcal{K} = \langle \mathcal{T}, \mathcal{A} \rangle$ is constituted by two components: a TBox \mathcal{T} , used to represent intensional knowledge, and an ABox \mathcal{A} , used to represent extensional knowledge. $DL\text{-Lite}_{\mathcal{FR}}$ TBox assertions are of the form:

$B \sqsubseteq C$	<i>concept inclusion</i>	(funct P)	<i>role functionality</i>
$Q \sqsubseteq R$	<i>role inclusion</i>	(funct P^-)	<i>inverse role functionality</i>
$E \sqsubseteq F$	<i>value-domain inclusion</i>	(funct U_C)	<i>concept attribute functionality</i>
$U_C \sqsubseteq V_C$	<i>concept attribute inclusion</i>	(funct U_R)	<i>role attribute functionality</i>
$U_R \sqsubseteq V_R$	<i>role attribute inclusion</i>		

A concept inclusion assertion expresses that a (basic) concept B is subsumed by a (general) concept C . Analogously for the other types of inclusion assertions. A role functionality assertion expresses the (global) functionality of an atomic role. Analogously for the other types of functionality assertions.

A $DL\text{-Lite}_{\mathcal{FR}}$ ABox is a finite set of assertions of the form:

$$A(a), \quad P(a, b), \quad U_C(a, b), \quad U_R(a, b, c)$$

where a, b and c are constants in the alphabet Γ .

Following the classical approach in DLs, the semantics of $DL\text{-Lite}_{\mathcal{FR}}$ is given in terms of first-order logic interpretations. All such interpretations agree on the semantics assigned to each value-domain T_i and to each constant in Γ_V . In particular, each T_i is interpreted as the set $val(T_i)$ of values of the corresponding RDF data type, and each $c_i \in \Gamma_V$ is interpreted as one specific value, denoted $val(c_i)$, in $val(T_i)$. Note that, for $i \neq j$, it holds that $val(T_i) \cap val(T_j) = \emptyset$.

Based on the above observations, we can now define the notion of interpretation in $DL\text{-Lite}_{\mathcal{FR}}$. An *interpretation* is a pair $I = (\Delta^I, \cdot^I)$, where

- $\Delta^{\mathcal{I}}$ is the interpretation domain, that is the disjoint union of two sets: $\Delta_O^{\mathcal{I}}$, called the *domain of objects*, and $\Delta_V^{\mathcal{I}}$, called the *domain of values*. In turn, $\Delta_V^{\mathcal{I}}$ is the union of $val(T_1), \dots, val(T_n)$.
- \mathcal{I} is the *interpretation function*, i.e., a function that assigns an element of $\Delta^{\mathcal{I}}$ to each constant in Γ ,
 - for each $a \in \Gamma_V$, $a^{\mathcal{I}} = val(a)$,
 - for each $a \in \Gamma_O$, $a^{\mathcal{I}} \in \Delta_O^{\mathcal{I}}$,
 - for each $a, b \in \Gamma$, $a \neq b$ implies $a^{\mathcal{I}} \neq b^{\mathcal{I}}$,
 - for each T_i , $T_i^{\mathcal{I}} = val(T_i)$,
 - the following conditions are satisfied:

$$\begin{array}{ll}
 \top_C^{\mathcal{I}} = \Delta_O^{\mathcal{I}} & (P^-)^{\mathcal{I}} = \{ (o, o') \mid (o', o) \in P^{\mathcal{I}} \} \\
 \top_D^{\mathcal{I}} = \Delta_V^{\mathcal{I}} & (\rho(U_C))^{\mathcal{I}} = \{ v \mid \exists o. (o, v) \in U_C^{\mathcal{I}} \} \\
 A^{\mathcal{I}} \subseteq \Delta_O^{\mathcal{I}} & (\rho(U_R))^{\mathcal{I}} = \{ v \mid \exists o, o'. (o, o', v) \in U_R^{\mathcal{I}} \} \\
 P^{\mathcal{I}} \subseteq \Delta_O^{\mathcal{I}} \times \Delta_O^{\mathcal{I}} & (\delta(U_C))^{\mathcal{I}} = \{ o \mid \exists v. (o, v) \in U_C^{\mathcal{I}} \} \\
 U_C^{\mathcal{I}} \subseteq \Delta_O^{\mathcal{I}} \times \Delta_V^{\mathcal{I}} & (\delta(U_R))^{\mathcal{I}} = \{ (o, o') \mid \exists v. (o, o', v) \in U_R^{\mathcal{I}} \} \\
 U_R^{\mathcal{I}} \subseteq \Delta_O^{\mathcal{I}} \times \Delta_O^{\mathcal{I}} \times \Delta_V^{\mathcal{I}} & (\exists Q)^{\mathcal{I}} = \{ o \mid \exists o'. (o, o') \in Q^{\mathcal{I}} \} \\
 (\neg U_C)^{\mathcal{I}} = (\Delta_O^{\mathcal{I}} \times \Delta_V^{\mathcal{I}}) \setminus U_C^{\mathcal{I}} & (\neg Q)^{\mathcal{I}} = (\Delta_O^{\mathcal{I}} \times \Delta_O^{\mathcal{I}}) \setminus Q^{\mathcal{I}} \\
 (\neg U_R)^{\mathcal{I}} = (\Delta_O^{\mathcal{I}} \times \Delta_O^{\mathcal{I}} \times \Delta_V^{\mathcal{I}}) \setminus U_R^{\mathcal{I}} & (\neg B)^{\mathcal{I}} = \Delta_O^{\mathcal{I}} \setminus B^{\mathcal{I}}
 \end{array}$$

Note that the above definition implies that different constants are interpreted differently in the domain, i.e., *DL-Lite_{FR}* adopts the so-called unique name assumption.

We define when an interpretation \mathcal{I} satisfies an assertion (i.e., is a model of it) as follows (below, each o , possibly with subscript, is an element of $\Delta^{\mathcal{I}}$, and a, b and c are constants in Γ). Specifically, an interpretation \mathcal{I} *satisfies* (i) an inclusion assertion $\alpha \sqsubseteq \beta$, if $\alpha^{\mathcal{I}} \subseteq \beta^{\mathcal{I}}$; (ii) a functional assertion (funct γ), where γ is either P, P^- , or U_C , if, for each o_1, o_2, o_3 , $(o_1, o_2) \in \gamma^{\mathcal{I}}$ and $(o_1, o_3) \in \gamma^{\mathcal{I}}$ implies $o_2 = o_3$; (iii) a functional assertion (funct U_R), if for each o_1, o_2, o_3, o_4 , $(o_1, o_2, o_3) \in U_R^{\mathcal{I}}$ and $(o_1, o_2, o_4) \in U_R^{\mathcal{I}}$ implies $o_3 = o_4$; (iv) a membership assertion $\alpha(a)$, where α is either A or D , if $a^{\mathcal{I}} \in \alpha^{\mathcal{I}}$; (v) a membership assertion $\beta(a, b)$, where β is either P or U_C , if $(a^{\mathcal{I}}, b^{\mathcal{I}}) \in \beta^{\mathcal{I}}$; (vi) a membership assertion $U_R(a, b, c)$, if $(a^{\mathcal{I}}, b^{\mathcal{I}}, c^{\mathcal{I}}) \in U_R^{\mathcal{I}}$. A *model of a KB* \mathcal{K} is an interpretation \mathcal{I} that is a model of all assertions in \mathcal{K} . A KB is *satisfiable* if it has at least one model.

A conjunctive query (CQ) q over a knowledge base \mathcal{K} is an expression of the form $q(\mathbf{x}) \leftarrow \exists \mathbf{y}. conj(\mathbf{x}, \mathbf{y})$, where \mathbf{x} are the so-called *distinguished variables*, \mathbf{y} are existentially quantified variables called the *non-distinguished variables*, and $conj(\mathbf{x}, \mathbf{y})$ is a conjunction of atoms of the form $A(x), D(x), P(x, y), U_C(x, y), U_R(x, y, z)$, or $x = y$, where x, y, z are either variables in \mathbf{x} or in \mathbf{y} or constants in Γ . A *union of conjunctive queries* (UCQ) is a query of the form $q(\mathbf{x}) \leftarrow \bigvee_i \exists \mathbf{y}_i. conj(\mathbf{x}, \mathbf{y}_i)$. A query $q(\mathbf{x}) \leftarrow \varphi(\mathbf{x})$ is interpreted in \mathcal{I} as the set $q^{\mathcal{I}}$ of tuples $\mathbf{o} \in \Delta^{\mathcal{I}} \times \dots \times \Delta^{\mathcal{I}}$ such that, when we assign \mathbf{o} to the variables \mathbf{x} , the formula $\varphi(\mathbf{x})$ evaluates to true in \mathcal{I} .

The reasoning service we are interested in is *query answering*: given a knowledge base \mathcal{K} and a UCQ $q(\mathbf{x})$ over \mathcal{K} , return the *certain answers* to $q(\mathbf{x})$ over \mathcal{K} , i.e., all tuples \mathbf{t} of elements of Γ such that for every model \mathcal{I} of \mathcal{K} .

From the results in [4] it follows that, in general, query answering over *DL-Lite_{FR}* KBs is PTIME-hard in data complexity (i.e., the complexity measured w.r.t. the size of the ABox only). As a consequence, to solve query answering over *DL-Lite_{FR}* KBs, we

need at least the power of general recursive Datalog. Since we are interested in DLs where query answering can be done in LOGSPACE, we introduce the DL $DL-Lite_{\mathcal{A}}^+$, which is obtained from $DL-Lite_{\mathcal{FR}}$ by imposing a limitation on the use of the functionality assertions in the TBox, as described next.

Definition 1. A $DL-Lite_{\mathcal{A}}^+$ knowledge base is pair $\langle \mathcal{T}, \mathcal{A} \rangle$, where \mathcal{A} is a $DL-Lite_{\mathcal{FR}}$ ABox, and \mathcal{T} is a $DL-Lite_{\mathcal{FR}}$ TBox satisfying the following conditions:

1. for every role inclusion assertion $Q \sqsubseteq R$ in \mathcal{T} , where R is an atomic role or the inverse of an atomic role, the assertions $(\text{funct } R)$ and $(\text{funct } R^-)$ are not in \mathcal{T} ;
2. for every concept attribute inclusion assertion $U_C \sqsubseteq V_C$ in \mathcal{T} , where V_C is an atomic concept attribute, the assertion $(\text{funct } V_C)$ is not in \mathcal{T} ;
3. for every role attribute inclusion assertion $U_R \sqsubseteq V_R$ in \mathcal{T} , where V_R is an atomic role attribute, the assertion $(\text{funct } V_R)$ is not in \mathcal{T} .

Roughly speaking, a $DL-Lite_{\mathcal{A}}^+$ TBox imposes the condition that *every functional role cannot be specialized* by using it in the right-hand side of role inclusion assertions; the same condition is also imposed on every functional (role or concept) attribute.

In fact, it turns out that the above restriction is necessary in order to perform query answering over a $DL-Lite_{\mathcal{A}}^+$ ontology by following a technique similar to the one developed for the other logics in the $DL-Lite$ family [3]. In particular, it can be shown [12] that query answering can be reduced to the evaluation of a first-order query over a relational database representing the ontology ABox. Such query is obtained by reformulating the original query based on the TBox assertions. Notably, this reformulation does not depend on the data, and hence query answering is LOGSPACE in data complexity.

3 Linking data to $DL-Lite_{\mathcal{A}}^+$ ontologies

Most work on DLs do not deal with the problem of how to store ABox assertions, nor do they address the issue of how to acquire ABox assertions from existing data sources. It is our opinion that this topic is of special importance in several contexts where the use of ontologies is advocated, especially in the case where the ontology is used to provide a unified conceptual model of an organization (e.g., in Enterprise Application Integration). In these contexts, the problem can be described as follows: the ontology is a virtual representation of a universe of discourse, and the instances of concepts and roles in the ontology are simply an abstract representation of some real data stored in existing data sources. Therefore, the problem arises of establishing sound mechanisms for linking existing data to the instances of the concepts and the roles in the ontology.

In this section we sketch our solution, by presenting a mapping mechanism that enables a designer to link data sources to an ontology expressed in $DL-Lite_{\mathcal{A}}^+$. Before delving into the details of the method, a preliminary discussion on the notorious impedance mismatch problem between data and objects is in order. When mapping data sources to ontologies, one should take into account that sources store data, whereas instances of concepts are objects, where each object should be denoted by an ad hoc identifier (e.g., a constant in logic), not to be confused with any data item. In $DL-Lite_{\mathcal{A}}^+$, we address this problem by keeping data value constants separate from object identifiers, and by accepting that object identifiers be created using data values, in particular as (logic)

terms over data items. Note that this idea traces back to the work done in deductive object-oriented databases [7].

To realize this idea, we modify the set Γ_O as follows. We assume that data appearing at the sources are denoted by constants in Γ_V , and we introduce a new alphabet Λ of function symbols in $DL-Lite^+_{\mathcal{A}}$, where each function symbol has an associated arity, specifying the number of argument it accepts. On the basis of Γ_V and Λ , we inductively define the set $\tau(\Lambda, \Gamma_V)$ of all *terms* of the form $f(d_1, \dots, d_n)$ such that (i) $f \in \Lambda$, (ii) the arity of f is $n > 0$, and (iii) $d_1, \dots, d_n \in \Gamma_V$. We finally sanction that the set Γ_O of symbols used in $DL-Lite^+_{\mathcal{A}}$ for denoting objects actually coincides with $\tau(\Lambda, \Gamma_V)$. In other words, we use the terms built out of Γ_V using the function symbols in Λ for denoting the instances of concepts in $DL-Lite^+_{\mathcal{A}}$ ontologies.

All the notions defined for our logics remain unchanged. In particular, an interpretation $I = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ still assigns a different element of $\Delta^{\mathcal{I}}$ to every element of Γ , which means that different terms in $\tau(\Lambda, \Gamma_V)$ are interpreted as different objects in $\Delta^{\mathcal{I}}$, i.e., we enforce the unique name assumption on terms.

Let us now turn our attention to the problem of linking data in the sources to objects in the ontology. To this aim, as we said before, we assume that all value constants stored in DB belong to Γ_V , and that the data sources are wrapped into a relational database DB (constituted by the relational schema, and the extensions of the relations), so that we can query such data by using SQL. Then, we adapt principles and techniques from the literature on data integration [9]. In particular, we use the notion of *mapping*, which we now introduce by means of an example.

Example 1. Consider a $DL-Lite^+_{\mathcal{A}}$ TBox in which *person* is a concept name, *age* and *cityName* are concept attributes names, *CITY-OF-BIRTH* is a role name, and a relational database contains the ternary relation symbols *S1* and *S2* and the unary relation symbol *S3*. We want to model the situation where every tuple $(n, s, a) \in S1$ corresponds to a person whose name is n , whose surname is s , and whose age is a , and we want to denote such a person with $p(n, s)$. Note that this implies that we know that there are no two persons in our application that have the same pair (n, s) stored in *S1*. Similarly, we want to model the fact that every tuple $(n, s, cb) \in S2$ corresponds to a person whose name is n , whose surname is s , and whose city of birth is cb . Finally, we know that source *S3* directly stores object constants denoting instances of person. The following is the set of mapping assertions modeling the above situation.

$$\begin{aligned} S1(n, s, a) &\rightsquigarrow \text{person}(p(n, s)), \text{age}(p(n, s), a) \\ S2(n, s, cb) &\rightsquigarrow \text{CITY-OF-BIRTH}(p(n, s), ct(cb)), \text{cityName}(ct(cb), cb) \\ S3(q) &\rightsquigarrow \text{person}(q). \end{aligned}$$

Above, n, s, a, cb and q are variable symbols, p and ct are function symbols, whereas $p(n, s)$ and $ct(n)$ are so-called variable terms (see below). ■

The example shows that, in specifying mapping assertions, we need variable terms, i.e., terms containing variables. Indeed, we extend terms to *variable terms* of the form $f(z)$, where f is a function symbol in Λ of arity m , and z denotes an m -tuple of variables.

We can now provide the definition of mapping assertions. Through a mapping we associate a conjunctive query over atomic concepts, domains, roles, attributes, and role

attributes (generically referred to as *predicates* in the following) with a first-order (more precisely, SQL) query of the appropriate arity over the database. The intuition is that, by evaluating such a query, we retrieve the facts that constitute the ABox assertions for the predicates appearing in the conjunctive query. Formally, a *mapping assertion* is an assertion of the form: $\varphi \rightsquigarrow \psi$, where φ is an arbitrary SQL query of arity $n > 0$ over DB , and ψ is a UCQ over \mathcal{T} of arity $n' > 0$ without non-distinguished variables, that possibly involves variable terms.

We now describe the semantics of mapping assertions. To this end, we introduce the notion of ground instance of a formula. Let $\gamma(\mathbf{x})$ be a formula over a $DL\text{-}Lite_{\mathcal{A}}^+$ TBox with n distinguished variables \mathbf{x} , and let \mathbf{v} a tuple of value constants of arity n . Then the ground instance $\gamma[\mathbf{x}/\mathbf{v}]$ of $\gamma(\mathbf{x})$ is the formula obtained by substituting every occurrence of x_i with v_i (for $i \in \{1, \dots, n\}$) in $\psi(\mathbf{x})$. Let M be a mapping assertion of the form $\varphi(\mathbf{x}) \rightsquigarrow \psi(\mathbf{t}, \mathbf{y})$, where \mathbf{x} and \mathbf{y} are variables, $\mathbf{y} \subseteq \mathbf{x}$ and \mathbf{t} are variable terms of the form $f(\mathbf{z})$, $f \in \mathcal{A}$ and $\mathbf{z} \subseteq \mathbf{x}$. We say that I satisfies M with respect to a database DB , if for every tuple of values \mathbf{v} such that $\mathbf{v} \in \text{ans}(\varphi, DB)$, and for each ground atom X in $\psi[\mathbf{x}/\mathbf{v}]$, we have that: (i) if X has the form $\alpha(s)$, where α is either A or D , then $s^{\mathcal{I}} \in \alpha^{\mathcal{I}}$; (ii) if X has the form $\beta(s_1, s_2)$, where β is either P or U_C , then $(s_1^{\mathcal{I}}, s_2^{\mathcal{I}}) \in \beta^{\mathcal{I}}$; (iii) if X has the form $U_R(s_1, s_2, s_3)$, then $(s_1^{\mathcal{I}}, s_2^{\mathcal{I}}, s_3^{\mathcal{I}}) \in U_R^{\mathcal{I}}$.

Finally, we can summarize the semantics of a $DL\text{-}Lite_{\mathcal{A}}^+$ ontology with mapping assertions, denoted with $\langle \mathcal{T}, \mathcal{M}, DB \rangle$, where DB is a database as defined above, \mathcal{T} is a $DL\text{-}Lite_{\mathcal{A}}^+$ TBox, and \mathcal{M} a set of mapping assertions between DB and \mathcal{T} . An interpretation $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ is a *model* of if \mathcal{I} is a model of \mathcal{T} and satisfies all mapping assertions in \mathcal{M} wrt DB . The notion of certain answer to queries posed to $\langle \mathcal{T}, \mathcal{M}, DB \rangle$ remains the same as the one described in the previous section.

We now briefly sketch the technique for query answering over a $DL\text{-}Lite_{\mathcal{A}}^+$ ontology with mappings. First, we split each mapping assertion $\varphi \rightsquigarrow \psi$ into several assertions of the form $\varphi \rightsquigarrow p$, one for each atom p in ψ . Then, we unify in all possible ways the atoms in the query q to be evaluated with the right-hand side atoms of the (split) mappings, thus obtaining a (bigger) union of conjunctive queries containing variable object terms. Then, we unfold each atom with the corresponding left-hand side mapping query. Observe that, after unfolding, we obtain an SQL query that can be evaluated over DB , and possibly returns terms built from values extracted from DB .

Example 2. Refer to the previous example, and consider now the following query over the TBox, asking for the age of those people that are born in Rome:

$$q(z) \leftarrow \exists x, y. \text{person}(x), \text{CITY-OF-BIRTH}(x, y), \text{cityName}(y, \text{Roma}), \text{age}(x, z)$$

Let us, for simplicity, assume that no reasoning on the TBox has to be done in order to answer the query q , and hence let us directly evaluate such a query by exploiting the mapping, without materializing the ABox of the KB.

We first split the mapping (left as an exercise), and then unify the atoms in the query with the right-hand side atoms in the split mapping, thus obtaining

$$q(z) \leftarrow \text{person}(p(n, s)), \text{CITY-OF-BIRTH}(p(n, s), \text{ct}(\text{Roma})), \\ \text{cityName}(\text{ct}(\text{Roma}), \text{Roma}), \text{age}(p(n, s), z).$$

Then, we unfold each atom with the corresponding left-hand side of the mapping query, and obtain the query: $q(z) \leftarrow S2(n, s, \text{Roma}), SI(n, s, z)$, which can be simply evaluated over the database to get the certain answers to q . ■

4 Conclusions

We argue that, for ontology-based data access, ontologies need to be expressed in a fragment of OWL that is LOGSPACE in data complexity, and that allows for delegating to the relational DBMS managing the data layer the part of reasoning that deals with the data. We have proposed one such fragment, $DL\text{-Lite}_A^+$, which is in fact the biggest fragment currently known to satisfy the above requirements. In this paper we have looked at binary roles only, but all the results presented here can be extended to relations of arbitrary arity. All features introduced in this paper, have been implemented in the QuOnto system² [1] (originally based on a DL, called $DL\text{-Lite}_F$, that is a subset of $DL\text{-Lite}_A^+$).

Acknowledgments. This research has been partially supported by FET project TONES (Thinking ONtologiES), funded by the EU under contract number FP6-7603, by project HYPER, funded by IBM through a Shared University Research (SUR) Award grant, and by MIUR FIRB 2005 project “Tecnologie Orientate alla Conoscenza per Aggregazioni di Imprese in Internet” (TOCAI.IT).

References

1. A. Acciari, D. Calvanese, G. De Giacomo, D. Lembo, M. Lenzerini, M. Palmieri, and R. Rosati. QUONTO: Querying ONTOLOGIES. In *Proc. of AAAI 2005*, pages 1670–1671, 2005.
2. F. Baader, S. Brandt, and C. Lutz. Pushing the \mathcal{EL} envelope. In *Proc. of IJCAI 2005*, pages 364–369, 2005.
3. D. Calvanese, G. De Giacomo, D. Lembo, M. Lenzerini, and R. Rosati. DL-Lite: Tractable description logics for ontologies. In *Proc. of AAAI 2005*, pages 602–607, 2005.
4. D. Calvanese, G. De Giacomo, D. Lembo, M. Lenzerini, and R. Rosati. Data complexity of query answering in description logics. In *Proc. of KR 2006*, pages 260–270, 2006.
5. B. N. Grosz, I. Horrocks, R. Volz, and S. Decker. Description logic programs: Combining logic programs with description logic. In *Proc. of WWW 2003*, pages 48–57, 2003.
6. J. Heflin and J. Hendler. A portrait of the Semantic Web in action. *IEEE Intelligent Systems*, 16(2):54–59, 2001.
7. R. Hull. A survey of theoretical research on typed complex database objects. In J. Paredaens, editor, *Databases*, pages 193–256. Academic Press, 1988.
8. U. Hustadt, B. Motik, and U. Sattler. Data complexity of reasoning in very expressive description logics. In *Proc. of IJCAI 2005*, pages 466–471, 2005.
9. M. Lenzerini. Data integration: A theoretical perspective. In *Proc. of PODS 2002*, pages 233–246, 2002.
10. C. Lutz. Description logics with concrete domains: A survey. In P. Balbiani, N.-Y. Suzuki, F. Wolter, and M. Zakharyashev, editors, *Advances in Modal Logics*, volume 4. King’s College Publications, 2003.
11. M. M. Ortiz, D. Calvanese, and T. Eiter. Characterizing data complexity for conjunctive query answering in expressive description logics. In *Proc. of AAAI 2006*, 2006.
12. A. Poggi. *Structured and Semi-Structured Data Integration*. PhD thesis, Dip. di Inf. e Sist., Univ. di Roma “La Sapienza”, 2006.

² <http://www.dis.uniroma1.it/~quonto/>

A Data Mining Methodology for Anomaly Detection in Network Data: Choosing System-Defined Decision Boundaries

Costantina Caruso

Dipartimento di Informatica, Università degli Studi di Bari
Via E. Orabona 4 - 70126 Bari - Italy
caruso@di.uniba.it

Abstract. Anomaly detection is based on profiles that represent normal behavior of the analyzed system and detects novelties as significant deviations from these profiles. The methodology we use is based on the application of several data mining methods and returns an adaptive normal model of the network traffic of a LAN. An anomaly can both be a change point and an outlier. We rank the identified anomalies by means of two new measures in order to add to the model only change points and not outliers. This means to use two different thresholds which are critical parameters as to the model behavior and meaningfulness. In this work we analyze how to choose system-defined values for the thresholds. In our experiments we use an entire monthly set of incoming network connections of our Department network.

Keywords: Anomaly detection, ranking anomalies, decision boundary, adaptive model, data mining, machine learning, network traffic analysis.

1. Introduction and related work

Our aim is to build an anomaly detector based on real observations (i.e. network connections) and in this work we refine the choice of the decision boundaries to be used to identify and rank anomalies as to the methodology we proposed in [20].

There are different aspects of anomaly detection that are important to underline.

Anomaly detectors should be able to recognize anomalies and to differentiate them in order to allow human operators to analyze them easily. Only recently the distinction between *outliers* and *change points* has emerged. Roughly, outlier detection is mainly interested in looking for isolated and exceptional points while change point detection looks for changing patterns. This difference is clarified both in [1], which considers two different behaviors when analyzing static or dynamic datasets, and in [2], which deal with the issue of detecting outliers and change points from time series. Similarly, also the adaptive property appears in recent works [3]. However, most of works face the problem of creating the normal model by applying classical learning algorithms and then detecting anomalies in a static context.

Classical supervised learning algorithms are hardly useful for anomaly detection tasks. They are designed for classification tasks and need labeled examples belonging to all the classes the analyzed system can present. In anomaly detection, we are typically given only one class of examples. In some cases, we can have normal (clean) data, i.e. data we know to be generated from the system in a normal operating

condition, and we learn the model from this single class. However, it is difficult to have clean data because they can contain attacks if they are real, while they represent only a partial view of the system if they are simulated.

Works on anomaly detection reported in literature can be classified primarily by their type of training data (supervised or not), and, secondarily, by their modeling technique: distance-based, statistical (probabilistic) approach, and profiling techniques. *Distance-based methods* [4, 5, 6, 7, 8] are widely used as unsupervised approach: they can work properly without previous knowledge. The idea in distance based methods is to represent every observation by means of a feature vector and to apply a distance function to measure how much the two observations are far apart. *Statistical approaches* [9, 10, 11, 12] are used in both supervised and unsupervised cases. Data points are modelled using a stochastic distribution; the drawback is that with high dimensions, it is difficult to estimate distributions. *Profiling techniques* [13, 14, 15] are applied when observation units concerning either human or machine or system behavior are both supervised and arranged in sequences or time series.

The learning process must generate a model that has to be able to differentiate exactly between positive and negative cases. Thus we face the *decision boundary* problem: to fix a practical criterion (which is typically based on thresholds use) to decide if a new behavior is an anomaly or not. Thresholds values are either defined by an expert or chosen on the basis of statistical properties of the analyzed system. The first choice depends on the expert's intuition while the latter fails if applied to systems whose statistical properties vary along the time as it happens in network traffic.

The methodology, initially outlined in [19], has two stages. Its preprocessing phase begins with the application of an unsupervised (clustering) method to feature vectors representing daily network connections. Clusters are then described as sets of rules. The last steps of the preprocessing phase consist of the transformation of rules into symbolic objects (SOs) and in the subsequent computation of similarities between symbolic objects in order to make the model adaptive. An anomaly is defined to be a symbolic object which deviates too much from the model; a ranking mechanism is used to differentiate anomalies in order to identify more precisely change points, which express an evolution of the system, and true outliers which can be caused by either malfunctioning or intrusions.

We need to use some thresholds to rank anomalies. It is important to identify system-dependent values in order to avoid a random choice. In this work we identify a general criterion for choosing thresholds values. We use two clustering algorithms, K-means and EM, as first step of our methodology and, even if clusterized data we obtain are different, the adaptive models behaves similarly when the same selective criterion is used to fix thresholds values.

2. Outline of our methodology

Source data are divided in successive time units, t . For each time unit t a normal *static* model $So(t)$ is generated in the pre-processing *three-stepped* phase of the

¹ The time unit we choose in our experiments is the entire day but the approach is general and the most suitable time unit can be used: seconds, minutes, hours or years and so on.

methodology and, then, change points identified for each normal static model are added to the normal *adaptive* model $M(t)$ in order to update it. In the start-up phase, $M(t)$ is initialized to $So(0)$ i.e. the first static normal model generated for $t=0$.

The first step of the preprocessing phase of the methodology has to be based on an EDA technique, since we know nothing about our system and we have no labeled examples. This justifies the use of clustering as our starting point.

Clusters are groups of similar objects and are an extensional form of knowledge representation. Intensional descriptions are essential in our context since they provide human-interpretable patterns of network traffic, and they are a lighter form of knowledge representation which is a good quality for a light computationally tool to use in real-time. Therefore, a set of rules, whose consequents represent the cluster membership, as an intensional description of clusters, is generated in the second step. A rule R corresponds to homogeneous groups of connections, that is, to second-order objects, or symbolic objects (SO), according to the terminology used in symbolic data analysis [18]. For obtaining a dynamic representation of network traffic, we transform rules into symbolic objects and then compute the dissimilarities between different SOs. The transformation of a rule R into the corresponding symbolic object is illustrated in [19] and represents the last step of the pre-processing phase.

The symbolic objects set $So(t)$ generated for the time unit t is the *static normal model* of network traffic observed at time t . The normal adaptive model $M(t)$ at time t is given by the union of the normal adaptive model $M(t-1)$ at time $(t-1)$ and the set of all the change points found in $So(t)$; formally:

$$M(t) = M(t-1) \cup ChPoints(t). \quad (2.1)$$

$ChPoints(t)$ is formed by all the anomalies in $So(t)$ but ranked as change points and not as outliers.

Fixed a threshold T , given a dissimilarity measure D and a symbolic object So in $M(t)$, all the SOs in $So(t)$ whose dissimilarity from So is less than T are considered similar to So , otherwise they are tagged as anomalies. Formally:

Definition 2.1. The symbolic object $So_k(t)$ in $So(t)$ is an *anomaly* if

$$D_{jk} = D(So_j M(t), So_k(t)) > T \quad \text{for each } j=1, \dots, |M(t)| \quad (2.2)$$

We need to differentiate anomalies on the fly in order to adapt the model only by means of novel events of the system and not by real outliers. In most research works, outlier detection and change point detection have not been related explicitly and the adaptive properties, when considered, are *built-in* in the model. We consider this approach unsound, and we propose an explicit ranking mechanism between anomalies. We define two further quantities to differentiate among the various type of anomalies: **mean and minimum dissimilarities**.

Mean dissimilarity When analyzing SOs at time t , it is significant to know how much a SO is similar to all the SOs belonging to the adaptive model $M(t)$ in order to rank its level of dissimilarity. This information about a symbolic object $So_k(t)$ can be obtained by computing its dissimilarity mean value defined as follows:

$$D_{mean}(So_k(t)) = \frac{\sum_{j=1}^{|M_j|} D(So_k(t), So_j M(t))}{|M_j|} \quad (2.3)$$

Minimum dissimilarity Another interesting parameter is the minimum value of dissimilarity of a SO; indeed, a SO with high mean dissimilarity could be similar to few others but very dissimilar from the remaining ones and the mean value is not able to capture this situation. Therefore we compute the minimum value of dissimilarity between a fixed SO and all the SOs belonging to $M(t)$:

$$D_{\min}(So_k(t)) = \min_{j=1}^{|M_j|} D(So_k(t), So_j M(t)) \quad (2.4)$$

Definition 2.2. A symbolic object So_k in $So(t)$ is an anomaly of *prevalent type* PA_k if and only if the following condition holds:

$$(D_{\text{mean}}(So_k(t)) \leq T_{\text{mean}}) \wedge (D_{\min}(So_k(t)) \leq T_{\min})$$

Definition 2.3. A symbolic object So_k in $So(t)$ is a anomaly of *secondary type* SA_k if and only if the following condition holds:

$$(D_{\text{mean}}(So_k(t)) > T_{\text{mean}}) \wedge (D_{\min}(So_k(t)) \leq T_{\min})$$

Definition 2.4. A symbolic object So_k in $So(t)$ is an *outlier* O_k if and only if the following condition holds:

$$(D_{\text{mean}}(So_k(t)) > T_{\text{mean}}) \wedge (D_{\min}(So_k(t)) > T_{\min})$$

Definition 2.5. Let $ChPoints(t) = \{\text{the set of all } PA_k \text{ and } SA_k \text{ found in } So(t)\}$

3. Choosing the thresholds

We need to identify three thresholds values: T , T_{mean} and T_{\min} . It must be avoid a random or an expert-dependent choice of the three parameters. The idea is to give a general criterion to apply in order to identify a set of three single values or three values ranges which have to be system-dependent.

T , T_{mean} and T_{\min} , all of them influence deeply the behavior of the adaptive model. However the most critical parameters are T_{mean} and T_{\min} as the ultimate decision (the ranking process) is delegated to them. T can be given a *narrow* value which is useful to *push out* all the anomalies. A small value for T minimizes the false negative rate while the false positive rate is controlled by the other two parameters.

The question we try to answer is: do a set of thresholds values exist which demonstrates to be system-dependent and is able to capture the behavior of our incoming Internet traffic?

First of all we try to identify the range of values for T which is the first parameter to differentiate between a known behaviour and an anomaly. We look for system-dependent values by computing the *averaged minimum* of all minimum dissimilarity values and the *averaged mean* of all mean dissimilarity values as pivot values. The averaged minimum represents the system-dependent distance between two different SOs i.e. two SOs are not closer in average; the averaged mean suggests that two SOs can not be, in average, more distant from each other.

The averaged minimum and the averaged mean are also used to identify the range of values to give to T_{mean} and T_{\min} . The first is given a value interval, the averaged

mean is the central point of; the latter is given a value interval, the averaged minimum is the central point of.

Our experiments suggest us the best ranges the thresholds values must vary in, are:

$$T = [\textit{averaged minimum} .. \textit{averaged mean}] \quad (3.1)$$

$$T_{\textit{mean}} = [\textit{averaged mean} - \mathbf{n}, .. , \textit{averaged mean} + \mathbf{n}] \quad (3.2)$$

$$T_{\textit{min}} = [\textit{averaged minimum} - \mathbf{n}, .. , \textit{averaged minimum} + \mathbf{n}] \quad (3.3)$$

where \mathbf{n} has to be fixed but it is a small integer (in our experiments \mathbf{n} is equal to 2).

This is confirmed by the results we obtain by using two different clustering methods as the first step of the normal static model generation phase. The targeted data set is the same but the clusterized data are different.

4. Experiments and results

4.1 Data collection and preprocessing

The source data set is formed by four successive weeks of firewall logs of our Department, from May 31st to June 27th, 2004. A transaction is a single logged packet. The target dataset is built by reconstructing the entire connections from single packets. In our analysis we have used only accepted ingoing connections and we have a file for every day that contains all the connections opened and closed in that day.

Since the goal is to create the daily description of our connections, we have chosen to work with few but fundamental attributes, namely: *Proto* (nominal): which can have only two values: udp and tcp; *StartHalfHour* (integer), the time field of a packet; *Dst* (integer) which represents the IP of department public servers; *Service* (nominal): the requested service (http, ftp, smtp and many other ports); *NumPackets* (integer): the number of connection packets; *Length* (integer): the time length of the connection; *NationCode* (nominal): the two digit code of the nation the source IP belongs to.

In the clustering step we used k-means [16] and EM [21] while the rules, which provide an intensional description of clusters, are generated by means of the algorithm PART [17].

The first 10 rules with maximum support and confidence ≥ 0.9 , are selected and transformed into symbolic objects [19].

The SOs we obtain represent the most prevalent aspects in our network traffic and can contain only change points i.e. points which represent a natural change of the network traffic.

4.2 Evaluating the normal adaptive model

The normal adaptive model $M(t)$ we build is very simple but we must evaluate its performance i.e. we have to demonstrate it is able to represent significantly the network traffic and its evolution.

It is not possible to use techniques like ROC curves to evaluate its performance because we know nothing about data. We can obtain only statistical profiles about data to know what was the network traffic behavior along the 4 analyzed weeks.

In Fig.1 we have represented the statistical profiles of two significant attributes: *Proto*, *Dst*. In each graph the trends of the values (or the most significant ones) the attributes can assume, are represented. Thus in *Proto* trends, we have the tendencies

of the two only values the attribute *Proto* can have: *udp* (the line) and *tcp* (the solid). The *Dst* attribute can assume a lot of values; we reported the tendencies for the set of values {8, 135, 10, 45, 153} which identify five public servers of our department.

It is evident that we can aggregate the 28 days in two big groups according to their statistical profiles: group A formed by {31, 1, 4, 5, 6, 22, 23, 24} and group B all the remaining days belong to. The different behaviour is ought to the p2p connections explosion we have in group A days.

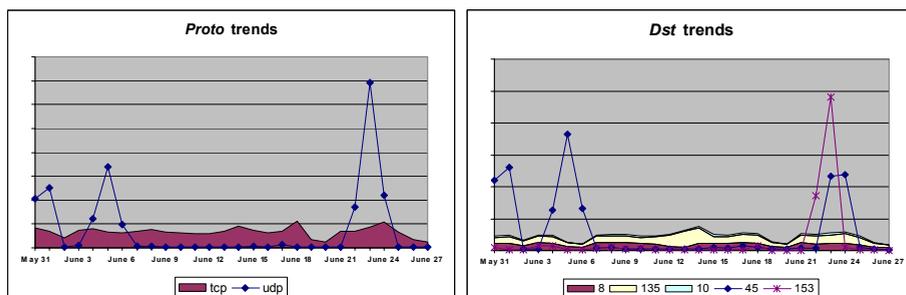


Fig. 1. In the two graphs, *Proto* and *Dst*, trends are represented; on the y-axis there is the number of instances while on the x-axis day labels are reported.

The idea is to show that the system-dependent sets of the three threshold values for T , T_{mean} , T_{min} , defined in (3.1), (3.2) and (3.3) generate a normal adaptive model able to differentiate these two days groups.

The normal adaptive model $M(t)$ is meaningful if new SOs are added to it only when the statistical profile of the analyzed day is different from that one of the previous days. The model should not be modified when we face days whose statistical profile is already known. We expect that the model will be modified strongly in the first week while new SOs are less and less added as days go on. Further, the network traffic behavior of the days with an already known profile, has to be represented mainly by means of SOs generated in similar previous days.

4.3 K-means experiments

$M(0)$, i.e. the start-up model, is initialized to the entire set of SOs generated for the first day (May 31st).

In the experiments, the three thresholds vary in these intervals: $T = [2..5]$, $T_{mean} = [3..7]$, $T_{min} = [1..4]$ where the value 5.0 is the averaged mean and the value 2.0 is the averaged minimum of all the monthly dissimilarity quantities.

The models obtained for $T=[3..5]$ along all the 4 weeks, are not satisfactory because they possess almost the same set of SOs of the starting point, $M(0)$. The models obtained for $T=2$ along all the 4 weeks react to the different network traffic behaviors as shown in **Fig.2**.

In **Fig.3** we report the number of SOs added to the model along the four weeks for the threshold set (2,7,4) while in **Fig. 4** shows that the model we obtain represents successive group A days by means of group A's SOs and successive group B days by means of group B's SOs.

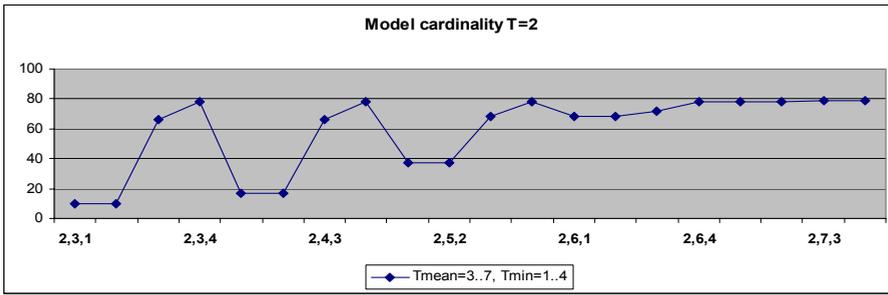


Fig. 2. Model cardinality: K-means, T=2.

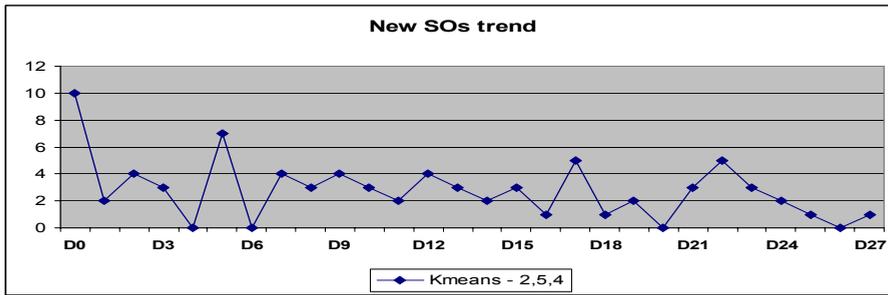


Fig. 3. New SOs trend for the set (2,5,4).

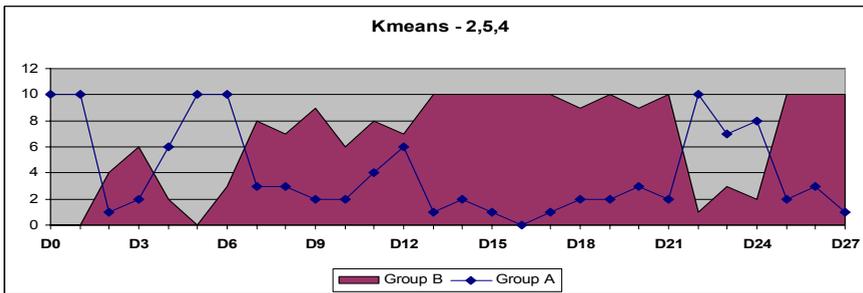


Fig. 4. Group A and Group B components for the set (2,5,4).

4.4 EM experiments

$M(0)$, i.e. the start-up model, is initialized to the entire set of SOs generated for the first day (May 31st).

In the experiments, the three thresholds vary in these intervals: $T = [3..7]$, $T_{mean} = [5..9]$, $T_{min} = [2..5]$ where the value 7.0 is the averaged mean and the value 3.0 is the averaged minimum of all the monthly dissimilarity quantities.

The models obtained for $T=[4..7]$ along all the 4 weeks, are not satisfactory because they possess almost the same set of SOs of the starting point, $M(0)$. The models obtained for $T=3$ along all the 4 weeks react to the different network traffic behaviors as shown in Fig.5.

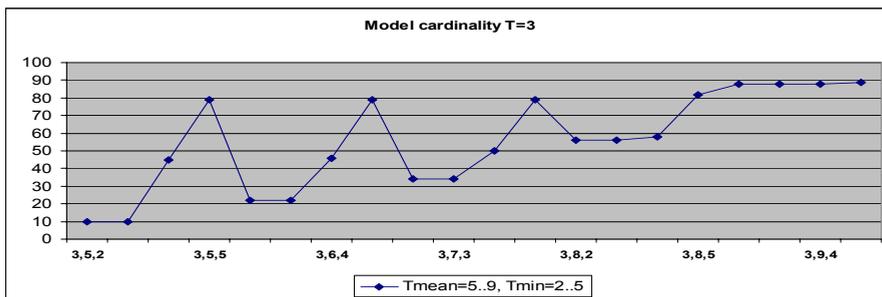


Fig. 5. Model cardinality: K-means, $T=2$.

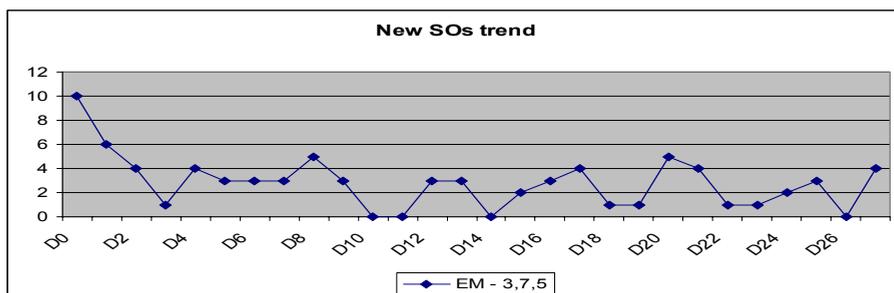


Fig. 6. New SOs trend for the set $(2,7,5)$.

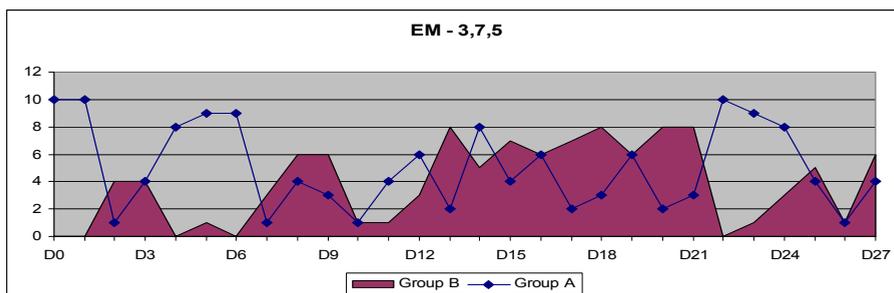


Fig. 7. Group A and Group B components for the set $(3,7,5)$.

In Fig.6 we report the number of SOs added to the model along the four weeks for the threshold set $(3,7,5)$ while in Fig. 7 shows that the model we obtain represents successive group A days by means of group A's SOs and successive group B days by means of group B's SOs.

5. Conclusions and future work

The main aim of this work is to investigate a general criterion for choosing thresholds values. We used two clustering algorithms, K-means and EM, as first step of our methodology and, even if clustered data we obtain are different, the adaptive models behavior similarly by using the same selective criterion in fixing thresholds values.

The experiments have been made on the network traffic of our department but the

features used are general and so this methodology can be applied to the traffic of every network active device. Source data represent prevalent aspects of our network traffic and this implies that they only contain legitimate behavior. Our future work will concentrate on verifying the proposed general criterion in fixing thresholds values by using different source data and varying the EDA step of the preprocessing phase of our methodology by means of other data mining techniques.

References

1. Ghoting, A., Otey, M.E., Parthasarathy, S.: Loaded: Link-based Outlier and Anomaly detection in Evolving Data Sets. In Proc. of the IEEE Int. Conf. on Data Mining, 2004
2. Takeuchi, J., Yamanashi, K.: A Unifying Framework for Identifying Changing Points and Outliers. IEEE Transactions on Knowledge and Data Engineering. Vol.18, No.4, 2006
3. Wang, K., Stolfo, S.: Anomalous Payload-based Network Intrusion Detection. RAID2004
4. Knorr, N., Ng, P.: Algorithms for Mining Distance-Based Outliers in Large Datasets. Proc. of VLDB. 1998
5. Ramaswamy, S., Rastogi, R., Kyuseok, S.: Efficient Algorithms for Mining Outliers from Large Data Sets. ACM SIGMOD Conference on Management of Data. 2000.
6. Breunig, et al.: LOF: Identifying Density-Based Local Outliers. KDD 2000.
7. Lazarevic, L. et al.: A comparative Study of Anomaly Detection Schemes in Network Intrusion Detection. SIAM 2003.
8. Eskin, E., Arnold, A., Prerau, M., Portnoy, L., Stolfo, S.: A Geometric Framework for Unsupervised Anomaly Detection: Detecting Intrusions in Unlabeled Data. 2002.
9. Shmueli, G.: Current and Potential Statistical Methods for Anomaly Detection in Modern Time Series Data: The Case of Biosurveillance. Data Mining Methods for Anomaly Detection KDD-2005.
10. Yamanishi, K.: On-line unsupervised outlier detection using finite mixture with discounting learning algorithms. KDD 2000.
11. Mahoney, M., Chan, P.: Learning Nonstationary Models of Normal Network Traffic for Detecting Novel Attacks. 8th ACM KDD. 2002.
12. Mahenshikumar, R.S., Neill, D.B., Moore, A.W.: Detecting Anomalous Patterns in Pharmacy Retail Data. Data Mining Methods for Anomaly Detection KDD-2005.
13. Hofmeyr, S. et al.: Intrusion Detection using Sequences of System Calls. 1997.
14. Tandon, G., Chan, P.: Learning Rules from System Call Arguments and Sequences for Anomaly Detection. Workshop on Data Mining for Computer Security. ICDM 2003.
15. Wang, K., Stolfo, S.: One Class Training for Masquerade Detection. Workshop on Data Mining for Computer Security. ICDM 2003.
16. Jain, A.K., Murty, M.N., Flynn, P.J. Data Clustering: a Review. ACM Computing Surveys, Vol.31, No.3.1999.
17. Witten I., Frank E., Generate Accurate Rule Sets Without Global Optimisation. Proc. of the 15th Int. Conf. on Machine Learning, Morgan Kaufmann, San Francisco, USA (1998)
18. Gowda, K. C., Diday, E.: Symbolic Clustering Using a New Dissimilarity Measure. In Pattern Recognition, Vol. 24, No. 6 (1991) 567-578
19. C. Caruso, D. Malerba, D. Papagni. Learning the daily model of network traffic. Proceedings of ISMIS 2005, 15th International Symposium, Saratoga Springs, NY, USA, May 2005. Springer, LNAI 3488; Foundations of Intelligent Systems; pagg. 131-141.
20. C. Caruso, D. Malerba. A Data Mining Methodology for Anomaly Detection in Network Data. Proc. of the 11th Int. Conf. on Knowledge-Based and Intelligent Information & Engineering Systems. 2007. To appear.
21. McLachlan, G.J., Krishan, T.: The EM Algorithm and Extensions. John Wiley & Sons. 1997.

Exploiting Peer Ontologies for Semantic Query Propagation *

Silvana Castano, Alfio Ferrara, and Stefano Montanelli

Università degli Studi di Milano
DICO - Via Comelico, 39, 20135 Milano - Italy
{castano,ferrara,montanelli}@dico.unimi.it

Abstract. A challenging issue to advance the existing P2P query propagation protocols is related to the capability of developing a routing mechanism where a semantically rich description of the context of each peer is explicitly taken into account for selecting the query recipients. In this paper, we present the *H-Link semantic routing approach* designed to exploit peer ontologies and ontology matchmaking results for providing a semantic overlay network where peers having similar contexts are recognized and interlinked as *semantic neighbors*.

1 Introduction

In order to provide scalable infrastructures for peer communications, semantic-based P2P query propagation protocols are being proposed with the aim of identifying those peers that are most likely to provide relevant results according to the query content [1]. However, most of the existing approaches rely on a rather simplifying assumption of a centralized repository of knowledge where mappings among the distributed peer resource descriptions are maintained [2, 3]. In some other approaches, the knowledge model supported by the peers is kept quite poor (i.e., metadata rather than ontologies) in order to reduce the complexity of the matching process. In such cases, syntactic matching techniques (e.g., string- and keyword-based techniques) are generally employed to compute the similarity among the resource descriptions of the different peers, thus leading to a poor accuracy in query recipient selection [4, 5].

In this paper, we present the *H-Link semantic routing mechanism* designed to exploit the results of an ontology matchmaking process for providing a semantic overlay network where peers having similar contexts are recognized and interlinked as *semantic neighbors*. In particular, H-LINK aims at advancing the existing query propagation protocols by combining ontology-based peer context descriptions and ontology matching techniques for providing query forwarding on a real semantic basis, in a completely decentralized way. Furthermore, H-LINK

* This paper has been partially funded by the ESTEEM PRIN project funded by the Italian Ministry of Education, University, and Research. An extended version of this paper is published on the *Knowledge Engineering Review Journal, Special Issue on Contexts and Ontologies*, 2007 (to appear).

aims at enforcing scalability in query forwarding by introducing a *credit-based* mechanism where the approximate number of desired replies is specified rather than the non-scalable number of hops to cross.

2 The H-Link approach to semantic query propagation

The key idea of H-LINK is to exploit the results of knowledge discovery interactions to train the behavior of the query propagation mechanism. To this end, peers are connected through matching-based *confidence* measures that keep track of the semantic affinity among the knowledge of different peers. As a result, peers are organized in a semantic overlay network where nodes having similar knowledge are interlinked as *semantic neighbors*. The following main features characterize the H-LINK mechanism.

Use of a dynamic knowledge discovery approach. In H-LINK, peers interact by submitting *discovery queries* with the aim to identify relevant partners with respect to one or more target concepts of interest. Receiving a discovery query, a peer evaluates whether it is capable of providing concepts matching the target request. According to the results of the matching process, the list of concepts found to be relevant are replied to the requesting peer together with their associated semantic affinity values. Semantic affinity values provide a measure of the level of similarity between the target concepts of the query and the discovered matching concepts. In H-LINK, the replying nodes are linked to the requesting peer as semantic neighbors and the returned affinity values are exploited to set the level of confidence of each semantic neighbor with respect to the discovered matching concepts. This way, as a peer learns about the network contents through discovery queries, also its network knowledge gradually evolves to reflect its newly acquired semantic neighbors.

Use of peer ontologies. In H-LINK, both queries and peer resources are expressed in terms of ontological descriptions. In particular, each query contains a list of target concept(s) of interest with possible properties and semantic relations that further specify the request. Furthermore, each peer joining the system provides a peer ontology where knowledge is organized into a two-layer architecture, namely the *content knowledge layer* describing the knowledge the peer brings to the network, and the *network knowledge layer* describing the knowledge the peer has of the semantic neighbors it has interacted with. In particular, the network knowledge layer is seen as a set of *network concepts* NC and *location relations* LR . A network concept $nc \in NC$ provides an abstract representation of a semantic neighbor (i.e., a peer) that has been identified during the knowledge discovery process. A location relation is defined to connect a network concept nc with a concept c in the content knowledge layer. A *confidence* value cf is associated to a location relation to keep track of the discovered semantic affinity between c and the peer ontology of the semantic neighbor represented with nc .

Use of ontology matching techniques. In H-LINK, each peer is capable of providing ontology matching functionalities through the use of a semantic matchmaker.

Ontology matching is employed by a peer during the dynamic knowledge discovery process in order to assess whether it can provide relevant knowledge in reply to an incoming discovery query. Furthermore, ontology matching is also exploited in H-LINK in order to select the recipients of a query according to the expected semantic affinity between the query contents and the peer ontology of the receiving peer.

3 Ontology matchmaking in H-Link

A key feature of H-LINK is the use of ontology matching techniques for query recipient selection that are currently performed by relying on the H-MATCH semantic matchmaker. H-MATCH performs ontology matching at different levels of depth by deploying four different *matching models* spanning from surface to intensive matching, with the goal of providing a wide spectrum of metrics suited for dealing with many different matching scenarios that can be encountered in comparing concept descriptions of real ontologies. H-MATCH takes two ontologies as input and returns the mappings that identify corresponding concepts in the two ontologies, namely the concepts with the same or the closest intended meaning. A threshold-based mechanism is enforced to set the minimum level of semantic affinity required to consider two concepts as matching concepts. Given two concepts c and c' , H-MATCH calculates a semantic affinity value $SA(c, c') \in [0, 1]$ as the linear combination of a linguistic affinity value $LA(c, c')$ and a contextual affinity value $CA(c, c')$. The linguistic affinity function of H-MATCH provides a measure of similarity between two ontology concepts c and c' computed on the basis of their linguistic features (i.e., concept names). For the linguistic affinity evaluation, H-MATCH relies on a thesaurus of terms and terminological relationships automatically extracted from the WordNet lexical system. The contextual affinity function of H-MATCH provides a measure of similarity by taking into account the contextual features of the ontology concepts c and c' . The context of a concept can include properties, semantic relations with other concepts, and property values. The context can be differently composed to consider different levels of semantic complexity, and four matching models, namely, *surface*, *shallow*, *deep*, and *intensive*, are defined to this end. In the surface matching, only the linguistic affinity between the concept names of c and c' is considered to determine concept similarity. In the shallow, deep, and intensive matching, also contextual affinity is taken into account to determine concept similarity. In particular, the shallow matching computes the contextual affinity by considering the context of c and c' as composed only by their properties. Deep and intensive matching extend the depth of concept context for the contextual affinity evaluation of c and c' , by considering also semantic relations with other concepts (deep matching model) as well as property values (intensive matching model), respectively.

A detailed description of the H-MATCH models and related techniques is provided in [6]. H-MATCH has been extensively tested on several real ontology matching cases in order to evaluate the matching models with respect to perfor-

mance and quality of results [6]. According to the obtained results, the choice to use H-MATCH for supporting H-LINK is motivated by the fact that H-MATCH can be dynamically configured to tune the tradeoff between performance and accuracy according to the requirements of the considered matching scenario. In this sense, other existing matching tools can however be used to enforce H-LINK in turn of H-MATCH provided that a dynamic and flexible configuration is supported.

4 Query propagation in H-Link

The H-LINK mechanism is based on the idea of exploiting the network knowledge layer of a peer ontology by using the H-MATCH semantic matchmaker for providing query distribution support according to semantic neighbor contents.

We consider a query q with a target concept tc ¹. Two different roles can be distinguished for a given peer p :

- *Requesting peer.* Peer p needs to submit to the network a query q in order to identify relevant partners for subsequent resource sharing. To this end, peer p invokes H-MATCH to compare the target concept tc against the content knowledge layer of its peer ontology \mathcal{O} . A list $MCL = \{\langle c_1, SA(tc, c_1) \rangle \dots \langle c_n, SA(tc, c_n) \rangle\}$ of matching concepts $c_1 \dots c_n \in \mathcal{O}$ and corresponding semantic affinity values $SA(tc, c_1) \dots SA(tc, c_n)$ is returned as a result. Peer p sets the *number of credits* N_{cr} to distribute to the query recipients in order to define the number of replies that peer p wish to receive as answers to the query q . Therefore, H-LINK is invoked by passing the list MCL to select the semantic neighbors for query q submission.
- *Receiving peer.* When a peer p receives a query q together with the number of credits nc from a requesting peer r , it needs to evaluate whether matching concepts can be provided back to peer r . To this end, H-MATCH is invoked by peer p and the list MCL of matching concepts is still produced as a results. If $MCL \neq \emptyset$, the peer p sends MCL back to peer r by consuming one credit, otherwise no reply is sent back to peer r and all the received credits are still available for forwarding. If at least one credit is available, H-LINK is invoked by peer p to select the semantic neighbors for query q forwarding; otherwise the propagation mechanism stops. Query replies are returned to the requesting peer by following the reverse query path in order to avoid a sudden burst of incoming messages as suggested in [5].

H-Link invocation. H-LINK is invoked for both query submission/forwarding provided that at least one credit is still available. Three main steps define H-LINK: *selection of semantic neighbors; ranking of semantic neighbors; distribution of credits.*

¹ For the sake of clarity, we consider the case of a single target concept in the query. The H-LINK mechanism can be easily extended to consider the case of multiple target concepts.

- 1- **Selection of semantic neighbors.** The network knowledge layer of the peer ontology is exploited to select the network concepts, together with the associated confidence values, that are connected to the concepts in *MCL* through a location relation. A list *SNL* of semantic neighbors is returned as a result. A semantic neighbor $sn \in SNL$ is described in the form $sn = \langle nc, \{c_1, cf_1 \dots c_m, cf_m\} \rangle$, where nc is the network concept featuring sn , while $c_1 \dots c_m \in MCL$ are the concepts of *MCL* connected to nc through a location relation, and $\{cf_1 \dots cf_m\}$ the corresponding confidence values.
- 2- **Ranking of semantic neighbors.** Semantic neighbors in *SNL* are ranked with respect to their relevance for the query target tc . To this end, the harmonic mean is used to combine the confidence values associated with the semantic neighbors in *SNL* and the semantic affinity values in *MCL*. Given a semantic neighbor $sn \in SNL$, the ranking value r_{sn} corresponds to the following formula:

$$r_{sn} = \frac{\sum_{i=1}^m \frac{2 \cdot cf_i \cdot SA(tc, c_i)}{cf_i + SA(tc, c_i)}}{m} \quad (1)$$

Finally, a ranked list *RSNL* of semantic neighbors with the corresponding ranking value is returned as a result. A threshold mechanism can be used to rule out the semantic neighbors with a ranking value lower than a predefined threshold t .

- 3- **Distribution of credits.** The semantic neighbors in *RSNL* determine the recipients of the query q . Available credits A_{cr} are proportionally distributed to the semantic neighbors in *RSNL* according to their ranking value.

When $MCL = \emptyset$, that is the peer ontology does not contain relevant concepts with respect to the target query, credits are proportionally distributed by ranking each peer according to a comprehensive *expertise* measure. For each peer nc in the network knowledge layer, expertise is computed as the average mean of the confidence values associated with all the location relations connected with nc . A detailed description H-LINK is provided in [7] where the H-LINK proposals for addressing some typical query propagation issues are also discussed.

Example. As an example of H-LINK query propagation, we consider a peer B with the associated peer ontology of Figure 1(b). The peer B intends to submit to the system the query Q described in Figure 1(a) with total number of credits to distribute $N_{cr} = 5$. The peer B uses H-MATCH to compare the query Q against its peer ontology. As a result, the following semantic affinity values are returned by H-MATCH: $SA(\text{Book}, \text{Volume})=0.79$ and $SA(\text{Book}, \text{Publication})=0.49$. By invoking H-LINK, we find that:

$MCL = \{ \langle \text{Volume}, 0.79 \rangle, \langle \text{Publication}, 0.49 \rangle \}$
 $SNL = \{ \langle \text{peer A}, \{ \text{Volume}, 0.74 \} \rangle, \langle \text{peer E}, \{ \text{Publication}, 0.81 \} \rangle, \langle \text{peer F}, \{ \text{Volume}, 0.875, \text{Publication}, 0.62 \} \rangle \}$

On the basis of such results, H-LINK computes the ranking of the semantic neighbors in *SNL* and assigns the corresponding number of credits, as follows:

Semantic neighbor	Ranking value	Assigned credits
peer A	0.764	2
peer E	0.611	1
peer F	0.689	2

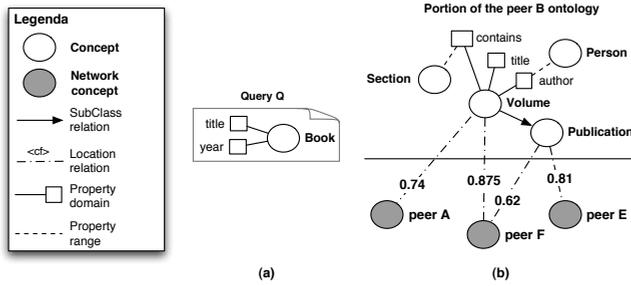


Fig. 1. (a) The query Q example and (b) a portion of the peer B ontology

The query Q is then submitted to the selected semantic neighbors together with the assigned number of credits. As shown in the query propagation schema of Figure 2, peer A receives the query, consumes one credit for replying to peer B, and forwards the query Q to peer D by assigning the last remaining credit. peer E consumes the unique credit received and stops the forwarding process, while the peer F forwards all the received credits to peer G as no reply is sent back to peer B.

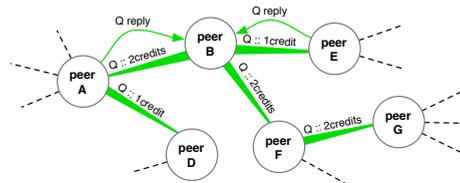
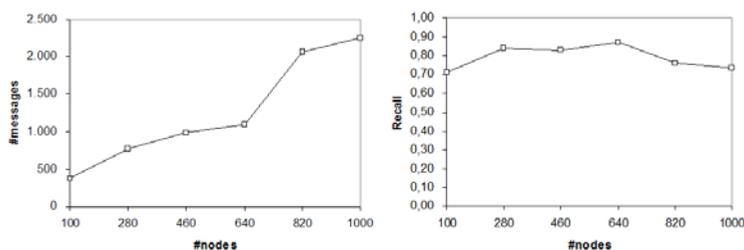


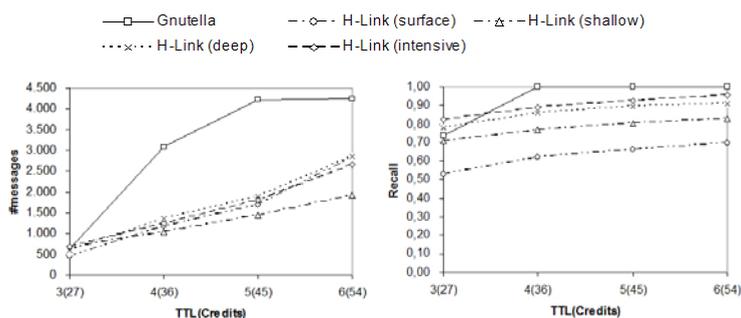
Fig. 2. The H-LINK propagation schema for the query Q

5 Experimental results

The goals of the experiments is to evaluate H-LINK in terms of generated traffic and recall. As generated traffic, we mean the overall number of messages routed during a complete simulation run, while recall is measured as the ratio of the number of relevant concepts retrieved by a H-LINK query to the total number of relevant concepts that was available in the network. Several tests have been produced by varying the most important configuration parameters (i.e., #nodes, #connections per node, #queries, #credits per query). In this section, we report some snapshots that are appropriate for discussing the H-LINK performance. The complete experimental results are provided in [7].



(a) - H-Link scalability: traffic and recall



(b) - Comparison with Gnutella: traffic and recall

Fig. 3. Evaluation of H-LINK with #queries = 5000

In Figure 3(a), we evaluate H-LINK in terms of scalability by measuring traffic and recall when #credits = 24 and the number of peers is a growing variable. We observe that the generated traffic follows a sub-linear growing, while recall is not significantly affected by the #peer variation. The growing traffic is due to the random generation of queries and peer ontologies during the simulation. We believe that this promising result can be further improved by exploiting more realistic distribution models (e.g., the Zipf distribution model) that we plan to consider in future experiments. As another experiment we compare H-LINK with a well-known query propagation mechanism, like Gnutella. In Figure 3(b), we show the results of the comparison when #nodes = 500 and the number of credits per query is a growing variable. In Gnutella, credit-based query propagation is not supported. For this reason, TTL is used to set the scope of Gnutella queries while credits are proportionally defined for H-LINK. Furthermore, the results of H-LINK are also analyzed by varying the H-MATCH model from surface to intensive in the peer confidence computation. We note that excellent results are obtained by H-LINK in terms of generated traffic. Moreover, we also note that the variation in the adopted matching model does not significantly affect the performance of H-LINK in terms of generated traffic. For what concern the recall values, the optimal behavior of Gnutella is motivated by the fact that Gnutella floods the network and succeeds in reaching a large part of either relevant and

irrelevant nodes, thus retrieving all the available concepts matching the target query. On the other hand, H-LINK presents very interesting results in terms of recall values especially with more accurate matching models.

6 Concluding remarks

In this paper, the main features of the H-LINK mechanism for query propagation has been presented. The results obtained in the experiments show that H-LINK succeeds in improving the effectiveness of traditional P2P query protocols by providing interesting results both in terms of scalability and accuracy. Future work will regard the definition of further experiments in order to compare H-LINK with recently proposed semantic routing protocols (e.g., REMINDIN' [1]). Moreover, H-LINK is the basic mechanism we plan to use for supporting the formation of semantic communities of peers. On these topic, we are currently working in the framework of the Esteem project and some initial results are presented in [8].

References

1. Staab, S., Tempich, C., Wranik, A.: REMINDIN': Semantic Query Routing in Peer-to-Peer Networks based on Social Metaphors. In: Proc. of the 13th Int. conference on World Wide Web (WWW 2004), New York, NY, USA (2004)
2. Tsoumakos, D., Roussopoulos, N.: Adaptive Probabilistic Search (APS) for Peer-to-Peer Networks. In: Proc. of the 3rd Int. IEEE Conference on P2P Computing, Linköping, Sweden (2003)
3. Haase, P., Siebes, R., van Harmelen, F.: Peer Selection in Peer-to-Peer Networks with Semantic Topologies. In: Proc. of the Int. Conference on Semantics of a Networked World (ICSNW), Paris, France (2004)
4. Joseph, S.: NeuroGrid: Semantically Routing Queries in Peer-to-Peer Networks. In: Proc. of the Int. Workshop on Peer-to-Peer Computing, Pisa, Italy (2002)
5. Borch, N., Vognild, N.: Searching Variably Connected Networks. In: Proc. of the Int. Conference on Pervasive Computing and Communications (PCC-04), Las Vegas, Nevada, USA (2004)
6. Castano, S., Ferrara, A., Montanelli, S.: Matching Ontologies in Open Networked Systems: Techniques and Applications. *Journal on Data Semantics* **V** (2006)
7. Castano, S., Montanelli, S.: Semantically Routing Queries in Peer-based Systems: the H-Link Approach. *The Knowledge Engineering Review* (2007) To appear.
8. Castano, S., Montanelli, S.: Semantic Self-Formation of Communities of Peers. In: Proc. of the ESWC Workshop on Ontologies in Peer-to-Peer Communities, Heraklion, Greece (2005)

Conditional Preferences: A New Semantics for Database Queries

Paolo Ciaccia
DEIS, University of Bologna, Italy
pciaccia@deis.unibo.it

Abstract. Preference queries aim to retrieve from large databases those objects that better match user’s requirements. Approaches proposed so far in the DB field for specifying preferences are limited when one needs to consider *conditional*, rather than absolute, preferences (e.g., I prefer driving by car in winter, and by motorbike in summer), which are common in context-aware applications. CP-nets are a powerful formalism for concisely representing such preferences, which has its roots in decision making problems. However, CP-nets, being based on a *ceteris paribus* (all else being equal) interpretation, are hardly applicable in complex DB scenarios. In this paper we introduce a new *totalitarian* (i.e., not *ceteris paribus*) semantics for CP-nets. We prove that our semantics is equivalent to *ceteris paribus* for complete acyclic CP-nets, whereas it avoids some counterintuitive effects of *ceteris paribus* when the CP-net is partially specified.

1 Introduction

The trend towards the personalization of information systems functionalities requires new models and techniques able to provide users with the “right information” at the “right time” in the “right place”. Context-aware applications are a remarkable step towards achieving this goal, the key idea being that of taking into account context information when processing user requests. In particular, ranking the result of a query should be based on the current user context, rather than on some absolute criterion.

Example 1. Consider the following database of hotels:

Name	Price	Stars	Rooms	Internet
Jolly	40	2	50	Yes
Continental	55	2	30	No
Excelsior	80	3	50	Yes
Rome	80	5	100	Yes
Holiday	60	4	20	No

When travelling for work, the user does not care about price and number of rooms, he preferring hotels with at least 4 stars and an Internet connection. In this case the best alternative is hotel Rome. However, if travelling for leisure, the user prefers small hotels (≤ 30 rooms) and whose price is at most 50 Euro. In this case no hotel satisfies both requirements, yet it can be argued that Continental, Jolly, and Holiday are the best available alternatives, since each of them satisfies one of the two user preferences. \square

Frameworks proposed so far in the DB field [Cho02,Kie02] have paid little attention to *conditional* preferences. On the other hand, these have been largely investigated by AI researchers, with a particular emphasis on *CP-nets* (Conditional Preference networks) [BBHP99,BBD⁺04,Wil04,GLTW05], a graph-based formalism able to “factorize” the specification of preference statements over a set of attributes. A CP-net statement like $\varphi = p : a_i > a_j$ is given a *ceteris paribus* interpretation, i.e., “given p prefer a_i to a_j only if values of other attributes are *equal*”.

In this paper we argue that the *ceteris paribus* semantics is unsuitable for real-world complex DB’s, since it provides counterintuitive results whenever the DB is *incomplete*, i.e., it does not contain all the possible alternatives for the preference attributes, and the CP-net is not completely specified (see next section for a definition of complete CP-nets). With the aim of preserving the strong points of CP-nets, we provide an alternative, so called *totalitarian*, semantics for CP-nets. We first show that, rather surprisingly, the new semantics is equivalent to *ceteris paribus* for complete acyclic CP-nets. Then we prove that for complete DB’s the two semantics, although leading to different preferences, always yield the same set of optimal results. Finally, we show that for incomplete DB’s and CP-nets the new semantics excludes from the result those tuples that are apparently sub-optimal with respect to user preferences.

2 Background on CP-nets

A CP-net over a set of attributes $X = \{A_1, \dots, A_n\}$ is a pair $N = (G, CPT)$, where $G = (X, E)$ is a directed graph and CPT is a function that associates to each $A_i \in X$ a *conditional preference table*, $CPT(A_i)$. If the arc $(A_j, A_i) \in E$, then A_j is a *parent* of A_i . Let P_i be the set of parents of A_i . Then, $CPT(A_i)$ consists of a set of preference statements φ of the form $\varphi = p : a_{i,1} > a_{i,2}$, where $p \in \text{dom}(P_i)$ and $a_{i,1}, a_{i,2} \in \text{dom}(A_i)$.¹ This expresses the conditional preference of $a_{i,1}$ with respect to $a_{i,2}$ given p . If A_i has no parents, then the statement simplifies to $\varphi = \perp : a_{i,1} > a_{i,2} \equiv a_{i,1} > a_{i,2}$, i.e., $a_{i,1}$ is unconditionally preferred to $a_{i,2}$.

Example 2. Figure 1 shows a simple CP-net over attributes RestaurantType (R), Table (T), and Price (P), thus $X = \{R, T, P\}$. For simplicity, all attributes have binary domains, in particular: $\text{dom}(R) = \{it, chn\}$ (italian or chinese), $\text{dom}(T) = \{in, out\}$ (inside or outside), and $\text{dom}(P) = \{low, high\}$. My preferences unconditionally go to italian restaurants ($it > chn$), for which I prefer to have a table inside ($it : in > out$) and pay the less ($it : low > high$). On other hand, in a chinese restaurant I prefer to sit outside ($chn : out > in$) and to pay more ($chn : high > low$). \square

Definition 1 A CP-net $N = (G, CPT)$ is:

- acyclic iff G is acyclic;
- locally consistent iff, for each attribute A_i , $CPT(A_i)$ does not include a “chain” of statements $\varphi_1, \dots, \varphi_m$ ($m > 1$), such that: $p : a_{i,1} > a_{i,2} > \dots > a_{i,1}$;
- complete iff, for each A_i and for each $p \in \text{dom}(P_i)$, $CPT(A_i)$ totally orders values in $\text{dom}(A_i)$, i.e., for each $a_{i,1}, a_{i,2}$ either $p : a_{i,1} > a_{i,2}$ or $p : a_{i,2} > a_{i,1}$.

¹ Equivalently, each statement might specify a conjunction of pair orderings of the form $a_{i,j} > a_{i,k}$, given a set of values from $\text{dom}(P_i)$.

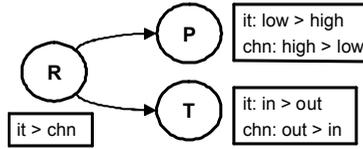


Fig. 1. A simple CP-net over 3 attributes

The CP-net in Figure 1 is acyclic and locally consistent. Further, it is also complete. Should we drop one statement (e.g., $it : in > out$) we would have an incomplete CP-net. If the CP-net is locally consistent, no contradiction is present as long as we consider preferences over any single attribute. In the following we only consider acyclic and locally consistent CP-nets.

The standard *ceteris paribus* interpretation of a statement $\varphi = p : a_{i,1} > a_{i,2}$, $\varphi \in CPT(A_i)$, is the set of pairs of tuples over X :

$$\varphi_{cp}^* = \{((p, a_{i,1}, y), (p, a_{i,2}, y)) \mid y \in \text{dom}(X - P_i - \{A_i\})\} \quad (1)$$

in which y is any value of $\text{dom}(Y_i)$, Y_i being the set of attributes not involved in φ . Thus, each preference induced by φ concerns two tuples that differ *only* in the value of A_i . Let $\Phi_{A_i, cp}^* = \bigcup_{\varphi \in CPT(A_i)} \varphi_{cp}^*$ denote all preferences induced by $CPT(A_i)$. Since the CP-net is locally consistent, no conflicts are present in $\Phi_{A_i, cp}^*$. Further, it is easy to see that, due to the cp semantics, any two tuples t_1 and t_2 are ordered by at most one $\Phi_{A_i, cp}^*$ set. Taking the union of such sets leads to:

$$\Phi_{cpu}^* = \bigcup_{A_i \in X} \Phi_{A_i, cp}^* \quad (2)$$

Finally, let \succ_{cpu} stand for the order obtained by taking the transitive closure of Φ_{cpu}^* . We say that tuple t_1 *dominates* tuple t_2 (according to the *ceteris paribus union* (cpu) semantics) iff $t_1 \succ_{cpu} t_2$, and that t_1 is *optimal* in a relation $r \subseteq \text{dom}(X)$ if it is undominated in r . A basic result on acyclic CP-nets is that \succ_{cpu} is always a *strict partial order*, thus not only transitive but also asymmetric (thus irreflexive). This guarantees that at least one optimal tuple exists. Further, if the CP-net is complete there is exactly one optimal tuple in $\text{dom}(X)$.

Example 3. Figure 2 shows the *preference graph* for the CP-net in Figure 1, where there is an arc from t_1 to t_2 iff the pair (t_1, t_2) is in Φ_{cpu}^* . Due to the *ceteris paribus* semantics, arcs exist only between tuples that differ in the value of a single attribute. There is a *path* in the graph from t_1 to t_2 iff $t_1 \succ_{cpu} t_2$. Since the CP-net is complete there is one optimal tuple in $\text{dom}(X)$, namely (it, in, low) . \square

For lack of space, here we do not provide details on the proof procedure of CP-nets, needed to check if $t_1 \succ_{cpu} t_2$. It suffices to say that for acyclic CP-nets its complexity can be exponential in the number of attributes, depending on the structure of the G graph and on how CPT 's are specified.

3 Totalitarian Semantics for CP-nets

Although the cpu semantics is adequate in many situations, it is a fact that in most cases a *complete CP-net* is assumed. When preferences are over many attributes and/or

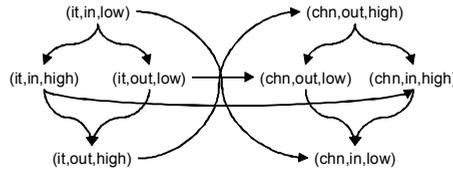
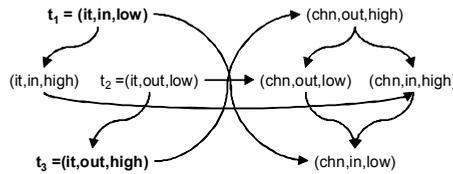


Fig. 2. The \succ_{cpu} order induced over tuples by the CP-net in Figure 1

domains have large cardinalities, it is unrealistic to expect that a user will completely specify all the CPT 's. The effect of having an incomplete CP-net can be seen by referring to our working example. Assume that $CPT(T)$ misses the entry for italian restaurants (i.e., $it : in > out$ is dropped from Figure 1), which is interpreted as “the user has no preference on where to sit”. We are left with the following preference graph:



If the DB relation is *complete*, i.e., $r = \text{dom}(X)$, then the optimal tuples are $t_1 = (it, in, low)$ and $t_2 = (it, out, low)$, which is perfectly reasonable given the absence of preference on where to sit. Assume now that $r = \{t_1 = (it, in, low), t_3 = (it, out, high)\}$. Since $t_1 \not\succeq_{\text{cpu}} t_3$ (there is no path from t_1 to t_3 in the above graph), we conclude that both t_1 and t_3 are optimal in r . We find this quite counter-intuitive, since t_3 has a high price, which contradicts the preference $it : low > high$. Ideally, we would like to have that t_1 dominates t_3 even if the CP-net is incomplete.

We tackle the problem by redefining the semantics of preference statements *and* the way the so-resulting preferences have to be combined. We start with a first version of the *totalitarian* (as opposed to *ceteris paribus*) semantics of statements.

Definition 2 Let $\varphi = p : a_{i,1} > a_{i,2}$ be a statement in $CPT(A_i)$. The strong totalitarian (*st*) interpretation of φ is the set of pairs of tuples:

$$\varphi_{\text{st}}^* = \{((p, a_{i,1}, y), (p, a_{i,2}, y')) \mid y, y' \in \text{dom}(X - P_i - \{A_i\})\} \quad (3)$$

Thus, tuples ordered by φ differ in the value of A_i and, possibly, also in the values of attributes Y_i not involved in φ .

Since the CP-net is locally consistent, the sets $\Phi_{A_i, \text{st}}^* = \bigcup_{\varphi \in CPT(A_i)} \varphi_{\text{st}}^*$ of preferences induced by $CPT(A_i)$ still have no conflicts inside. However, two tuples t_1 and t_2 might be differently ordered by two $\Phi_{A_i, \text{st}}^*$ sets, thus taking their union could introduce cycles in the preference graph. As an example, given $\varphi = it : in > out$ and $\varphi' = it : low > high$ and the tuples $t_1 = (it, in, high)$ and $t_2 = (it, out, low)$, we have that $(t_1, t_2) \in \Phi_{T, \text{st}}^*$ and $(t_2, t_1) \in \Phi_{P, \text{st}}^*$, i.e., a cycle if we take the union of $\Phi_{T, \text{st}}^*$ and $\Phi_{P, \text{st}}^*$.

A way to preserve the strict partial order properties is to compose preferences in the $\Phi_{A_i, \text{st}}^*$ sets using a *Pareto rule*. Intuitively, this is to say that tuple t_1 dominates t_2 iff it does so over at least one attribute and is never the case that this is true also for t_2 .

More precisely, we have that $(t_1, t_2) \in \Phi_{stp}^*$ iff there exists an attribute A_i such that $(t_1, t_2) \in \Phi_{A_i, st}^*$ and for no attribute A_j it is $(t_2, t_1) \in \Phi_{A_j, st}^*$. The *strong totalitarian Pareto (stp)* order \succ_{stp} is then defined as the transitive closure of Φ_{stp}^* .

Theorem 1 For any complete acyclic CP-net N , \succ_{stp} is a strict partial order such that $\succ_{cpu} \subseteq \succ_{stp}$.²

Above theorem shows that the strong totalitarian semantics includes *all* the ceteris paribus preferences. In many cases³ it is also true that all the additional preferences in $\Phi_{stp}^* - \Phi_{cpu}^*$ are in the transitive closure of Φ_{cpu}^* , thus $\succ_{stp} = \succ_{cpu}$. For instance, this happens in our working example on restaurants. However, as the following example shows, this does not hold in general.

Example 4. Consider the CP-net in Figure 3, along with the preference graph of Φ_{cpu}^* (solid arcs). The figure also shows as dashed arcs 3 of the preferences in $\Phi_{stp}^* - \Phi_{cpu}^*$. While the one from (a_1, b_1, c_1) to (a_2, b_2, c_1) , although not in Φ_{cpu}^* is in \succ_{cpu} (there is a path in the Φ_{cpu}^* graph) the other two are *not* derivable using the *cpu* semantics. For instance, consider the pair $(t, t') = ((a_1, b_2, c_1), (a_2, b_2, c_2))$. This is in Φ_{stp}^* since t is better than t' on A , $t[B] = t'[B]$, and on attribute C the two tuples cannot be compared, since they have different parent values $((a_1, b_2)$ and (a_2, b_2) , respectively). \square

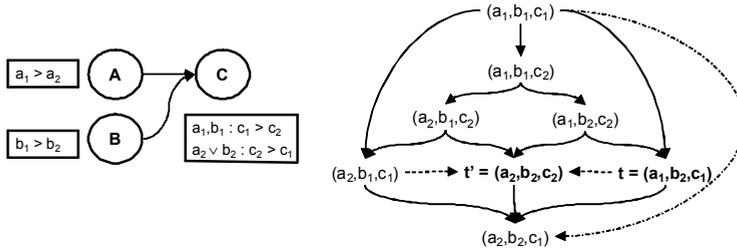


Fig. 3. A CP-net for which the *cpu* and *stp* semantics do not coincide

Is the *stp* semantics a “reasonable” one? We argue that *stp* is not completely exempt from problems, since it is unable to discover some *preference violations*. Refer to tuples t and t' in the above example and consider attribute C . Its *CPT*, written in the figure in a compact form, asserts that if $A = a_2$ or $B = b_2$ then preference is given to c_2 rather than to c_1 . We have $t'[C] = c_2$ and $t[C] = c_1$, thus t' should be better than t on attribute C , yet *stp* is unable to discover it. This motivates the introduction of a new (weak) totalitarian semantics for interpreting the statements in a *CPT*.⁴

² For lack of space, proofs of formal results are omitted.

³ A precise characterization of the CP-nets for which this occurs seems to be a difficult problem, since it depends not only on the net structure, but also on its *CPT*'s.

⁴ Indeed, this new semantics induces more preferences than the strong one from the *CPT*'s. However, the net effect is that *less* preferences among tuples survive after the Pareto composition, as Theorem 2 proves. This is why we say it is “weak”.

Definition 3 (Weak totalitarian Pareto semantics) Let $a_{i,1}, a_{i,2} \in \text{dom}(A_i)$ and t_1 and t_2 be two tuples with $t_1[A_i] = a_{i,1}$ and $t_2[A_i] = a_{i,2}$. Let P_i be the parents of A_i , and $p_1 = t_1[P_i]$, $p_2 = t_2[P_i]$. If $CPT(A_i)$ includes statements (not necessarily distinct) $\varphi_1 = p_1 : a_{i,1} > a_{i,2}$ and $\varphi_2 = p_2 : a_{i,1} > a_{i,2}$ then $(t_1, t_2) \in \Phi_{A_i, \text{wt}}^*$.

The set of all preferences, Φ_{wtp}^* , is the n -ary Pareto composition of the $\Phi_{A_i, \text{wt}}^*$ sets, and the weak totalitarian Pareto (**wtp**) order \succ_{wtp} is the transitive closure of Φ_{wtp}^* .

Consider again Figure 3. In $CPT(C)$ there are two statements (once we write them in extended form), $\varphi_1 = a_1, b_2 : c_2 > c_1$ and $\varphi_2 = a_2, b_2 : c_2 > c_1$, from which we conclude, according to the above definition, that the pair $(t', t) \in \Phi_{C, \text{wt}}^*$. Since $(t, t') \in \Phi_{A, \text{wt}}^*$ still holds, it follows that $(t, t') \notin \Phi_{\text{wtp}}^*$.

Given that we have redefined both statements' interpretation and the preference composition rule, the following is rather surprising:

Theorem 2 For any complete acyclic CP-net N it is $\succ_{\text{cpu}} = \succ_{\text{wtp}}$.

3.1 Incomplete CP-nets

Let us now analyze how **wtp** behaves on *incomplete* nets, which is the most relevant case for the DB scenarios we aim to consider. We start by showing that on incomplete CP-nets the equivalence of the **cpu** and **wtp** semantics breaks down (as required!):

Lemma 1 For any, possibly incomplete, acyclic CP-net N it is $\succ_{\text{cpu}} \subseteq \succ_{\text{wtp}}$.

For instance, in the example at the beginning of this section it is $t_1 = (it, in, low) \succ_{\text{wtp}} t_3 = (it, out, high)$ even if the statement $it : in > out$ has not been specified. This follows since $(t_1, t_3) \in \Phi_{P, \text{wt}}^*$ (t_1 is better on price than t_3) and no preferences over other attributes involve these two tuples. The fact that the optimal tuples in a relation r obtained from a CP-net N under the \succ_{wtp} semantics, denoted as $Opt_{\text{wtp}}(r; N)$, are a subset of those of \succ_{cpu} , $Opt_{\text{cpu}}(r; N)$, is not a case.

Corollary 1. For any acyclic CP-net N and any relation r it is $Opt_{\text{wtp}}(r; N) \subseteq Opt_{\text{cpu}}(r; N)$.

The result immediately follows from \succ_{cpu} being a subset of \succ_{wtp} , and can be refined in the case of complete relations, $r = \text{dom}(X)$.

Theorem 3 For any acyclic CP-net N it is $Opt_{\text{wtp}}(\text{dom}(X); N) = Opt_{\text{cpu}}(\text{dom}(X); N)$.

Besides above properties, how does CP-net incompleteness affect the **wtp** semantics? A first critical observation is that Definition 3 has to be properly extended in order to avoid cycles in the Φ_{wtp}^* graph.

Example 5. Consider the CP-net over attributes RestaurantType (R), Table (T), and SmokingArea (S), $\text{dom}(S) = \{yes, no\}$. As in Example 2, we have $it : in > out$ and $chn : out > in$, but now there is no preference on R (i.e., it and chn are not ordered). Preferences on S are conditional on T : if sitting inside, I do not want to stay in a smoking area ($in : no > yes$), but my preferences change should the table be outside ($out : yes > no$). According to Def. 3, we derive the following cycle of preferences:

- 1) $(it, in, yes) \succ_{\text{wtp}} (it, out, yes)$
- 2) $(it, out, yes) \succ_{\text{wtp}} (chn, out, no)$
- 3) $(chn, out, no) \succ_{\text{wtp}} (chn, in, no)$
- 4) $(chn, in, no) \succ_{\text{wtp}} (it, in, yes)$

Notice that 1) and 3) are also in \succ_{cpu} , whereas this is not the case for 2) and 4). \square

A simple solution to avoid above problem would be to inhibit ordering tuples when they have unordered values in some attributes. This is exactly what the `cpu` semantics would do and, as argued at the beginning of Section 3, is truly unsatisfactory.

The problem of allowing tuples to be ordered even if their attribute values are not completely ordered while, at the same time, preserving the strict partial order properties of \succ_{wtp} , can be solved by: a) slightly revising the notion of what “being better on an attribute” means, and b) limiting the type of incompleteness in the *CPT*’s. We discuss the two issues separately.

Consider first issue a). Referring to preference 2) in Example 5 (similar arguments hold for 4)), we see that the two tuples can be ordered only on *S*. However, looking at attribute *T* we might argue that, *since the parent values are unordered*, one should better consider comparing *(it, out)* and *(chn, out)* as a whole. Under this perspective, it seems natural to say that *(chn, out)* is *better* than *(it, out)*, since the latter does not respect the corresponding statement *it : in > out*. In other terms, when being unable to order parents’s values, one should look at how good is the attribute value under consideration (*out*) within the two different contexts (*it* and *chn*).

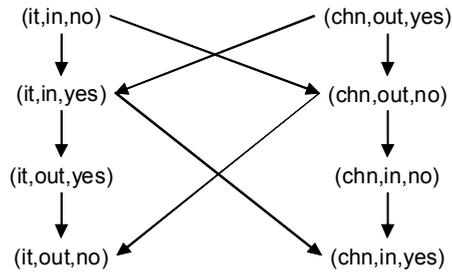
Let us now turn to issue b), i.e., the type of incompleteness in the *CPT*’s, and, for the sake of definiteness, consider first a CP-net in which all attributes have no parents (thus, all preferences are unconditional). In the most “liberal” case, the statements in *CPT*(*A_i*) might induce a generic strict partial order on *dom*(*A_i*). However, it is well known [Cho02] that the Pareto composition of strict partial orders is *not* a strict partial order anymore. As a simple example, if we have attributes *A* and *B*, and statements $a_1 > a_2$, $a_3 > a_4$, $b_2 > b_3$, and $b_4 > b_1$, these would lead to the cycle $(a_1, b_1) \succ_{\text{wtp}} (a_2, b_2) \succ_{\text{wtp}} (a_3, b_3) \succ_{\text{wtp}} (a_4, b_4) \succ_{\text{wtp}} (a_1, b_1)$.

This immediately rules out the possibility of having an uncontrolled amount of incompleteness. On the positive side, if the *CPT*(*A_i*) induce *weak orders*, their Pareto composition *is* a strict partial order. We remind that a weak order is a strict partial order that is also negatively transitive, i.e., for each triple of values *a*, *b*, *c*, if $a \not\succeq b$ and $b \not\succeq c$, then $a \not\succeq c$. Clearly, a total order is also a weak order, but the converse is not necessarily true. More intuitively, a weak order can be viewed as a “linear order with ties”. This is also to say that if $a_{i,1}$ and $a_{i,2}$ are not ordered, and $a_{i,1} > a_{i,3}$, then it should be $a_{i,2} > a_{i,3}$ as well. When attribute *A_i* has parents *P_i*, this restriction applies to each value of *P_i*. Clearly, if $p, p' \in \text{dom}(P_i)$ then the two weak orders they induce on *dom*(*A_i*) need not to be the same. Finally, note that if no statement matching *p* is present in *CPT*(*A_i*), then this induces a weak order in which all values are unordered.

Combining above considerations leads to extend the `wtp` semantics for the case of unordered values so that \succ_{wtp} is always a strict partial order. For lack of space we do not present here the formal definition, rather we show in the following figure the (transitively reduced) preference graph it induces for the CP-net in Example 5.

First, one should observe that tuples with unordered values can still be compared, yet no cycles arise. Second, it is interesting to see that optimal tuples for the two *R* contexts also dominate sub-optimal tuples in the other context (e.g., $(chn, out, yes) \succ_{\text{wtp}} (it, in, yes)$). This is a further evidence that sub-optimal results are excluded when the relation is incomplete. Finally, it can be seen that $(chn, out, no) \not\succeq_{\text{wtp}} (it, out, yes)$, as

expected, since the former is better on T , as explained above, whereas the latter is (still) better on S .



4 Conclusions

In this paper we have considered CP-nets as a viable tool to express user preferences in database queries, and have shown that their strength in compactly representing conditional preferences can be decoupled from the *ceteris paribus* (cpu) semantics. Our results show that one can use an alternative, weak totalitarian (wtp), semantics that overcomes the basic limitation of cpu when preferences are partially specified.

Being this the first work that investigates the use of (incomplete) CP-nets for querying databases, many issues need to be investigated. In particular, we need to develop a complete proof procedure to determine when $t \succ_{\text{wtp}} t'$, which is at the basis of all algorithms for computing the optimal tuples in a relation [Cho02,Kie02,Cia06]. Second, it would be interesting to provide a characterization of optimal tuples in terms of the incompleteness of the CP-net (for the cpu semantics the optimal results of an incomplete CP-net N are just the union of the optimal results of the possible completions of N). Third, we would like to better understand the implications of having an explicit distinction between DB and context attributes, the intuition being that the latter are, for any given query, set to constant values (or to a set of constants).

References

- [BBD⁺04] Craig Boutilier, Ronen I. Brafman, Carmel Domshlak, Holger H. Hoos, and David Poole. CP-nets: A Tool for Representing and Reasoning with Conditional Ceteris Paribus Preference Statements. *Journal of Artificial Intelligence Research (JAIR)*, 21:135–191, 2004.
- [BBHP99] Craig Boutilier, Ronen I. Brafman, Holger H. Hoos, and David Poole. Reasoning With Conditional Ceteris Paribus Preference Statements. In *UAI '99*, 71–80, 1999.
- [Cho02] Jan Chomicki. Querying with Intrinsic Preferences. In *EDBT 2002*, 34–51, 2002.
- [Cia06] Paolo Ciaccia. Processing Preference Queries in Standard Database Systems. In *ADVIS 2006*, 1–12, 2006. Invited paper.
- [GLTW05] Judy Goldsmith, Jérôme Lang, Mirosław Truszczynski, and Nic Wilson. The Computational Complexity of Dominance and Consistency in CP-nets. In *IJCAI-05*, 144–149, 2005.
- [Kie02] Werner Kießling. Foundations of Preferences in Database Systems. In *VLDB 2002*, 311–322, 2002.
- [Wil04] Nic Wilson. Extending CP-Nets with Stronger Conditional Preference Statements. In *AAAI 2004*, 735–741, 2004.

XML Type Projection: A Maximum Flow Approach

Dario Colazzo¹ and Carlo Sartiani²

¹ Laboratoire de Recherche en Informatique (LRI) - Université Paris Sud

² Dipartimento di Informatica - Università di Pisa

Abstract. In the contexts of data integration and data exchange, *schema mappings* are primarily used for query answering. As a consequence, their maintenance and, in particular, the detection of *corrupted* mappings is crucial. Corruption checking can be automatically performed by relying on an operation called *type projection*. This work describes an efficient algorithm for checking XML type projection, based on a characterization of type projection in terms of *type simulation*.

1 Introduction

XML is an universal data format that can be used to represent any kind of data sources. This property made XML a natural medium for integrating heterogeneous sources, in both a centralized fashion [1] and a *decentralized* way [2].

One of the most important problems in data integration systems (both centralized and decentralized) is the *maintenance of mappings*. Mappings are dependencies among schemas, that are used during query answering for reformulating queries or, as in data exchange systems [3], for generating canonical solutions.

Mapping maintenance is a time-consuming and expensive activity, and is usually performed by the system/site administrator, which manually inspects schemas and mappings. In [4] we proposed a mapping maintenance technique based on two key ideas: checking for correctness on both the source and the target schema, and using a type projection notion to make correctness checking on the target schema operational. By checking for correctness on both the source and the target schema, a system based on that technique can capture rules referring to non-existing (or no more existing) fragments of the source schema as well as capture rules that do not *match* the target schema. Type projection refines this *match* in terms of projection, so to capture the intuition of mapping as transformation + projection. Since each check on the target schema implies a type projection check, a crucial and key issue of this technique is the availability of an efficient algorithm for type projection checking.

In this paper we study a type projection operation for *unordered* XML types. In particular, we provide: a formal definition of type projection; a characterization of type projection in terms of *type simulation*; and, an efficient algorithm for checking type projection. All these contributions are relevant: from a theoretical point of view, the characterization of type projection in terms of type simulation allows for a better understanding of the properties of type projection; from a practical point of view, the algorithm makes the technique described in [4] a viable option for automatic mapping maintenance in p2p data integration systems.

As final remark, the solution proposed in this paper can be used in both p2p and centralized data integration systems, as well as in data exchange systems.

2 Motivation

We defined type projection to formalize the (very practical) problem of checking whether a mapping from a schema S_i into a schema S_j respects the structural properties of S_j . To this aim, we embraced the Piazza [5] characterization of schema mappings as “transformation + projection”. This characterization is common to most data integration and data exchange systems, and points out that a schema mapping is a *non-functional* transformation from a source schema S to a target schema T . Indeed, as a transformed data instance I_T may not contain all attributes/elements specified by the target schema T , I_T may not be an instance of T . The following example clarifies this issue.

Example 1. Consider a p2p data sharing system for music information. The system allows users to share data about their (legally owned) music files, so to discover information about their preferred songs and singers. Assume that a user in Cupertino publishes her music database according to the following schema (described in the XDUCE [6] type language).

```
CupMDB = mySongs[(Song)*]
Song = song[Title, Artist, Album, MyRating]
Title = title[String]
Artist = artist[String]
Album = album[String]
MyRating = myRating[Integer]
```

This schema groups data by song, and, for each song, represents the title, the artist name, the album title, as well as personal rating information.

Suppose now that another user in Seattle publishes her database according to the following (different) schema.

```
SeattleMDB = musicDB[Artist*]
Artist = artist[Name, Provenance, Track*]
Name = name[String]
Provenance = provenance[Continent, Country]
Continent = continent[String]
Country = country[String]
Track = track[Title, Year, Genre]
Title = title[String]
Year = year[Integer]
Genre = genre[String]
```

This schema groups data by artist and, for each artist, details her name and provenance, as well as the list of corresponding tracks.

To make these databases interact together, a proper schema mapping is required, as schemas nest data in very different ways. Assume that the user in Cupertino employs the following mapping (a set of XQuery-like queries) to map her schema into the Seattle-based schema.

```
SeattleMDB <-
Q1($input): for $t in $input/title return $t
Q2($input): for $a in $input//artist,
            return artist[ name[$a/data()],
                          for $s in $input//song,
                          $art in $s/artist
                          where $art/data() = $a/data()
                          return track[Q1($s)]]
Q3($input): for $db in /mySongs return musicDB[Q2($db)]
```

This mapping transforms data conforming to a fragment of the Cupertino schema (`album` and `myRating` elements are discarded) into data conforming to a fraction of the Seattle-based schema. This is a very common situation in data integration systems, as usually only a fraction of semantically related heterogeneous schemas can be reconciled. In particular, as the Seattle user schema does not support `album` and `myRating` elements, they must be ignored in the mapping. Furthermore, since the Cupertino schema does not provide information about song genre, corresponding elements are not generated, hence any transformed data instance must be regarded as a projection of a Seattle-compliant data instance.

In the following Sections we will provide a formalization of type projection for a wide class of schema mappings: we will regard a mapping as a set of rules that transform a source data instance $I_S : \mathcal{S}_i$ into a *fragment* of one or more data instances I_T conforming to \mathcal{S}_j .

While in [4] we grounded on an XQuery-like mapping language, in the spirit of Piazza, in this work we will extend our approach to any mapping language, like that of [3], provided that an output type for any given mapping can be inferred.

3 Data Model and Type Language

We represent an XML document as an *unranked, unordered*, node-labeled tree, as shown by the following grammar.

$$f ::= () \mid b \mid l[f] \mid f, \dots, f$$

f is a forest that may comprise the empty sequence $()$, base values (b), element nodes ($l[f]$), as well as the concatenation of other forests (f, \dots, f).

Since our study started from a p2p perspective, we drop ordering from the model, as no global order can be enforced on data coming from multiple sources. Hence, concatenation (f, \dots, f) is commutative, associative, and has $()$ as neutral element. On data model instances we define the following *value projection* relation.

Definition 1 (Value projection). *The value projection relation \lesssim is the minimal relation such that:*

$$\begin{array}{ll} () \lesssim f & f_1, f_2 \lesssim f_3, f_4 \quad \text{if } (f_1 \lesssim f_3 \wedge f_2 \lesssim f_4) \\ b_1 \lesssim b_2 & f_1 \lesssim f_3 \quad \text{if } \exists f_2 : f_1 \lesssim f_2 \wedge f_2 \lesssim f_3 \\ f \lesssim f, () & l[f_1] \lesssim l[f_2] \quad \text{if } f_1 \lesssim f_2 \\ f_1, f_2 \lesssim f_2, f_1 & \end{array}$$

\lesssim is an *injective* simulation relation among values, inspired by the projection operator of the relation data model. Intuitively, $d_1 \lesssim d_2$ if there exists a subterm d_3 in d_2 such that d_3 matches d_1 ; this is very close (up to simulation) to the relational projection, where $r_1 = \pi_A r_2$ if r_1 is equal to the fragment of r_2 obtained by discarding non- A attributes.

Our type language, based on XDuce [6] and XQuery [7] type languages, is shown below:

$$\mathbf{Types} \quad T ::= () \mid B \mid l[T] \mid T, T \mid T \mid T \mid T^*$$

$()$ is the type for the empty sequence value, B denotes the type for base values (without loss of generality, we only consider string base values), types T, U and $T \mid U$ are, respectively, product and union types, and, finally, T^* is the type for repetition. Types are unordered, as no global order on XML data dispersed on multiple sources

can be established: this aspect significantly increases the hardness of comparing two XML types, as usual heuristics and optimizations based on type ordering cannot be applied in this context.

The semantics of types is standard: as usual, $\llbracket _ \rrbracket$ is the minimal function from types to sets of forests that satisfies the following monotone equations:

$$\begin{aligned} \llbracket () \rrbracket &\triangleq \{()\} & \llbracket T_1 \mid T_2 \rrbracket &\triangleq \llbracket T_1 \rrbracket \cup \llbracket T_2 \rrbracket & \llbracket B \rrbracket &\triangleq \{b \mid b \text{ is a base value}\} \\ \llbracket T_1, T_2 \rrbracket &\triangleq \{f_1, f_2 \mid f_i \in \llbracket T_i \rrbracket\} & \llbracket l[T] \rrbracket &\triangleq \{l[f] \mid f \in \llbracket T \rrbracket\} & \llbracket T^* \rrbracket &\triangleq \llbracket T \rrbracket^* \end{aligned}$$

Subtyping is defined via type semantics: $T < U \iff \llbracket T \rrbracket \subseteq \llbracket U \rrbracket$.

4 Type Projection

Type projection is a generalization to types of the value projection relation, as shown by Definition 2.

Definition 2 (Type projection). *Given two types T_1 and T_2 , we say that T_1 is a projection of T_2 ($T_1 \lesssim T_2$) if and only if: $\forall d_1 : T_1 \exists d_2 : T_2. d_1 \lesssim d_2$.*

As for the value projection relation, the type projection relation is semantics, and states that a type T_1 is a projection of a type T_2 if, for each data instance d_1 conforming to T_1 , there exists a data instance d_2 conforming to T_2 such that d_1 is a projection of d_2 . Notice that this formalization perfectly fits the “transformation + projection” characterization we informally gave in Section 2. The decidability of type projection comes from the main result proved in [4], that relates type projection checking and subtype-checking over *unordered* types, whose decidability has been proved in [8].

5 Type Simulation

Type simulation is a *symbolic* relation among type equivalence classes (i.e., types are identified modulo commutativity, associativity, and $()$ -neutrality), whose main aim is to provide a convenient way to characterize and check for type projection. Type simulation is defined among types in *disjunctive normal form*, i.e., types where products are distributed across unions. A type T can be normalized by applying the normalization function $norm(T)$, defined as shown in Table 5.1.

Table 5.1. $norm()$ function.

$norm(())$	$\triangleq ()$
$norm(B)$	$\triangleq B$
$norm(l[T])$	$\triangleq \begin{cases} \cup l[A_i] & \text{if } norm(T) = A_1 \mid \dots \mid A_n \\ l[norm(T)] & \text{otherwise} \end{cases}$
$norm(T \mid U)$	$\triangleq norm(T) \mid norm(U)$
$norm(T^*, U^*, U)$	$\triangleq norm((T' \mid U')^*, U)$
$norm(T, U)$	$\triangleq \begin{cases} norm(A_1, U) \mid norm(A_2, U) & \text{if } norm(T) = (A_1 \mid A_2) \\ norm(T, A_1) \mid norm(T, A_2) & \text{if } norm(U) = (A_1 \mid A_2) \\ norm(T), norm(U) & \text{otherwise} \end{cases}$
$norm(T^*)$	$\triangleq norm(T)^*$

$norm()$ works by transforming types, while preserving their semantics (Lemma 1), so that the transformed types can be easily compared by the simulation relation (and by the corresponding algorithm). To eliminate some ambiguity, the rules of $norm()$ must be applied in the order in which they are defined. $norm()$ is essentially equivalent to the distribution of \wedge/\vee , hence its computational complexity is in EXPTIME. Despite this upper bound, for a vast class of types $norm()$ can be computed in PTIME. This class contains types where unions are always guarded by a $*$ -operator ($*$ -guarded types). Proving that for $*$ -guarded types $norm()$ is polynomial is straightforward. $*$ -guardedness is a property enjoyed by a large number of commonly used DTDs and XML schemas. The following lemma proves that $norm()$ preserves the semantics of types.

Lemma 1. For each type T , $\llbracket T \rrbracket = \llbracket norm(T) \rrbracket$.

Definition 3 (Type simulation). The type simulation relation \preceq among normalized types is defined as follows.

- 1) $B \preceq B$
- 2) $() \preceq U$
- 3) $l[T] \preceq l[U]$ if $T \preceq U$
- 4) $T_1 \preceq U_2, U_3$ if $T_1 \preceq U_2 \vee T_1 \preceq U_3$
- 5) $T_1, T_2 \preceq U_3, U_4$ if $(T_1 \preceq U_3 \wedge T_2 \preceq U_4)$
- 6) $T_1 \preceq U_2 \mid U_3$ if $T_1 \preceq U_2 \vee T_1 \preceq U_3$ and $T_1 \neq V_1 \mid V_2$
- 7) $T_1 \mid T_2 \preceq U$ if $T_1 \preceq U \wedge T_2 \preceq U$
- 8) $T \preceq U^*$ if $T \preceq U$
- 9) $T^* \preceq U^*$ if $T \preceq U^*$
- 10) $T_1, T_2 \preceq U^*$ if $T_1 \preceq U^* \wedge T_2 \preceq U^*$

We briefly discuss non-straightforward rules. Rule 5 shows that simulation between product types is *injective*, hence capturing the injective nature of projection: for instance, $T = Album, Album$ cannot be projected into $U = Album$, as data conforming to T have two distinct `album` elements, while data conforming to U have only one `album` element. Injectivity may be broken by repetition types, or when sequence types are in the immediate scope of a repetition type. Rules 6-7 describe the simulation for union types. These rules pinpoint the commutative and *non-injective* nature of union types.

The following theorem shows the equivalence between type projection and type simulation.

Theorem 1. Given two normalized types T and U : $T \lesssim U \Leftrightarrow T \preceq U$.

6 Type Projection Checking

The equivalence between type projection and type simulation allows for the construction of an *efficient, simulation-based* projection-checking algorithm. The algorithm is actually a *not-so-naive* implementation of the type simulation rules. Indeed, a naive implementation of these rules would lead to a super-exponential algorithm. The super-exponential complexity of a naive algorithm comes from two key factors. First of all, a recursive comparison of two types T_1 and T_2 , as suggested by the simulation rules, would lead to many *backtracking* operations, in particular when comparing union or product types. This problem can be solved by *flattening* T_1 and T_2 , and by constructing a memoization *type matrix* (*simTypes* in our algorithm).

The second key factor that makes naive algorithms super-exponential is the comparison among product types. The type simulation definition states that, if outside the

immediate scope of a repetition type, $Z_1, \dots, Z_n \preceq V_1, \dots, V_m$ if and only if each type Z_i can be mapped into a distinct type in V_1, \dots, V_m , so that there do not exist Z_i and Z_j ($i \neq j$) such that Z_i and Z_j are mapped into the same type term V_h . This problem can be naively solved by generating all possible assignments of Z_i to V_j types and by choosing an injective one: this can be done in super-exponential time, as the possible assignments are $O(\binom{m}{n})$.

An alternative solution for the comparison of product types can be obtained by observing that this problem is equivalent to a 0-1 maximum flow problem on bipartite graphs. Indeed, one can build a bipartite graph \mathcal{G} , whose first partition \mathcal{P}_1 contains one node per each Z_i type, and whose second partition \mathcal{P}_2 contains one node per each V_j type; nodes in \mathcal{P}_1 are connected to a source s , while nodes in \mathcal{P}_2 are connected to a sink t . \mathcal{P}_1 and \mathcal{P}_2 are connected together through edges satisfying the simulation relation, i.e., an edge from Z_i to V_j is inserted in \mathcal{G} if $Z_i \preceq V_j$. Each edge has two possible values for its flow: 0 and 1. The source s emits a flows of n units, so Z_1, \dots, Z_n simulates V_1, \dots, V_m if and only if a flow of n units reaches the sink t . This can be determined by using a 0 – 1 maximum flow algorithm on bipartite graph, whose complexity is $O((n + m)^3)$ [9].

6.1 Type Simulation Algorithm

Our algorithm for type simulation checking ($\text{SIM}(T_1, T_2)$) is shown in Figure 1. It consists of three main phases. During Phase 1, the algorithm creates and populates a type matrix *simTypes* with boolean values or symbolic references, both obtained by repeatedly calling the SIMPLESIM algorithm of Figure 1. The matrix has as many rows as the type terms in T_1 , and as many columns as type terms in T_2 . To ensure the proper behavior of the algorithm, type terms from both T_1 and T_2 have to be extracted with a pre-order visit, so that, if Z_i and Z_j are type terms in T_1 and $i < j$, then Z_i precedes Z_j in the pre-order visit of T_1 . The SIMPLESIM algorithm is called once per each cell in the type matrix, with the notable exception of those cells that correspond to types whose match is useless as they occupy incompatible positions in the type term hierarchy (this task is performed by the boolean function COMPARABLE).

SIMPLESIM returns a boolean value for any comparison that does not require any further type comparisons. If, instead, the comparison between T and U requires a further comparison, as in the type simulation rule for $l[Z] \preceq l[W]$ (Rule 3), then the algorithm returns a simple symbolic reference ($\text{ref}(U_x, V_y)$), a logical combination of simple references (e.g., $\bigwedge_{i=1}^n \text{ref}(U_i, T_2)$), or a product reference ($\text{ref}(\{U_1, \dots, U_n\} \otimes \{V_1, \dots, V_m\})$). A simple reference $\text{ref}(U_x, V_y)$ is a symbolic pointer to the content of the cell *simTypes*[x][y], while a product reference indicates that the maximum flow algorithm must be executed on top of a bipartite graph built from the partitions $\{U_1, \dots, U_n\}$ and $\{V_1, \dots, V_m\}$. References are left unevaluated, and they will be solved during Phase 2.

The output of Phase 1, thus, is a partially instantiated type matrix. This matrix is the input for Phase 2, whose objective is to solve symbolic references. By visiting the matrix in reverse order, starting from the bottom right and going right to left, the algorithm proceeds by replacing references of the form $\text{ref}(U_x, V_y)$ with the content of *simType*[x][y], by evaluating logical combinators, and by applying the maximum flow algorithm for \otimes references; in the latter case, an auxiliary (and trivial) function GRAPH-CONSTR is invoked with the aim of building the bipartite graph \mathcal{G} , while the maximum flow is computed with a standard maximum flow algorithm MAXIMUMFLOW. The result of this phase is a fully instantiated type matrix, since, when a cell *simTypes*[i][j] has been reached, all the cells that can be referenced by its content have already been

visited and instantiated. It should be observed that, as prescribed by Rule 10 of the simulation relation, nodes in the second partition of \mathcal{G} , corresponding to *-types in the right hand-side of the comparison, are marked as *special*, since they can be used to map more than one term of the left hand-side; from a flow point of view, this means the edges connecting these nodes with the sink of the graph have unbounded capacity.

Phase 3 is very simple and consists in returning a boolean value describing the result of the whole simulation. Since the types being compared correspond to the first row and to the first column of the type matrix, the algorithm just returns the content of $simTypes[1][1]$.

We can now state the correctness and completeness of SIM wrt type simulation.

Theorem 2. *The SIM algorithm is correct and complete wrt the type simulation relation.*

Theorem 3. *The worst case complexity of the SIM algorithm, while comparing T and U , is $O(nm(n+m)^3)$, where n is the number of terms in T , and m is the number of terms in U .*

7 Conclusions

In this paper we presented an efficient algorithm for projection-checking among XML types. We first characterized type projection in terms of type simulation, and, then, used the type simulation rules to define a checking algorithm. A novel technique for comparing product types, based on a 0 – 1 maximum flow approach, allowed for designing a correct and complete algorithm with polynomial time complexity on normalized types.

The type projection-checking algorithm forms the basis of the mapping maintenance technique for p2p data integration systems presented in [4].

References

1. Ives, Z.G., Halevy, A.Y., Weld, D.S.: Integrating network-bound XML data. *IEEE Data Eng. Bull.* **24** (2001) 20–26
2. Halevy, A.Y., Ives, Z.G., Madhavan, J., Mork, P., Suci, D., Tatarinov, I.: The Piazza peer data management system. *IEEE Trans. Knowl. Data Eng.* **16** (2004) 787–798
3. Arenas, M., Libkin, L.: Xml data exchange: Consistency and query answering. In: *PODS*. (2005)
4. Colazzo, D., Sartiani, C.: Mapping Maintenance in XML P2P Databases. In: *Proceedings of the 10th International Symposium on Data Bases and Programming Languages - DBPL 2005, Trondheim, Norway, August 28-29, 2005*. (2005)
5. Halevy, A.Y., Ives, Z.G., Mork, P., Tatarinov, I.: Piazza: data management infrastructure for semantic web applications. In: *Proceedings of the Twelfth International World Wide Web Conference, WWW2003, Budapest, Hungary, 20-24 May 2003*, ACM (2003) 556–567
6. Hosoya, H., Pierce, B.C.: Xduce: A statically typed XML processing language. *ACM Trans. Internet Techn.* **3** (2003) 117–148
7. Draper, D., Fankhauser, P., Fernandez, M., Malhotra, A., Rose, K., Rys, M., Siméon, J., Wadler, P.: XQuery 1.0 and XPath 2.0 Formal Semantics. Technical report, World Wide Web Consortium (2007) W3C Recommendation.
8. Dal-Zilio, S., Lugiez, D., Meysonnier, C.: A logic you can count on. In Jones, N.D., Leroy, X., eds.: *POPL*, ACM (2004) 135–146
9. Goldberg, A.V.: Recent developments in maximum flow algorithms. In: *SWAT: Scandinavian Workshop on Algorithm Theory*. (1998)

```

SIMPLESIM(Type  $T_1$ , Type  $T_2$ )
1  switch
2    case  $T_1 = T_2$  :
3      return true
4    case  $T_1 = ()$  :
5      return true
6    case  $T_1 = l[U_1] \wedge T_2 = l[U_2]$  :
7      return  $ref(U_1, U_2)$ 
8    case  $T_1 = U_1, \dots, U_n \wedge T_2 = V_1, \dots, V_m$  :
9      return  $ref(\{U_1, \dots, U_n\} \otimes \{V_1, \dots, V_m\})$ 
10   case  $T_1 = U_1, \dots, U_n \wedge T_2 = V_1 \mid \dots \mid V_m$  :
11     return  $(\bigvee_{j=1}^m ref(T_1, V_j))$ 
12   case  $T_1 = U_1 \mid \dots \mid U_n \wedge T_2 = V_1, \dots, V_m$  :
13     return  $(\bigwedge_{i=1}^n ref(U_i, T_2))$ 
14   case  $T_1 = U * \wedge T_2 = V *$  :
15     return  $ref(U, T_2)$ 
16   case  $T_1 = U_1, \dots, U_n \wedge T_2 = V *$  :
17     return  $(\bigwedge_{i=1}^n ref(U_i, T_2))$ 
18   case  $T_1 = U \wedge T_2 = V *$  :
19     return  $ref(U, V)$ 
20   case default :
21     return false

SIM(Type  $T_1$ , Type  $T_2$ )
1  // we assume  $T_1$  to be composed by  $n$  terms and  $T_2$  by  $m$  terms
2  // phase 1: type matrix construction
3  Array $[n][m]$  simTypes
4  for each  $U_i$  in  $T_1$ 
5    do for each  $V_j$  in  $T_2$ 
6      do if COMPARABLE( $U_i, V_j$ )
7        then simType $[i][j] = SIMPLESIM(U_i, V_j)$ 
8  // phase 2: reference resolution
9  for  $i \leftarrow n$  to 1
10 do for  $j \leftarrow m$  to 1
11   do if simTypes $[i][j] = ref(Z_p, U_q) \wedge simTypes[p][q] \in \{\text{true}, \text{false}\}$ 
12     then simTypes $[i][j] = simTypes[p][q]$ 
13   else if simTypes $[i][j]$  contains references different from  $\otimes$ 
14     then for each  $ref(Z_x, V_y)$  in simTypes $[i][j]$ 
15       do replace  $ref(U_x, V_y)$  with simTypes $[x][y]$ 
16       evaluate the logical expression in simTypes $[i][j]$ 
17   else if simTypes $[i][j] = ref(\{U_f, \dots, U_p\} \otimes \{V_g, \dots, V_q\})$ 
18     then  $\mathcal{P}_1 = \{U_f, \dots, U_p\}$ 
19          $\mathcal{P}_2 = \{V_g, \dots, V_q\}$ 
20     mark as special  $\mathcal{P}_2$  nodes corresponding to *-types
21      $G = \text{GRAPHCONSTR}(\mathcal{P}_1, \mathcal{P}_2)$ 
22     simTypes $[i][j] = \text{MAXIMUMFLOW}(G)$ 
23 // phase 3: result discovery
24 return simTypes $[1][1]$ 

```

Fig. 1. Type simulation algorithm.

Advanced OLAP Visualization of Multidimensional Data Cubes: A Semantics-driven Compression Approach

Alfredo Cuzzocrea¹, Vincenzo Russo¹, Domenico Sacca^{1,2}, Paolo Serafino¹

¹ Department of Electronics, Computer Science, and Systems
University of Calabria, I-87036 Cosenza, Italy
{cuzzocrea, russo, sacca, serafino}@si.deis.unical.it

² Institute of High Performance Computing and Networks
Italian National Research Council, I-87036 Cosenza, Italy
sacca@icar.cnr.it

Abstract. A novel semantics-driven compression technique for efficiently supporting advanced OLAP visualization of multidimensional data cubes is presented in this paper. The main contribution of our work consists in the amenity of using the data compression paradigm as a way of visualizing multidimensional OLAP domains, in order to overcome the natural disorientation and refractoriness of human beings in dealing with hyper-spaces. Experimental results conducted on several kinds of synthetic data sets clearly confirm the effectiveness and the efficiency of our technique, also in comparison with state-of-the-art proposals.

1 Introduction

Three relevant challenges of OLAP systems [10] have captured a lot of attention during the last years: (i) the *data querying* problem, which concerns with how data are accessed and queried to support summarized knowledge extraction from massive data cubes; (ii) the *data modeling* problem, which concerns with how data are represented and, thus, processed inside OLAP servers (e.g., during query evaluation); (iii) the *data visualization* problem, which concerns with how data are presented to OLAP users and decision makers in Data Warehouse environments. Indeed, research communities have mainly studied and investigated the first two problems, whereas the last one, even if important-with-practical-applications, has been very often neglected.

Approximate Query Answering (AQA) techniques address the first challenge, and can be reasonably considered as one of the most important topics in OLAP research. The main proposal of AQA techniques consists in providing approximate answers to resource-consuming OLAP queries (e.g., *range-queries* [12]) instead of computing exact answers, as decimal precision is usually negligible in OLAP query and report activities (e.g., see [7]). Due to a relevant interest from the Data Warehousing research community, AQA techniques have been intensively investigated during the last years, also achieving important results. Among the others, *histograms* (e.g., [2,4,11]), *wavelets* ([18]), and *sampling* (e.g., [9]) are the most successful techniques, and they have also inducted several applications in even different contexts than OLAP. *Conceptual data models* for OLAP are widely recognized as based on data cube concepts like *dimension*, *hierarchy*, *level*, *member*, and *measure*, first introduced by Gray *et al.* [10], which inspired various models for multidimensional databases and data cubes (e.g., [17]). Nevertheless, despite this effort, recently, several papers have put in evidence some formal limitations of accepted conceptual models for OLAP (e.g., [5]), or theoretical failures of popular data cube operations, like aggregation functions (e.g., [14,15]). Contrarily to data querying and modeling issues, since *data presentation models* do not properly belong to the well-founded conceptual-logical-physical hierarchy for relational database models (which has also been inherited from multidimensional models), the problem of OLAP data visualization has been studied and investigated only so far (e.g., [16]). On the other hand, being essentially OLAP a technology to support decision making, thus based on (sensitive) information exploration and browsing, it is easy to understand that, in future years, tools for advanced visualization of multidimensional data cubes will rapidly conquest the OLAP research scene.

Starting from the fundamentals of data cube compression and OLAP data visualization research issues, in this paper we argue to meaningfully exploit the main results coming from the first one and the goals of the second one in a combined manner, and propose *a novel technique for supporting advanced OLAP visualization of multidimensional data cubes*. The basic motivation of such an approach is realizing that (i) compressing data is an (efficient) way of visualizing data, and (ii) this intuition is well-founded at large (i.e., for any data-intensive system relying on massive data repositories), and, more specifically, it is particularly targeted to the OLAP context where accessing multidimensional data cubes can become a realistic bottleneck for Data Warehousing systems and applications. Briefly, our proposed technique relies on two steps. The first one consists in generating a two-dimensional OLAP view D from the input multidimensional data cube A by means of an innovative approach that allows us to *flatten OLAP dimensions* (of A), and, as a consequence, effectively support exploration and browsing activities against A (via D), by overcoming the natural disorientation and refractoriness of human beings in dealing with hyper-spaces. Particularly, the (two) OLAP dimensions on which D is defined are built from the dimensions of A according to the analysis goals of the target OLAP user/application. The second step consists in generating a bucket-based compressed representation of D named as *Hierarchy-driven Indexed Quad-Tree Summary* (H-IQTS), and denoted by $H\text{-IQTS}(D)$, which meaningfully extends the compression technique for two-dimensional summary data domains presented in [3], by introducing the amenity of *generating semantics-aware buckets*, i.e. buckets that “follow” groups of the OLAP hierarchies of D . In other words, *we use the OLAP hierarchies defined on the dimensions of D to drive the compression process*. The latter step allows us to achieve space efficiency, while, at the same time, supporting approximate query answering and advanced OLAP visualization features. Similarly to [3], $H\text{-IQTS}(D)$ is shaped as a quad-tree (thus, each “internal” bucket in $H\text{-IQTS}(D)$ has four child sub-buckets), and the information stored in its buckets is still the sum of the items contained within them. The technique we propose in this paper can be successfully applied to all those scenarios in which accessing and exploring massive multidimensional data cubes is a critical requirement. For instance, this is the case of *mobile OLAP systems and applications* (e.g., the system *Hand-OLAP* proposed by us in [8]), where users access corporate OLAP servers via handheld devices, which are usually characterized by specific properties (e.g., small storage space, small size of the display screen, discontinuance of the connection to the WLAN etc) that are often incompatible with the need of browsing and querying wireless-interconnected multidimensional data cubes.

2 Basic Definitions

In order to better understand our proposal, here we introduce some basic definitions regarding the constructs of OLAP conceptual data model we adopt, along with the notation we use in the rest of the paper. These definitions are compatible with main results of previous popular models (e.g., [10]).

Given an OLAP dimension d_i , and its domain of members $\Psi(d_i)$, each of them denoted by ρ_j , a hierarchy defined on d_i , denoted by $H(d_i)$, can be represented as a general tree (i.e., such that each node of the tree has a number $N \geq 0$ of child nodes) built on top of $\Psi(d_i)$. $H(d_i)$ is usually built according to a *bottom-up strategy* by (i) setting as leaf nodes of $H(d_i)$ members in $\Psi(d_i)$, and (ii) iteratively aggregating sets of members in $\Psi(d_i)$ to obtain other (internal) members, each of them denoted by σ_j , (the latter are internal nodes in $H(d_i)$). In turn, internal members in $H(d_i)$ can be further aggregated to form other super-members until a unique aggregation of members is obtained; the latter corresponds to the root node of $H(d_i)$, and it is known-in-literature as the *aggregation ALL*. Each member in $H(d_i)$ is characterized by a level (of the hierarchy), denoted by L_j ; as a consequence, we can define a level L_j in $H(d_i)$ as a *collection of members*. For each level L_j , the ordering of L_j , denoted by $O(L_j)$, is the one exposed by the OLAP server platform for the target data cube on which $H(d_i)$ is defined.

Given a multidimensional data cube A such that $Dim(A) = \{d_0, d_1, \dots, d_{n-1}\}$ is the set of dimensions of A and $Hie(A) = \{H(d_0), H(d_1), \dots, H(d_{n-1})\}$ the set of hierarchies defined on A , the collection of members σ_j at level $L_j \geq 0$ (note that, when $L_j = 0$, $\sigma_j \equiv \rho_j$) of each hierarchy $H(d_i)$ in $Hie(A)$ univocally refers, in a multidimensional fashion, a certain (OLAP) data cell C_p in A at level L_j (in other words, C_p is the OLAP aggregation of data cells in A at level L_j). We name such collection as j -level OLAP Metadata (for C_p), and denote it as $J-M(C_p)$.

Given a member σ_j at level L_j of $H(d_i)$ and the set of its child nodes $Child(\sigma_j)$, which are members at level L_{j+1} , we define as the *Left Boundary Member* (LBM) of σ_j the child node of σ_j in $Child(\sigma_j)$ that is the *first* in the ordering $O(L_{j+1})$. Analogously, we define as the *Right Boundary Member* (RBM) of σ_j the child node of σ_j in $Child(\sigma_j)$ that is the *last* in the ordering $O(L_{j+1})$.

3 OLAP Dimension Flattening

The OLAP dimension flattening process is the first step of our technique for supporting advanced OLAP visualization of multidimensional data cubes. In more detail, we flatten dimensions of the input multidimensional data cube A into two specialized dimensions called *Visualization Dimensions* (VDs). VDs support advanced OLAP visualization of A via constructing an ad-hoc two-dimensional OLAP view D defined just on VDs.

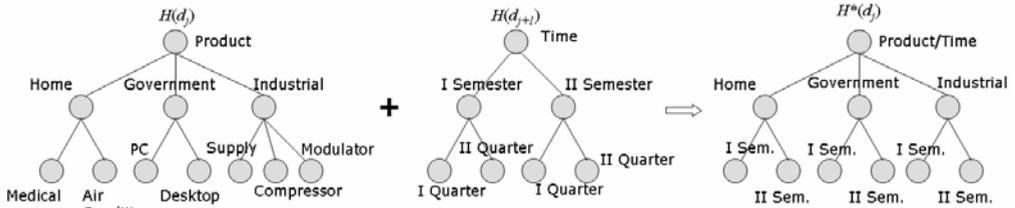


Fig. 1. Merging OLAP hierarchies

The process that allows us to obtain the two VDs from the dimensions of A works as follows. Letting $Dim(A)$ and $Hie(A)$ be the set of dimensions and the set of hierarchies of A , respectively, each VD is a tuple $v_i = \langle d_i, H^*(d_i) \rangle$ such that (i) d_i is the dimension selected by the target OLAP user/application, (ii) $H^*(d_i)$ is a hierarchy built from meaningfully merging the “original” hierarchy $H(d_i)$ of d_i with the hierarchies of other dimensions in A according to an *ordered definition set* $MD(v_i) = \{\langle HL_i, d_j, P_j \rangle, \langle HL_j, d_{j+1}, P_{j+1} \rangle, \dots, \langle HL_{j+K-1}, d_{j+K}, P_{j+K} \rangle\}$, where $K = |MD(v_i)| - 1$. The ordering is established by the user/OLAP-application that edits the overall process for extracting the two-dimensional view from the target data cube. In more detail, for each pair of consecutive tuples $\langle \langle HL_j, d_{j+1}, P_{j+1} \rangle, \langle HL_{j+1}, d_{j+2}, P_{j+2} \rangle \rangle$ in $MD(v_i)$, the sub-tree of $H(d_{j+2})$ rooted in the root node of $H(d_{j+2})$ and having depth equal to P_{j+2} , denoted by $H_S^{P_{j+2}}(d_{j+2})$, is merged to $H(d_{j+1})$ by appending a *clone* of it to *each* member σ_{j+1} at level HL_{j+1} , named as *hooking level*, in $H(d_{j+1})$. In other words, each item in $MD(v_i)$ represents a combination of members from the original dimensional hierarchies. From the described approach, it follows that: (i) the ordering of items in $MD(v_i)$ defines the way of building $H^*(d_i)$; (ii) the first hierarchy to be processed is just $H(d_i)$. As an example of the flattening process of two OLAP dimensions into a new one, consider Fig. 1, where the hierarchy $H^*(d_j)$ is obtained by merging $H(d_{j+1})$ to $H(d_j)$ via setting $P_{j+1} = 1$ and $HL_j = 1$.

4 Hierarchy-Driven Compression of Two-Dimensional OLAP Views

Compressing the two-dimensional OLAP view D (extracted from A according to the OLAP dimension flattening process presented in Sect. 3) is the second step of our proposed technique. Given D , for each step j of our compression algorithm, we need to (1) greedily select the leaf bucket b of $H-IQTS(D)$ having maximum *Sum of the Squared Errors* (SSE) [3], (2) split b in four

sub-buckets through investigating, for each dimension d_k of D , levels of the hierarchy $H(d_k)$. The task (1) is similar to what proposed in [3] for two-dimensional summary data domains, whereas the novelty proposed in this paper consists in the task (2).

Formally, given the current bucket $b_j = D[l_{j,0}:u_{j,0}][l_{j,1}:u_{j,1}]$ to be split at step j of our compression algorithm, such that $[l_{j,k}:u_{j,k}]$ is the range of b_j on the dimension d_k of D , the problem is finding, for each dimension d_k of D , a *splitting position* $S_{j,k}$ belonging to $[l_{j,k}:u_{j,k}]$. To this end, for each dimension d_k of D , our splitting strategy aims at (i) grouping items into buckets related to the *same semantic domain*, and (ii) maintaining *as more balanced as possible* the hierarchy $H(d_k)$. Specifically, the first aspect lets the benefits highlighted in Sect. 1; the second aspect allows us to sensitively improve query estimation capabilities as, on the basis of this amenity, we finally obtain buckets with balanced “numerousness” (of items) that introduce a smaller approximation error in the evaluation of (OLAP) queries involving several buckets rather than the contrary case (see [3] for further investigations).

4.1 A Hierarchy-Driven Algorithm for Compressing Two-Dimensional OLAP Views. For the sake of simplicity, we will present our hierarchy-driven compression algorithm for two-dimensional OLAP views through showing how to handle the hierarchy of an OLAP dimension d_k (i.e., how to determine a splitting position $S_{j,k}$ on d_k). Obviously, this technique must be performed for both the dimensions of the target (two-dimensional) OLAP view D , thus obtaining, for each pair of splits at step j of our algorithm (i.e., $S_{j,0}$ and $S_{j,1}$), four two-dimensional buckets to be added to the current partition of D .

Let $b_j = D[l_{j,0}:u_{j,0}][l_{j,1}:u_{j,1}]$ be the current bucket to be split at step j . Consider the range $[l_{j,k}:u_{j,k}]$ of b_j on the dimension d_k of D . To determine $S_{j,k}$ on $[l_{j,k}:u_{j,k}]$, we denote as $T_{j,k}(l_{j,k}:u_{j,k})$ the sub-tree of $H(d_k)$ whose (i) leaf nodes are the members of the sets $0-M(C_w)$ defined on data cells C_w in $D[l_{j,k}:u_{j,k}]$ (i.e., the one-dimensional bucket obtained by projecting b_j with respect to the dimension d_k), and (ii) the root node is the (singleton) member of the set $P_k-M(C_r)$ defined on the data cell C_r that is the (singleton) aggregation of $D[l_{j,k}:u_{j,k}]$ at level L_P of $H(d_k)$, being P_k the depth of $H(d_k)$ (and also the depth of $T_{j,k}(l_{j,k}:u_{j,k})$).

Formally, let (i) d_k be the dimension of D to be processed; (ii) $H(d_k)$ the hierarchy defined on d_k ; (iii) $b_j = D[l_{j,k}:u_{j,k}]$ the current (one-dimensional) bucket to be split at step j of our algorithm; (iv) $T_{j,k}(l_{j,k}:u_{j,k})$ the tree related to b_j . In order to select the splitting position $S_{j,k}$ on $[l_{j,k}:u_{j,k}]$, letting $T_{j,k}^1(l_{j,k}:u_{j,k})$ be the *second level* of $T_{j,k}(l_{j,k}:u_{j,k})$, we initially consider the data cell C_k in $D[l_{j,k}:u_{j,k}]$ whose indexer is in the middle of $D[l_{j,k}:u_{j,k}]$, denoted by $X_{j,D} = \left\lfloor \frac{1}{2} \cdot |D[l_{j,k}:u_{j,k}]| \right\rfloor$. It should be

noted that processing the second level of $T_{j,k}(l_{j,k}:u_{j,k})$ (i.e., $T_{j,k}^1(l_{j,k}:u_{j,k})$) derives from the use of the aggregation ALL in OLAP conceptual models, which, in total, introduces an *additional level* in the general tree modeling an OLAP hierarchy.

Then, starting from ρ_k , being ρ_k the (singleton – see Sect. 2) member in the set $0-M(C_k)$, we go up on $H(d_k)$ until the parent of ρ_k at level $T_{j,k}^1(l_{j,k}:u_{j,k})$, denoted by σ_k , is reached, and we decide how to determine $S_{j,k}$ on the basis of the nature of σ_k . If σ_k is the LBM of the root node of $T_{j,k}(l_{j,k}:u_{j,k})$, denoted by $R_{j,k}$, then $S_{j,k} = \left\lfloor \frac{1}{2} \cdot |D[l_{j,k}:u_{j,k}]| \right\rfloor - 1$ and, as a consequence, we obtain the

following two (one-dimensional) buckets as child buckets of b_j : $b'_{j+1} = D \left[l_{j,k} : \left\lfloor \frac{1}{2} \cdot |D[l_{j,k}:u_{j,k}]| \right\rfloor - 1 \right]$ and $b''_{j+1} = D \left[\left\lfloor \frac{1}{2} \cdot |D[l_{j,k}:u_{j,k}]| \right\rfloor : u_{j,k} \right]$. Otherwise, if σ_k is

the RBM of $R_{j,k}$, then $S_{j,k} = \left\lfloor \frac{1}{2} \cdot |D[l_{j,k}:u_{j,k}]| \right\rfloor$ and, as a consequence,

$b'_{j+1} = D \left[l_{j,k} : \left\lfloor \frac{1}{2} \cdot |D[l_{j,k} : u_{j,k}]| \right\rfloor \right]$ and $b''_{j+1} = D \left[\left\lfloor \frac{1}{2} \cdot |D[l_{j,k} : u_{j,k}]| \right\rfloor + 1 : u_{j,k} \right]$. Finally, if σ_k is different from both the LBM and the RBM of $R_{j,k}$, i.e. it follows the LBM of $R_{j,k}$ in the ordering $O(T^1_{j,k}(l_{j,k}:u_{j,k}))$ and precedes the RBM of $R_{j,k}$ in the ordering $O(T^1_{j,k}(l_{j,k}:u_{j,k}))$, we perform a finite number of shift operations on the indexers of $D[l_{j,k}:u_{j,k}]$ starting from the middle indexer $X_{j,D}$ and within the range $\Gamma_{j,k} = [\Gamma_{j,k}^{lo} : \Gamma_{j,k}^{up}]$, such that $\Gamma_{j,k}^{lo} = \left\lfloor \frac{1}{2} \cdot |D[l_{j,k} : u_{j,k}]| \right\rfloor - \left\lfloor \frac{1}{3} \cdot |D[l_{j,k} : u_{j,k}]| \right\rfloor$ and $\Gamma_{j,k}^{up} = \left\lfloor \frac{1}{2} \cdot |D[l_{j,k} : u_{j,k}]| \right\rfloor + \left\lfloor \frac{1}{3} \cdot |D[l_{j,k} : u_{j,k}]| \right\rfloor$, until a data cell V_k in $D[l_{j,k}:u_{j,k}]$ such that the corresponding member σ_k at level $T^1_{j,k}(l_{j,k}:u_{j,k})$ is the LBM or the RBM of $R_{j,k}$, if exists, is found. It should be noted that admitting a maximum offset of $\pm \left\lfloor \frac{1}{3} \cdot |D[l_{j,k} : u_{j,k}]| \right\rfloor$ with respect to the middle of the current bucket is coherent with the aim of maintaining as more balanced as possible the hierarchy $H(d_k)$, which allows us to take advantages from the above-highlighted benefits (see Sect. 4).

To this end, starting from the middle of $\Gamma_{j,k}$ (which is equal to the one of $D[l_{j,k}:u_{j,k}]$, $X_{j,D}$), we iteratively consider indexers $l_{j,q}$ within $\Gamma_{j,k}$ on the basis of the following function: $l_{j,q} = \{X_{j,D} \text{ if } q = 0; l_{j,q-1} + (-1)^q \cdot q \text{ if } q > 1\}$. If such data cell V_k exists, then $S_{j,k}$ is set as equal to the so-determined indexer $I^*_{j,q}$, and, as a consequence, we obtain the pairs of buckets $b'_{j+1} = D[l_{j,k} : I^*_{j,q} - 1]$ and $b''_{j+1} = D[I^*_{j,q} : u_{j,k}]$ if $I^*_{j,q}$ is the LBM of $R_{j,k}$, or, alternatively, the pairs of buckets $b'_{j+1} = D[l_{j,k} : I^*_{j,q}]$ and $b''_{j+1} = D[I^*_{j,q} + 1 : u_{j,k}]$ if $I^*_{j,q}$ is the RBM of $R_{j,k}$. On the contrary, if such data cell V_k does not exist, then we do not perform any split on $D[l_{j,k}:u_{j,k}]$, and we “remand” the splitting at next step of the algorithm (i.e., $j + 1$) where the splitting position $S_{j+1,k}$ is determined by processing the third level $T^2_{j+1,k}(l_{j+1,k}:u_{j+1,k})$ of the tree $T_{j+1,k}(l_{j+1,k}:u_{j+1,k})$ (i.e., by decreasing the aggregation level of OLAP data with respect to the previous step). The latter approach is iteratively repeated until a data cell V_k verifying the above condition is found; otherwise, if the leaf level of $T_{j,k}(l_{j,k}:u_{j,k})$ is reached without finding any admissible splitting point, then $D[l_{j,k}:u_{j,k}]$ is added to the current partition of the OLAP view without being split. We point out that this way to do still pursues the aim of obtaining balanced partitions of the input OLAP view.

5 Experimental Study

5.1 Definitions and Metrics. In order to test the effectiveness of our proposed technique, we introduce two kinds of experiments. The first one is oriented to probe the data cube *compression performance* (or, equally, the *accuracy*) of our technique, whereas the second one is instead oriented to probe the *visualization capabilities* of our technique in meaningfully supporting advanced OLAP visualization of multidimensional data cubes.

As regards the data layer of our experimental framework, we engineered two kinds of synthetic two-dimensional OLAP views (which, in turn, have been extracted from synthetic multidimensional data cubes via a random flattening process): (i) the view $D_C(L_1, L_2)$, for which data are uniformly distributed on a given range $[L_1, L_2]$ (i.e., the well-known *Continuous Values Assumption* (CVA) [6] holds), and (ii) the view $D_Z(z_{min}, z_{max})$, for which data are distributed according to a Zipf distribution whose parameter z is randomly chosen on a given range $[z_{min}, z_{max}]$.

As regards the outcomes of our study, we define the following metrics. For the first kind of experiments (i.e., that focused on the accuracy), given a population of synthetic range-SUM queries Q_S , we measure the *Average Relative Error* (ARE) between exact and approximate answers to

queries in Q_S , i.e. $\bar{E}_{rel} = \frac{1}{|Q_S|} \cdot \sum_{k=0}^{|Q_S|-1} E_{rel}(Q_k)$, such that, for each query Q_k in Q_S ,

$$E_{rel}(Q_k) = \frac{|A(Q_k) - \tilde{A}(Q_k)|}{|A(Q_k)|}, \text{ where (i) } A(Q_k) \text{ is the exact answer to } Q_k, \text{ and (ii) } \tilde{A}(Q_k) \text{ is the}$$

approximate answer to Q_k . Particularly, fixing a range sizes Δ_k for each dimension d_k of the target synthetic OLAP view D , we generate queries in Q_S through spanning D by means of the “seed” $\Delta_0 \times \Delta_1$ query Q^s .

For the second kind of experiments, we have been inspired from the *Hierarchical Range Queries* (HRQ) introduced by Koudas *et al.* in [13]. In our implementation, a HRQ $Q_H(W_H, P_H)$ is a full tree such that: (i) the depth of such tree is equal to P_H ; (ii) each internal node N_i has a fan-out degree equal to W_H ; (iii) each node N_i stores the definition of a (“traditional”) range-SUM query Q_i ; (iv) for each node N_i in $Q_H(W_H, P_H)$, there not exists any sibling node N_j of N_i such that $Q_i \cap Q_j \subsetneq \emptyset$. Similarly to the previous kind of experiments, for each node N_i in $Q_H(W_H, P_H)$, the population of queries $Q_{S,i}$ to be used as input query set was generated by means of the above-described spanning technique. It should be noted that, due the nature of HRQs, the selectivity of seed queries $Q_{i,k}^s$ of nodes N_i at level k of $Q_H(W_H, P_H)$ must decreases as the depth P_k of $Q_H(W_H, P_H)$ increases.

In consequence of this, letting γ be an input parameter and $\|D\|$ the selectivity of the target OLAP view D , we first impose that the selectivity of the seed query of the root node N_0 in $Q_H(W_H, P_H)$, denoted by $\|Q_{0,0}^s\|$, is equal to the γ % of $\|D\|$, and then, for each internal node N_i in $Q_H(W_H, P_H)$ at level k , we randomly determine the seed queries of the child nodes of N_i by checking the follow-

ing constraint: $\sum_{i=0}^{\lfloor (W_H)^{k+1} - 1 \rfloor} \|Q_{i,k+1}^s\| \leq \|Q_{i,k}^s\|$ and $Q_{i,k+1}^s \cap Q_{j,k+1}^s = \emptyset$ for each i and j in $[0, \lfloor (W_H)^{k+1} - 1 \rfloor]$, with $i \triangleleft j$, and adopting the criterion of maximizing each $\|Q_{i,k}^s\|$.

Given a HRB $Q_H(W_H, P_H)$, we measure the *Average Accessed Bucket Number* (AABN), which models the average number of buckets accessed during the evaluation of $Q_H(W_H, P_H)$, and it is defined as follows:

$$AABN(Q_H(W_H, P_H)) = \sum_{k=0}^{P_H} \frac{1}{(W_H)^k} \cdot \sum_{\ell=0}^{\lfloor (W_H)^k - 1 \rfloor} AABN(N_\ell), \text{ where, in turn,}$$

$AABN(N_\ell)$ is the average number of buckets accessed during the evaluation of the population of queries $Q_{S,\ell}$ of the node N_ℓ in $Q_H(W_H, P_H)$, i.e. $AABN(N_\ell) = \frac{1}{|Q_{S,\ell}|} \cdot \sum_{k=0}^{|Q_{S,\ell}|-1} ABN(Q_k)$, such that for

each query Q_k in $Q_{S,\ell}$, $ABN(Q_k)$ is the number of buckets accessed during the evaluation of Q_k .

5.2 Experimental Results. In our experimental study, we compared the performances of our proposed technique (under the two metrics defined above) against the following well-known histogram-based techniques for compressing data cubes: *MinSkew* by Acharya *et al.* [2], *GenHist* by Gunopulos *et al.* [11], and *STHoles* by Bruno *et al.* [4]. In more detail, having fixed the space budget G (i.e., the storage space available for housing the compressed representation of the input OLAP view), we derived, for each comparison technique, the configuration of the input parameters that the respective authors consider the best in their papers. This ensures a *fair* experimental analysis, i.e. an analysis such that each comparison technique provides its *best* performance.

For all the comparison techniques, letting r be the parametric compression ratio and $size(D)$ the total occupancy of D , we set the space budget G as equal to the r % of $size(D)$. For instance, $r = 10$ (i.e., G is equal to the 10 % of $size(D)$) is widely-recognized as a reasonable setting (e.g., see [4]). Fig. 2 shows experimental results for what regards the accuracy of the compression techniques on the 1.000×1.000 two-dimensional OLAP views $D_C(25,70)$ (left side) and $D_Z(0.5,1.5)$

(right side), respectively, and with respect to r ranging on the interval [5:20] (i.e., G on the interval [5:20] % of $size(D)$). The selectivity of input queries has been fixed to the value $\|Q\| = 350 \times 300$. This kind of experiments allows us to measure the *scalability* of the compression techniques, which is a critical aspect in OLAP systems (e.g., see [7]). Fig. 3 shows experimental results for what regards the “visualization capabilities” of the comparison techniques (according to the guidelines drawn throughout the paper) with respect to the depth of HRQs (i.e., P_H) having fan-out degree W_H equal to 5 and the parameter γ equal to 70 %.

From Fig. 2 and Fig. 3, it follows that, with respect to the accuracy metrics, our proposed technique is comparable with *MinSkew*, which represents the best on two-dimensional views (indeed, as well-recognized-in-literature, *MinSkew* presents severe limitations on multidimensional domains); instead, with respect to the visualization metrics, our proposed technique overcomes the comparison techniques, thus confirming its suitability in efficiently supporting advanced OLAP visualization of multidimensional data cubes.

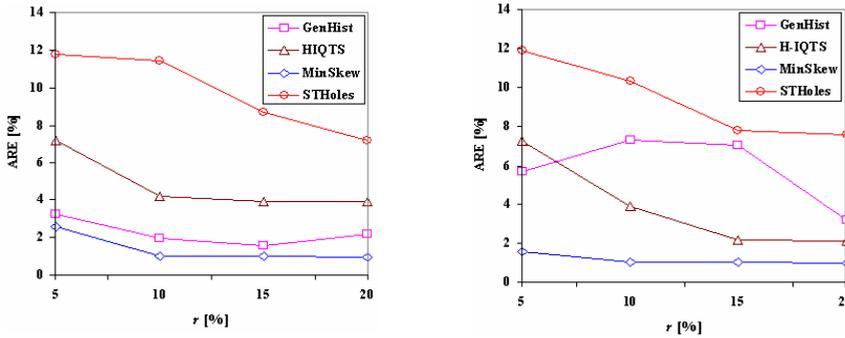


Fig. 2. Experimental results for the accuracy metrics with respect to the compression ratio r on the 1.000×1.000 two-dimensional OLAP views $D_C(25,70)$ (left side) and $D_X(0.5,1.5)$ (right side) with $\|Q\| = 350 \times 300$

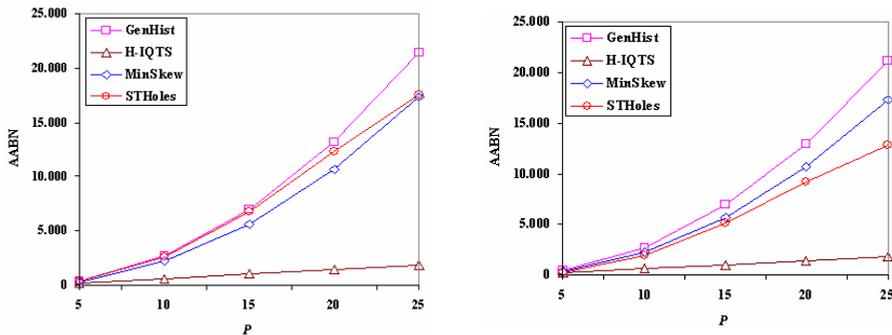


Fig. 3. Experimental results for the visualization metrics with respect to the depth of HRQs P on the 1.000×1.000 two-dimensional OLAP views $D_C(25,70)$ (left side) and $D_X(0.5,1.5)$ (right side) with $W_H = 5$, $r = 10$, and $\gamma = 70$ %

6 Conclusions and Future Work

In this paper, we have presented an innovative technique for supporting advanced OLAP visualization of multidimensional data cubes, which is particularly suitable for mobile OLAP scenarios (like, for instance, those addressed by the system Hand-OLAP [8]). Founding on very efficient two-dimensional summary data domain compression solutions [3], our technique meaningfully

exploits the data compression paradigm that, in this paper, has been proposed as a way of visualizing multidimensional OLAP domains to overcome the natural disorientation and refractoriness of human beings in dealing with hyper-spaces. In this direction, the OLAP dimension flattening process and the amenity of computing semantics-aware buckets are, to the best of our knowledge, innovative contributions to the state-of-the-art OLAP research. Finally, various experimental results performed on different kinds of synthetic two-dimensional OLAP views extracted from (synthetic) multidimensional data cubes have clearly confirmed the benefits of our proposed technique in the OLAP visualization context, also in comparison with well-known data cube compression techniques. Future work is mainly focused on making the proposed technique capable of building m -dimensional OLAP views over massive n -dimensional data cubes, with $m \ll n$ and $m > 2$, by extending the algorithms presented in this paper. A possible solution could be found in the results coming from the *High-dimensional Data and Information Visualization* research area (e.g., see [1]), which are already suitable to be applied to the problem of visualizing multidimensional databases and data cubes.

References

1. 2D, 3D and High-dimensional Data and Information Visualization research group. University of Hannover (2005) available at http://www.iwi.uni-hannover.de/lv/seminar_ss05/bartke/home.htm
2. Acharya, S., Poosala, V., Ramaswamy, S.: Selectivity Estimation in Spatial Databases. Proc. of ACM SIGMOD (1999) 13-24
3. Buccafurri, F., Furfaro, F., Saccà, D., Sirangelo, C.: A Quad-Tree Based Multiresolution Approach for Two-Dimensional Summary Data. Proc. of IEEE SSDBM (2003) 127-140
4. Bruno, N., Chaudhuri, S., Gravano, L.: STHoles: A Multidimensional Workload-Aware Histogram. Proc. of ACM SIGMOD (2001) 211-222
5. Cabibbo, L., Torlone, R.: From a Procedural to a Visual Query Language for OLAP. Proc. of IEEE SSDBM (1998) 74-83
6. Colliat, G.: OLAP, Relational, and Multidimensional Database Systems. SIGMOD Record, Vol. 25, No. 3 (1996) 64-69
7. Cuzzocrea, A.: Overcoming Limitations of Approximate Query Answering in OLAP. Proc. of IEEE IDEAS (2005) 200-209
8. Cuzzocrea, A., Furfaro, F., Saccà, D.: Hand-OLAP: A System for Delivering OLAP Services on Handheld Devices. Proc. IEEE ISADS (2003) 80-87
9. Gibbons, P.B., Matias, Y.: New Sampling-Based Summary Statistics for Improving Approximate Query Answers. Proc. of ACM SIGMOD (1998) 331-342
10. Gray, J., Chaudhuri, S., Bosworth, A., Layman, A., Reichart, D., Venkatrao, M.: Data Cube: A Relational Aggregation Operator Generalizing Group-By, Cross-Tab, and Sub-Totals. Data Mining and Knowledge Discovery, Vol. 1, No. 1 (1997) 29-53
11. Gunopulos, D., Kollios, G., Tsotras, V.J., Domeniconi, C.: Approximating Multi-Dimensional Aggregate Range Queries over Real Attributes. Proc. of ACM SIGMOD (2000) 463-474
12. Ho, C.-T., Agrawal, R., Megiddo, N., Srikant, R.: Range Queries in OLAP Data Cubes. Proc. of ACM SIGMOD (1997) 73-88
13. Koudas N., Muthukrishnan S., Srivastava D.: Optimal Histograms for Hierarchical Range Queries. Proc. of ACM PODS (2000) 196-204
14. Lehner, W., Albrecht, J., Wedekind, H.: Normal Forms for Multivariate Databases. Proc. of IEEE SSDBM (1998) 63-72
15. Lenz, H.-J., Thalheim, B.: OLAP Databases and Aggregation Functions. In Proc. of IEEE SSDBM (2001) 91-100
16. Maniatis, A., Vassiliadis, P., Skiadopoulos, S., Vassiliou, Y.: CPM: A Cube Presentation Model for OLAP. Proc. of DaWaK (2003) 4-13
17. Vassiliadis, P.: Modeling Multidimensional Databases, Cubes and Cube Operations. Proc. of IEEE SSDBM (1998) 53-62
18. Vitter, J.S., Wang, M., Iyer, B.: Data Cube Approximation and Histograms via Wavelets. Proc. of ACM CIKM (1998) 96-104

Constraint Hardness for Modelling, Matching and Ranking Semantic Web Services

Claudia d'Amato

Department of Computer Science, University of Bari - Italy
claudia.damato@di.uniba.it

Abstract. In the real life scenario, human requests for a service are characterized by aspects that are strictly necessary for the requester and aspects that could not be satisfied. Optional aspects are relaxed by hand when found results do not fulfill the requester. Based on this aspect, a framework for modeling Semantic Web Services is proposed. It is based on Description Logic and it allows for expressing constraints in service descriptions of different strengths, i.e. *Hard* and *Soft Constraints*. Hence it showed as such constraints can be effectively used during the matching process and to obtain a ranking of the provided services satisfying the request. The ranking method ranks (matched) service descriptions on the grounds of their semantic similarity w.r.t. the service request, by preferring those that are able to better satisfy both Hard and Soft Constraints.

1 Introduction

The main strength of Web Service (WS) technology is to allow uniform access, via Web standards, to software components residing on various platforms and written in various programming languages. Their major limitation is that their retrieval and composition still require manual effort. To solve this problem, WS have been enriched by semantic description of their functionality[8, 10].

In this paper we propose a framework for describing services, based on Description Logic (DL) [1]. DL is the theoretical foundation of OWL¹ language and it is endowed by a formal semantics, thus allowing expressive service descriptions. Moreover the service discovery task can be performed by algorithms defined in terms of standard and non-standard DL inferences. The use of DL in service descriptions and discovery task is not new [7, 4, 9, 13, 5, 2]. We propose a framework that enriches the guidelines of [5] allowing to express *Hard Constraints (HC)* and *Soft Constraints (SC)* of a service description and most of all of a service request. Specifically, we call *HC* those features that have to be necessarily satisfied by the target services while we call *SC* those features whose satisfaction is only preferable.

To be able to distinguish *HC* and *SC* is important both for *business-to-consumer* interaction and for service discovery task. For the former, *HC* and *SC* allow to express the real necessities of the user; for the latter, the distinction between *HC* and *SC* makes possible to relax some needs, increasing the possibility of satisfying a request.

¹ The ontology language for the Semantic Web, <http://www.w3.org/2004/OWL/>.

Several works have focused such aspect exploiting different approaches. In [6] an approach to hybrid semantic Web service matching that complements logic based reasoning with approximate matching based on syntactic similarity computations is presented. In [12] *SC* are managed by considering numerical preferences. Here, we propose a totally semantic based framework for expressing *HC* and *SC* in which every kind of preferences can be expressed. We propose also a new procedure for ranking the services (discovered in the previous phase), that is able to manage the variance introduced by *HC* and *SC*. The procedure uses a semantic similarity measure for DL concept descriptions that assigns higher ranks to services that are more similar to the request and that satisfy both its *HC* and *SC*, while services that are less similar and/or satisfy only *HC* receive a low rank.

The framework for describing services is presented in Sect. 2. In Sect. 3 the discovery and ranking processes are detailed. Conclusions of the work are drawn in Sect. 4.

2 Service Descriptions by the use of Description Logics

A *service description* is expressed as a set of constraints that have to be satisfied by the service providers. It can be thought as an abstract class acting as a template for *service instances*; namely, a service description defines a space of possible service instances (as in [11]), thus introducing *variance*.

Variance is the phenomenon of having more than one instance and/or more than one interpretation for a service description. Following [5], we distinguish between *variance due to intended diversity* and *variance due to incomplete knowledge*. To explain these concepts, the notion of *possible worlds* (borrowed from the first-order logic semantics) is used. Under open-world semantics, a modeler must explicitly state which service instances are not covered by the service description. For each aspect of the service description that is not fully specified there are *several possible worlds*, reflecting a way of resolving incompleteness (*variance due to incomplete knowledge*). Besides, given a possible world, the lack of constraints possibly allows for many instances satisfying a service description (*variance due to intended diversity*).

In order to cope with the effects of the *variance* on the semantics of a service description, it is necessary to adopt a language characterized by well-defined semantics. This is one of the peculiarities of the DL family. We intend to enrich the DL-based framework for describing services presented in [5], introducing the notions of *HC* and *SC* for a service description. The framework is reported below.

- A service description is expressed by a set of DL-axioms $D = \{S, \phi_1, \phi_2, \dots, \phi_n\}$, where the axioms ϕ_i impose restrictions on an atomic concept S , which represents the service to be performed.
- Domain-specific background knowledge is represented by a *knowledge base* (KB) that contains all relevant domain-level facts.
- A *possible world*, resolving incomplete knowledge issues, is represented by a single DL model (interpretation) I of $KB \sqcup D$.
- The service instances that are acceptable w.r.t. a service description D , are the individuals in the extension S^I of the concept S representing the service.

- Variance due to intended diversity is given by S^I containing different individuals.
- Variance due to incomplete knowledge is reflected by $KB \sqcup D$ having several models I_1, I_2, \dots .

The axioms in a service description D constrain the set of acceptable service instances in S^I . These constraints are generally referred to the properties used in a description. Here, various ways for constraining a property using DL are reported.

Variety: a property can either be restricted to a fixed value or it can range over instances of a certain class. This is expressed by $\forall r.i$ (or $\exists r.i$) and $\forall r.C$ (or $\exists r.C$), respectively. For any acceptable service instance, the value of such a property must either be an individual or a member of a class.

Multiplicity: a property can either be multi-valued, allowing service instances with several property values, or single-valued, requiring service instances to have at most one value for the property. By the number restriction $\leq 1 r$, a property is marked as single-valued. Using the restrictions $\leq m r$ (with $m \geq 2$) $\geq n r$, $\exists r.\top$, $\exists r.C$, and $\forall r.C$ a property is marked as multi-valued.

Coverage: a property can be explicitly known to cover a range. If a property is *range-covering*, the service description enforces that in every possible world, there is an acceptable service instance with this property value. This introduces variance due to intended diversity. This kind of constraint is expressed by an axiom of the form $C \sqsubseteq \exists r^{\neg}.S$ in D , where the concept C is the range of the property r to be covered. A non-range-covering property induces variance due to incomplete knowledge, as in distinct possible worlds different subsets of the range will be covered.

Example 2.1. Let us consider the following requested service description

$$D_r = \{ S_r \equiv \text{Company} \sqcap \exists \text{payment}.\text{EPayment} \sqcap \exists \text{to}.\{\text{bari}\} \sqcap \\ \sqcap \exists \text{from}.\{\text{cologne,hahn}\} \sqcap \leq 1 \text{hasAlliance} \sqcap \\ \sqcap \forall \text{hasFidelityCard}.\{\text{milesAndMore}\}; \\ \{\text{cologne,hahn}\} \sqsubseteq \exists \text{from}^{\neg}.S_r \}$$

$$KB = \{ \text{cologne:Germany, hahn:Germany, bari:Italy, milesAndMore:Card} \}$$

The requester asks for companies that fly from Cologne and Hahn to Bari and accept electronic payment for tickets. Then it is required that a company has at most one alliance with another flight company and, if it has a fidelity program, it is "Miles and More". In the description, several kinds of constraints are reported. *Variety* constraints are used with the properties *to*, *from* and *hasFidelityCard*, that are restricted to a fixed value. The *at-most* number restriction (≤ 1) for the property *hasAlliance* is a *Multiplicity* constraint with which the property is declared to be single-value. A *Coverage* constraint is expressed by the last axiom in D_r which makes explicit the range covered by the property *from* that is $\{\text{cologne,hahn}\}$.

The service presented in the example represent a simple description. In real scenarios a service request is typically characterized by some needs that *must* be necessarily satisfied and others that *should* be satisfied (preferences). The former will be considered as *Hard Constraints (HC)* and the latter as *Soft Constraints (SC)*. Taking this difference into account makes the service description and management more complex.

A possible way to express such constraints can be to describe a service in terms of two different sets: an *HC* set and a *SC* set, whose elements are expressed as seen above. More formally, let $D_r^{HC} = \{S_r^{HC}, \sigma_1^{HC}, \dots, \sigma_n^{HC}\}$ be the set of *HC* for a request D_r and let $D_r^{SC} = \{S_r^{SC}, \sigma_1^{SC}, \dots, \sigma_m^{SC}\}$ be the set of *SC* for D_r . The complete description of D_r is given by $D_r = \{S_r \equiv S_r^{HC} \sqcup S_r^{SC}, \sigma_1^{HC}, \dots, \sigma_n^{HC}, \sigma_1^{SC}, \dots, \sigma_m^{SC}\}$.

Example 2.2. Let us consider a slightly modified version of the previous example distinguishing between *HC* and *SC*:

$$D_r = \{ S_r \equiv \text{Flight} \sqcap \exists \text{from}.\{\text{Cologne, Hahn, Frankfurt}\} \sqcap \exists \text{to}.\{\text{Bari}\} \sqcap \\ \sqcap \forall \text{hasFidelityCard}.\{\text{MilesAndMore}\}; \\ \{\text{Cologne, Hahn, Frankfurt}\} \sqsubseteq \exists \text{from}^- .S_r; \quad \{\text{Bari}\} \sqsubseteq \exists \text{to}^- .S_r \}$$

where

$$HC_r = \{ \text{Flight} \sqcap \exists \text{to}.\{\text{Bari}\} \sqcap \exists \text{from}.\{\text{Cologne, Hahn, Frankfurt}\}; \\ \{\text{Cologne, Hahn, Frankfurt}\} \sqsubseteq \exists \text{from}^- .S_r; \quad \{\text{Bari}\} \sqsubseteq \exists \text{to}^- .S_r \}$$

$$SC_r = \{ \text{Flight} \sqcap \forall \text{hasFidelityCard}.\{\text{MilesAndMore}\}; \}$$

With this service description a requester asks for flights starting from Frankfurt or Cologne or Hahn and arriving at Bari. The use of "Miles And More" card would be preferred. Departure and arrival places are expressed as *HC*. This means that provided services must fulfil these constraints. Instead the use of "Miles And More" card is expressed as *SC*, namely flights that allow the use of this card are preferred, but the requester accepts also flights that do not allow the use of this card.

This new representation can better model the requester's needs, allowing to express real-life preferences, feature not considered in the original framework [5]. Moreover expressing *SC* allows to have service instances satisfying a request even if part of it is ignored, thus augmenting the possibility of having response for a request.

3 Discovery and Ranking Provided Service Descriptions

Service Discovery is the task of locating service providers that can satisfy the requester's needs. In this scenario, semantic service descriptions can be used to automatize the task. Discovery is performed by matching a requested service description to the service descriptions of potential providers, in order to detect relevant ones.

Let D_r and D_p respectively a requested service description and a provided service description, expressed as a set of axioms imposing restrictions on the services that have to be performed: S_r and S_p respectively (see Sect. 2). The matching process (w.r.t. a KB) can be defined as a boolean function $match(KB, D_r, D_p)$ which specifies how to apply DL inferences to perform matching. Various matching procedure, based on DL inferences, have been proposed [7, 13, 11]. We fix our attention to those proposed in [5] where a valid match occurs when there exists a common instance service between a provider's service description D_p and a requestor's service description D_r w.r.t. KB , in *every possible world*. It can be formalized as:

$$KB \cup D_r \cup D_p \models \exists x S_r(x) \wedge S_p(x) \Leftrightarrow KB \cup D_r \cup D_p \cup \{S_r \sqcap S_p \sqsubseteq \perp\} \text{unsatisfiable}$$

This check is called *Entailment of Concept Non-Disjointness*. Considering the framework in Sect. 2, the matching can be performed first of all by considering S_r^{HC} , thus obtaining all the services able to satisfy the *HC* of the request. Then, in order to find services able to satisfy both *HC* and *SC* of the request, the matching can be performed again by considering $S_r^{HC} \sqcap S_r^{SC}$. The services selected by the matching have to be ranked w.r.t. certain criteria and then returned to the requestor. We propose a ranking procedure grounded on a semantic similarity measure for DL (presented in the follow).

3.1 The semantic similarity measure

Service descriptions (see Sect. 2) can be regarded as DL concept descriptions asserted in a *T-Box* and their instances as concept assertions in an *A-Box*. The *Canonical Interpretation* can be considered (see [1]). Hence the following similarity measure [3] can be used as follows:

$$s(S_r, S_p) = \frac{|(S_r \sqcap S_p)^{\mathcal{I}}|}{|(S_r \sqcup S_p)^{\mathcal{I}}|} \cdot \max\left(\frac{|(S_r \sqcap S_p)^{\mathcal{I}}|}{|S_r^{\mathcal{I}}|}, \frac{|(S_r \sqcap S_p)^{\mathcal{I}}|}{|S_p^{\mathcal{I}}|}\right)$$

where $(\cdot)^{\mathcal{I}}$ computes the concept extension w.r.t. \mathcal{I} , $|\cdot|$ returns the cardinality of a set.

This function assigns the maximum value in case of semantic equivalence of the service descriptions. Otherwise it assigns a value in the range $[0, 1[$. This value grows with the increasing of the amount of the shared service instances. In order to compute the similarity between a requested service description S_r and a provided service description S_p , their extensions have to be computed. For every provided service, the set of instances is known. We need to define the extension of S_r . Note that the measure is applied after the matching process and that the chosen matching procedure (see Sect.3) selects all provided services S_p that have at least one instance satisfying S_r . So the set of instances for S_r is given by the union of the provided service instances that satisfy S_r . Namely: $S_r^{\mathcal{I}} = \bigcup_{j=1}^n \{x | S_r(x) \wedge S_p^j(x)\}$ where n is the number of provided services selected by the matching process.

3.2 Ranking Procedure

The goal of the ranking process is to put in higher positions, provided services that are most similar to the request and that satisfy both *HC* and *SC*. Instead, services that are less similar and/or satisfy only *HC* are ranked in lower positions.

The rationale of the procedure consists in measuring the similarity between the requested and the provided services, selected by the matching phase: a higher similarity will result in higher rankings. The presented measure assigns highest values to services that share most of the instances with S_r so, as in [5], the criterion used is based on *variance*, namely, a provider is better than another if the variance it provides is greater than the other. However, differently from [5], we are able to supply a total order of the provided services (rather than a partial order). Anyway this is not enough for ensuring that provided services satisfying both *HC* and *SC* of S_r will be in the higher positions. Let us consider the following scenario: let S_r be the requested service and let S_p^l and S_p^k two provided services, selected by the matching procedure. Let us suppose that S_r is described by both *HC* and *SC*, that S_p^l is a provided service whose instances all satisfy

only the HC of S_r and that S_p^k is a provided service whose instances all satisfy both HC and SC of S_r . So, given the canonical interpretation, it is straightforward to see that $\forall x : S_p^k(x) \rightarrow S_p^l(x) \Leftrightarrow (S_p^k)^I \subseteq (S_p^l)^I \Rightarrow |(S_p^k)^I| \leq |(S_p^l)^I| \Rightarrow s(S_r, S_p^k) \leq s(S_r, S_p^l)$. This is the opposite result w.r.t. our criterion, as we want that provided services satisfying both HC and SC of S_r and more similar to S_r are on top of the ranking. For achieving this goal, the ranking procedure is:

given $S_r = \{S_r^{HC}, S_r^{SC}\}$ service request; $S_p^i (i = 1, \dots, n)$ provided services selected by $match(KB, D_r, D_p^i)$;
for $i = 1, \dots, n$ **do** compute $\bar{s}_i := s(S_r^{HC}, S_p^i)$
let be $S_r^{new} \equiv S_r^{HC} \sqcap S_r^{SC}$
for $i = 1, \dots, n$ **do**
 compute $\bar{\bar{s}}_i := s(S_r^{new}, S_p^i)$
 $s_i := (\bar{s}_i + \bar{\bar{s}}_i)/2$

Let us call S_r^{HC} the HC of the request and S_r^{SC} the SC of the request. For all S_p^i , the similarity values $\bar{s} := s(S_r^{HC}, S_p^i)$ are computed. Hence, a new service description $S_r^{new} \equiv S_r^{HC} \sqcap S_r^{SC}$ is considered. The instances of S_r^{new} satisfy both HC and SC of S_r . So, for all S_p^i the similarity values $\bar{\bar{s}} := s(S_r^{new}, S_p^i)$ are computed. A S_p^i satisfying only HC of S_r will has $\bar{\bar{s}} = 0$. For all S_p^i , the final similarity value s_i is given by the average between \bar{s} and $\bar{\bar{s}}$. This last value s_i is used for setting the rank of the services.

Let clarify this process considering the following example. D_r is a requested service description, D_p^l and D_p^k are two provided service description, selected by the matching process. We rank D_p^l and D_p^k . Let note that here D_p^l and D_p^k are described specifying their HC and SC . This is in order to show how the ranking process works in this case. However, the procedure can rank provided services even if they are described without any specification about their constraint hardness.

$$D_r = \{ S_r \equiv \text{Flight} \sqcap \forall \text{operatedBy.LowCostCompany} \sqcap \exists \text{to.}\{\text{bari}\} \sqcap \\ \sqcap \exists \text{from.}\{\text{cologne,hahn}\} \sqcap \forall \text{applicableToFlight.Card;} \\ \{\text{cologne,hahn}\} \sqsubseteq \exists \text{from}^- .S_r \}$$

where

$$HC_r = \{ \text{Flight} \sqcap \exists \text{to.}\{\text{bari}\} \sqcap \exists \text{from.}\{\text{cologne,hahn}\} \\ \{\text{cologne,hahn}\} \sqsubseteq \exists \text{from}^- .S_r \}$$

$$SC_r = \{ \text{Flight} \sqcap \forall \text{operatedBy.LowCostCompany} \sqcap \forall \text{applicableToFlight.Card} \};$$

$$D_p^l = \{ S_p^l \equiv \text{Flight} \sqcap \exists \text{to.Italy} \sqcap \exists \text{from.Germany;} \\ \text{Germany} \sqsubseteq \exists \text{from}^- .S_p^l; \quad \text{Italy} \sqsubseteq \exists \text{to}^- .S_p^l \}$$

where

$$HC_p^l = \{ \text{Flight} \sqcap \exists \text{to.Italy} \sqcap \exists \text{from.Germany;} \quad SC_p^l = \{ \} \\ \text{Germany} \sqsubseteq \exists \text{from}^- .S_p^l; \quad \text{Italy} \sqsubseteq \exists \text{to}^- .S_p^l \}$$

$$D_p^k = \{ S_p^k \equiv \text{Flight} \sqcap \forall \text{operatedBy.LowCostCompany} \sqcap \exists \text{to.Italy} \sqcap \\ \sqcap \exists \text{from.Germany;} \\ \text{Germany} \sqsubseteq \exists \text{from}^- .S_p^k; \quad \text{Italy} \sqsubseteq \exists \text{to}^- .S_p^k \}$$

where

$$\begin{aligned}
HC_p^k &= \{ \text{Flight} \sqcap \exists \text{to} . \text{Italy} \sqcap \exists \text{from} . \text{Germany}; \\
&\quad \text{Germany} \sqsubseteq \exists \text{from}^- . S_p^k; \quad \text{Italy} \sqsubseteq \exists \text{to}^- . S_p^k \} \\
SC_p^k &= \{ \text{Flight} \sqcap \forall \text{operatedBy} . \text{LowCostCompany} \};
\end{aligned}$$

Let us consider S_r , S_p^l and S_p^k . Let note that S_p^l satisfies only HC of S_r while S_p^k satisfies both HC and SC of S_r . Let us suppose that the extensions of S_p^l and S_p^k are: $|(S_p^l)^{\mathcal{I}}| = 8$ and $|(S_p^k)^{\mathcal{I}}| = 5$ and all instances satisfy S_r . Note that $S_p^k \sqsubseteq S_p^l$ then $(S_p^k)^{\mathcal{I}} \subseteq (S_p^l)^{\mathcal{I}}$. So $|(S_r)^{\mathcal{I}}| = 8$. Furthermore, $S_r \not\sqsubseteq S_p^l$ and $S_r \not\sqsubseteq S_p^k$. Let us consider S_r^{HC} and S_r^{SC} . Known that all the instances of S_p^l and S_p^k satisfy S_r and particularly that S_p^l satisfies only HC of S_r while S_p^k satisfies both HC and SC of S_r , it is straightforward to see that $|(S_r^{HC} \sqcap S_p^l)^{\mathcal{I}}| = 8$ and that $|(S_r^{HC} \sqcap S_r^{SC} \sqcap S_p^l)^{\mathcal{I}}| = |(S_r^{new} \sqcap S_p^l)^{\mathcal{I}}| = 0$, consequently $\overline{s_l} = 0$. In the other case, we know that $|(S_p^k)^{\mathcal{I}}| = 5$ and that S_p^k satisfies both HC and SC of S_r . So, some instances of S_p^k can satisfy only HC of S_r and others satisfy both HC and SC (in the better case we could have that all instances of S_p^k satisfy both HC and SC). Let us suppose that instances of S_p^k that satisfy both HC and SC of S_r , namely that satisfy $S_r^{new} \sqsubseteq S_r^{HC} \sqcap S_r^{SC}$ are 3. Let applying the procedure:

$$\begin{aligned}
\overline{s_l} &:= s(S_r^{HC}, S_p^l) = \frac{|(S_r^{HC} \sqcap S_p^l)^{\mathcal{I}}|}{|(S_r^{HC} \sqcup S_p^l)^{\mathcal{I}}|} \cdot \max\left(\frac{|(S_r^{HC} \sqcap S_p^l)^{\mathcal{I}}|}{|(S_r^{HC})^{\mathcal{I}}|}, \frac{|(S_r^{HC} \sqcap S_p^l)^{\mathcal{I}}|}{|(S_p^l)^{\mathcal{I}}|}\right) = \frac{8}{8} \cdot \max\left(\frac{8}{8}, \frac{8}{8}\right) = 1 \\
\overline{s_k} &:= s(S_r^{HC}, S_p^k) = \frac{|(S_r^{HC} \sqcap S_p^k)^{\mathcal{I}}|}{|(S_r^{HC} \sqcup S_p^k)^{\mathcal{I}}|} \cdot \max\left(\frac{|(S_r^{HC} \sqcap S_p^k)^{\mathcal{I}}|}{|(S_r^{HC})^{\mathcal{I}}|}, \frac{|(S_r^{HC} \sqcap S_p^k)^{\mathcal{I}}|}{|(S_p^k)^{\mathcal{I}}|}\right) = 0.625
\end{aligned}$$

The next step is computing $\overline{s_l}$ and $\overline{s_k}$, that are given by:

$$\overline{s_l} := s(S_r^{new}, S_p^l) = 0 \quad \overline{s_k} := \frac{3}{5} \cdot \max\left(\frac{3}{3}, \frac{3}{5}\right) = \frac{3}{5} = 0.6$$

Hence the final similarity values are: $s_l = 0.5$, $s_k = 0.6125$ and so the ranking of the provided services is: 1. S_p^k Similarity Value 0.6125 2. S_p^l Similarity Value 0.5

This result is consistent with the goal. Namely, using this procedure, provided services are ranked w.r.t. both variance and satisfaction of S_r 's SC .

At the first time, the procedure ranks services that satisfy HC w.r.t. variance criteria, indeed provided services that share most of service instances with S_r have higher similarity value. Hence SC are considered. The procedure assigns an additional similarity value to provided services that satisfy also SC . This similarity value is assigned, again using the variance criteria. Let note that in computing the additional similarity value are not considered all the service instances satisfying SC of S_r but only the service instances satisfying both HC and SC of S_r . This avoid to have in higher ranking position provided services that are very similar to SC but dissimilar from HC , whose instances are obviously not preferred w.r.t. to services mostly similar to HC . Indeed the latter can have a lot of instances satisfying SC but that are not relevant at all for the main request.

The main source of complexity of the ranking procedure is given by the computation of the similarity values. As the complexity of the similarity measure mainly depends from the complexity of *instance checking* operator for the chosen DL, the complexity of the ranking procedure depends from the complexity of the instance checking as well.

4 Conclusion and Future Work

This paper proposes a DL-based framework for describing services. Differently from [5], to which it is inspired, our framework allows to express hard and soft constraints,

thus obtaining a more flexible framework for service modeling. Moreover, such constraints can be useful for supplying to the requester the most appropriate services among those selected by the matching phase. To this end, a new ranking procedure was presented. The procedure can take into account the presence of *HC* and *SC*, and the *variance* exploiting a measure that can assess the semantic similarity between service descriptions. This yields a total order among the selected services, differently from [5] where the ranking procedure provides only a partial order and is not able to manage *HC* and *SC*. For the future, an experimentation of the presented work is necessary to show the improvement of the quality of the results supplied to the requester. Moreover, a new matching process could be useful for further increasing the quality of the discovery process and reduce the noise in the selection of services.

References

- [1] F. Baader, D. Calvanese, D. McGuinness, D. Nardi, and P. Patel-Schneider, editors. *The Description Logic Handbook*. Cambridge University Press, 2003.
- [2] Andrea Cali, Diego Calvanese, Simona Colucci, Tommaso Di Noia, and Francesco M. Donini. A description logic based approach for matching user profiles. In *Description Logics*, 2004.
- [3] C. d'Amato, N. Fanizzi, and F. Esposito. A semantic similarity measure for expressive description logics. In A. Pettorossi, editor, *Proceedings of Convegno Italiano di Logica Computazionale, CILC05*, Rome, Italy, 2005.
- [4] J. Gonzales-Castillo, D. Trastour, and C. Bartolini. Description logics for matchmaking of services. In *Proc. of KI-2001 Workshop on Applications of Description Logics*, page vol. 44, 2001.
- [5] S. Grimm, B. Motik, and C. Preist. Variance in e-business service discovery. In *Proceedings of the ISWC Workshop on Semantic Web Services*, 2004.
- [6] M. Klusch, B. Fries, and K. Sycara. Automated semantic web service discovery with owls-mx. In *AAMAS '06: Proceedings of the fifth international joint conference on Autonomous agents and multiagent systems*, pages 915–922, New York, NY, USA, 2006. ACM Press.
- [7] L. Li and I. Horrocks. A software framework for matchmaking based on semantic web technology. In *WWW '03: Proceedings of the 12th international conference on World Wide Web*, pages 331–339, New York, NY, USA, 2003. ACM Press.
- [8] Sheila A. McIlraith and David L. Martin. Bringing semantics to web services. *IEEE Intelligent Systems*, 18(1):90–93, 2003.
- [9] M. Paolucci, T. Kawamura, T.R. Payne, and K.P. Sycara. Semantic matching of web services capabilities. In *International Semantic Web Conference*, pages 333–347, 2002.
- [10] Massimo Paolucci and Katia P. Sycara. Autonomous semantic web services. *IEEE Internet Computing*, 7(5):34–41, 2003.
- [11] Chris Preist. A conceptual architecture for semantic web services. In *Proceeding of International Semantic Web Conference*, pages 395–409, 2004.
- [12] A. Ragone, U. Straccia, T. Di Noia, E. Di Sciascio, and F. M. Donini. Vague knowledge bases for matchmaking in p2p e-marketplaces. In *4th European Semantic Web Conference (ESWC-07)*, LNCS, pages 414–428. Springer Verlag, 2007.
- [13] D. Trastour, C. Bartolini, and C. Preist. Semantic web support for the business-to-business e-commerce lifecycle. In *WWW '02: Proceedings of the 11th international conference on World Wide Web*, pages 89–98, New York, NY, USA, 2002. ACM Press.

Checking e-Service Consistency Using Description Logics

Luigi Dragone*

Dip. di Informatica e Sistemistica – Univ. di Roma “la Sapienza”
via Salaria, 113 – 00198 Roma – Italy
dragone@dis.uniroma1.it
CM Sistemi S.p.A. – via Simone Martini, 126 – 00146 Roma – Italy
luigi.dragone@gruppocm.it

Abstract. We propose a new framework for the analysis of functional properties of e-services supporting the development of *Cooperative Information Systems*. The framework aims at extending and integrating different approaches providing both a rich domain specification and a suitable operational semantics of the e-service contract, on which we define functional consistency properties. It allows for specifying complex e-services based on the *IOPE* paradigm, in which the static properties of the modeled system are specified using a *Description Logic* knowledge base, as assumed in *Semantic Web* applications. Moreover, it enforces a minimal change semantics for the axiomatization of the update operator and also includes the ability to reason about update repairing w.r.t. the domain constraints, thus allowing for incomplete service specification. On this foundation, we formally devise several consistency and validity properties of services, providing decidable checking procedures.

1 Introduction

The *service-oriented computing* paradigm [1] has gained in recent years a lot of interest from the industrial and scientific communities in the field of information technology, in general, and in the design and implementation of information systems, in particular. This paradigm is based upon the metaphor of service (or *e-service*) as a mean to totally encapsulate software application features, in order to make them openly available to highly decoupled clients, implementing a flexible machine-to-machine interaction and building a complex network of dynamically interacting actors. In this scenario, there is a strong requirement on the well-foundedness of the service contract specification between involved actors [2]. Generally, the adoption of a highly expressive language as *First-Order Predicate Logic* (FOL) for modeling dynamic systems easily leads to face very hard (or even unsolvable) problems in terms of automatic verification of formal properties of e-services. So, in order to devise a suitable expressive language for e-service functional modeling, preserving the computability of the associated reasoning problems, we turn our attention to the family of *Description Logics* (DL) [3]. We adopt expressive DL languages (i.e., *ALCQI* and *ALCQIO* [3]) to describe the static world properties, but since we need to cope with some dynamic, and non-deterministic, features like updates, we need to extend the language adequately in order to formulate our

* We are very pleased to acknowledge the invaluable assistance of Prof. Riccardo Rosati.

problems in terms of computable reasoning tasks. In particular, our proposal relies upon the decidable fragment of FOL C^2 [4]: function-free predicate logic with at most two variables and counting quantifiers. The main goal of this work is to design an e-service modeling framework that allows for analyzing functional properties at a high level of abstraction, based on a formal semantic characterization, in order to devise development and execution support tools in the construction of service-oriented solutions. The full version of the paper which includes proofs of theorems is available in [5].

2 Scenario and Related Work

We consider a cooperative information system built according to *service-oriented architecture* [1]. This is a general integration model leveraging on strong encapsulation to decouple implementations. Starting from this assumption, there is a general agreement upon the adoption of a modeling paradigm based on the elicitation of which (not how) information is exchanged during the enactment between requestor and provider, which are the admissible states of the world before the enactment, and which are the possible world states after the correct completion of the execution, according to the so-called IOPE paradigm (*inputs, outputs, preconditions and effects*). In our intentions, the service specification should be intended as the formalization of the service contract. Under a fairness assumption, we suppose that agents act according to domain constraints, which means that they prevent inconsistent evolutions of the world state and that they enforce service contracts. Despite many recent works deal with the problem of modeling and inventorying e-services, the solutions proposed so far are not entirely satisfactory. Given a knowledge representation language (i.e., a *Semantic Web* language), it is always possible to build a classification model of available services, but most of such kind of approaches (e.g., [6], [7]) essentially ignore dynamic features. Several planning-based approaches (e.g., [8], [9]) share with our framework the emphasis on the operational nature, but while they aim at verifying if the available services are sufficient to achieve a specific goal and, possibly, devising an action plan, we are essentially interested in the characterization of the suitability of a set of available services w.r.t. a domain constraint specification. The problem of complex service verification and analysis has been also addressed in similar works based upon the notion of relational transducers [10]. However, in such kind of works, a single complex service, that operates on a relational database, is analyzed in order to derive some correctness properties essentially w.r.t. the service provider state transition constraints. Also the approach proposed in [11] relies on update operational semantics, but it is not directly comparable with our framework, because there are different assumptions on the role played by constraints. Other verification approaches, based upon formal tools like Model Checking, Process Algebra and Petri nets, are also proposed for e-service applications by many authors (e.g., [12]). These are mainly focused on the analysis of the interaction protocol among actors, but generally they do not enforce any assumption about the semantics of the performed operations and often require high-order logical reasoning frameworks, in which many interesting properties are undecidable.

3 The Framework

In this section we introduce the definitions of the primitive constructs on which the proposed framework relies, assuming that the reader is familiar with Description Logics

and FOL. We assume that an infinite countable universe \mathfrak{U} is given and that the system is described using an alphabet composed of a finite set of unary predicate names (or concept names) \mathbf{A} , a finite set of binary predicate names (or role names) \mathbf{P} and a finite set of constant names (or object names) \mathbf{O} , that are constantly interpreted according to the *standard names assumption*. A world state is *legal* iff it is a suitable model for the given specification, while a specification is *consistent* iff it admits at least a legal world state. The specification is an arbitrary \mathcal{ALCQI} knowledge base including both the terminological (TBox) and assertion components (ABox). Given a domain specification, assuming, w.l.o.g., that Top and New are new concept names, we define a knowledge base \tilde{KB} composed by the instantiation of the axiom schema¹ reported in Tab. 1. The basic idea to deal with dynamic features using a “traditional” logic language, in the sense that it does not provide native temporal primitives, is to embed a system state transition, described in terms of initial and final states, assignments, etc., into a single structure on which we solve some reasoning tasks (satisfiability or entailment) obtained by accordingly encoding the problem instance into a suitable set of axioms. The link between original and “working” interpretation structures is caught by the following definition:

Table 1. Axiom schema related to model embedding

$\top \sqsubseteq \text{Top} \sqcup \text{New}$	$A \sqsubseteq \text{Top}$	$\top \sqsubseteq \forall P^-. \text{Top}$
$\text{Top} \sqcap \text{New} \sqsubseteq \perp$	$\top \sqsubseteq \forall P. \text{Top}$	$o : \text{Top}$

Definition 1 (Embedding relation). Let $\omega = \langle \Delta^\omega, \cdot^\omega \rangle$ be an arbitrary world state defined on an interpretation domain $\Delta^\omega \subseteq \mathfrak{U}$, and let $\hat{\omega} = \langle \mathfrak{U}, \cdot^{\hat{\omega}} \rangle$ any interpretation over the alphabet $\langle \mathbf{A} \cup \{\text{Top}\}, \mathbf{P}, \mathbf{O} \rangle$. The world state is embedded into the interpretation ($\omega \rightsquigarrow \hat{\omega}$) iff the following conditions hold: $\Delta^\omega = \text{Top}^{\hat{\omega}}$, $N^\omega = N^{\hat{\omega}}$ and $o^\omega = o^{\hat{\omega}}$, for each $N \in \mathbf{A} \cup \mathbf{P}$ and $o \in \mathbf{O}$.

We inductively define a translation function τ over the concept expressions of the DL language \mathcal{ALCQIO} , from the alphabet $\langle \mathbf{A}, \mathbf{P}, \mathbf{O} \rangle$ to the alphabet $\langle \mathbf{A} \cup \{\text{Top}\}, \mathbf{P}, \mathbf{O} \rangle$, as follows: $\tau(A) \triangleq A$, $\tau(C \sqcap C') \triangleq \tau(C) \sqcap \tau(C')$, $\tau((\bowtie n R C)) \triangleq (\bowtie n R \tau(C))$, $\tau(\{o\}) \triangleq \{o\}$, and $\tau(\neg C) \triangleq \text{Top} \sqcap \neg \tau(C)$.

Proposition 1. Let be ω and $\hat{\omega}$ be respectively a world state and an arbitrary interpretation s.t. the world state is embedded into the interpretation ($\omega \rightsquigarrow \hat{\omega}$), then $C^\omega = [\tau(C)]^{\hat{\omega}}$ (resp. $R^\omega = R^{\hat{\omega}}$) for each \mathcal{ALCQIO} concept (resp. role) expression C (resp. R) over the domain specification alphabet $\langle \mathbf{A}, \mathbf{P}, \mathbf{O} \rangle$.

Theorem 1. The knowledge base KB is satisfiable on an arbitrary interpretation domain $\Delta \subseteq \mathfrak{U}$ iff the knowledge base $\tilde{KB} \wedge \tau(KB)$ is satisfiable on \mathfrak{U} .

The system can dynamically evolve from a state to another one, generally as the result of the execution of some actions. These actions can essentially alter the extensional level or, in other words, perform an update of the system state specification like: (1) creating new objects, which means adding (to the active domain of the resulting

¹ We use *axiom schemas* as a useful notation shorthands: given an alphabet, the instantiated theory is obtained by replacing name placeholders (e.g., A, P, o) with any compatible name.

state) elements that are not included in the active domain of the initial state (assuming that $\mathcal{U} \setminus \Delta^\omega \neq \emptyset$), and that can be viewed as new; (2) adding or removing elements to/from a concept extension or links between elements. The devised framework allows for various complex primitives, generally available in such kind of settings, like: **variables/parameters**, encoded as singleton concepts or *nominals*; **parameterized queries**, encoded as arbitrary *ALCQI*-concept expressions involving also variable names. Generally, an *assignment* binds each variable name to a domain element. We employ the variable construct also to model the object instantiation: *instantiation variables* are bound to newly instantiated objects or, in other words, their assignment is outside the current active domain. Since each instantiation variable is a new distinct object, the assignment function must map different names to different instances, i.e., must be an injective function. We have proved that, given a world state and a variable assignment, the instantiation assignment is quite irrelevant, since all valid extended interpretations are isomorphic at least w.r.t. the domain alphabet. In other words, every interpretation is indistinguishable from each other using the *ALCQI* language [13].

Definition 2 (Simple e-service). Given a domain specification $\langle \mathbf{A}, \mathbf{P}, \mathbf{O} \rangle$, a simple e-service is a quadruple formed by: (1) a (possibly empty) finite set of input variable names \mathbf{X}_S ; (2) a (possibly empty) finite set of output or instantiation variable names \mathbf{Y}_S ; (3) a (possibly empty) finite set of invocation precondition constraint \mathcal{P}_S ; (4) a simple effect specification E_S .

Generally speaking, a *precondition constraint* is a conjunction of positive (resp. negative) atomic conditions that are satisfied if the query result is not empty (resp. is empty) given an input variable assignment and a world state.

An *atomic precondition* is a pair $\langle s, Q(\mathbf{X}) \rangle$ where: (1) $s \in \{+, -\}$ is the sign of the precondition (positive or negative); (2) $Q(\mathbf{X})$ is a query over the domain specification in the variables $\mathbf{X} \subseteq \mathbf{X}_S$. A *simple effect* E is an arbitrary set of atomic concept and role effects built according to its variable names and domain specification. An *atomic concept* (resp. *role*) *effect* is a triple $\langle s, A, a \rangle$ (resp. a quadruple $\langle s, P, l, r \rangle$) s.t.: (1) $s \in \{+, -\}$ is the sign of the effect (insert or delete); (2) $A \in \mathbf{A}$ (resp. $P \in \mathbf{P}$) is the target concept (resp. role) name; (3) a (resp. l and r) is the argument of the update (positive or negative) according to the sign of the effect. A positive (resp. negative) effect argument is any element $Y \in \mathbf{Y}_S$ or any query $Q(\mathbf{X})$ over the domain specification in the variables $\mathbf{X} \subseteq \mathbf{X}_S$ (resp. any query $Q(\mathbf{X})$ over the domain specification in the variables $\mathbf{X} \subseteq \mathbf{X}_S$).

Roughly speaking, a simple effect specifies which elements (or pair of elements) are inserted or removed from a set (or a binary relation).

Example 1. Let \mathcal{W} (Tab. 2) be an axiomatization of a simple domain, where people interact with e-services provided by public administrations. According to the given

Table 2.

$\exists \text{resIn}^- . \top \sqsubseteq \text{Town}$	$\text{Citizen} \sqsubseteq (= 1 \text{ resIn } \top)$	$\text{Citizen} \sqcap \text{Town} \sqsubseteq \perp$
$\exists \text{resIn} . \top \sqsubseteq \text{Citizen}$	$\text{Vehicle} \sqsubseteq (= 1 \text{ regIn } \top)$	$\text{Citizen} \sqcap \text{Goods} \sqsubseteq \perp$
$\exists \text{regIn}^- . \top \sqsubseteq \text{Town}$	$\text{Goods} \sqsubseteq (\leq 1 \text{ own } \top)$	$\text{Goods} \sqcap \text{Town} \sqsubseteq \perp$
$\exists \text{own}^- . \top \sqsubseteq \text{Citizen}$	$t_1, t_2 : \text{Town}$	$\text{Vehicle} \sqsubseteq \text{Goods}$

axiomatization, each goods has an owner, while vehicles must be registered to the local administrative department. Suppose, for example, that there exists a service S that allows a citizen to change its own residence and to specify the new one. The preconditions can be expressed as $\{\{ \langle +, x_1 \sqcap \text{Citizen} \rangle, \langle +, x_2 \sqcap \text{Town} \rangle \}\}$, while effects as $\{-\text{resIn}(x_1, \exists \text{resIn}^- . x_1), +\text{resIn}(x_1, x_2)\}$. The input parameters x_1 and x_2 denote, respectively, the citizen who is asking for the change and the new residence town. This service is accessible by any citizen, and allows to select any town as the new residence place. The town t_1 provides also the following version only to its inhabitants that ask for a residence change that is capable also to accordingly change the registration of vehicles belonging to the requestor, in the sense that the vehicles belonging to the requestor will be registered to the authority of the new town. The preconditions are $\mathcal{P} = \{\{ \langle +, x_1 \sqcap \exists \text{resIn} . \{t_1\} \rangle, \langle +, x_2 \sqcap \text{Town} \rangle \}\}$, while the effects are $E = \{+\text{resIn}(x_1, x_2), -\text{resIn}(x_1, \exists \text{resIn}^- . x_1), -\text{regIn}(\text{Vehicle} \sqcap \exists \text{own} . x_1, \{t_1\})\} \cup \{+\text{regIn}(\text{Vehicle} \sqcap \exists \text{own} . x_1, x_2)\}$.

A set of precondition constraints is interpreted as a disjunction of such constraints. Moreover, a service is *accessible* iff there exists a legal world state and an assignment s.t. it can be activated consistently with its preconditions.

Theorem 2. *Given a world specification \mathcal{W} and a simple e-service S , both defined over the same domain $\langle \mathbf{A}, \mathbf{P}, \mathbf{O} \rangle$, let ω be a world state and σ be an input variable assignment. Then, there exists a *ALCQIO* boolean knowledge base KB^P , having a length linearly bounded by the problem setting size, s.t. S is accessible from ω using σ iff there exists a model $\hat{\omega}$ of KB^P and the world state and the assignment are embedded into it.*

In [5] we show an effective procedure that is able to build KB^P for an arbitrary problem instance. Generally, each insert (resp. delete) effect specification for a target name $N \in \mathbf{A} \cup \mathbf{P}$ defines a set of objects (or links) N^+ (resp. N^-) by the evaluation of its queries given the world state and a variable assignment. In order to provide a consistent definition of service effects, we need also to verify that, for every concept or role, the insert and delete sets are always distinct, as done in other similar approaches (e.g., [14]). An effect is *consistent* iff for each legal world state ω and for each consistent assignment, there is no element or element pair that belongs both to the insert and the delete sets of some concept or role. A detailed analysis of effect consistency checking is described in [5]. Given a pair of world states ω and ω' , we say that ω' is a (potential) *successor state* of ω , resulting from the execution of a service S given an input and an output variable assignments σ_X and σ'_Y , iff: (1) the interpretation domain $\Delta^{\omega'}$ of the successor state is the smallest subset of \mathcal{U} s.t. $\Delta^\omega \cup \text{cod}(\sigma'_Y) \subseteq \Delta^{\omega'}$; (2) the interpretation of object names is preserved ($o^\omega = o^{\omega'}$); (3) for each concept or role name $N \in \mathbf{A} \cup \mathbf{P}$, the insert set is included in the successor state interpretation, i.e. $N^{\omega'} \supseteq N^+(\omega, \sigma_X, \sigma'_Y)$, and the delete set is excluded, i.e. $N^{\omega'} \cap N^-(\omega, \sigma_X) \subseteq \emptyset$. The implemented *state transition relation* is from a state to the closest successor one, according the *set symmetric difference* (∇) applied to interpretation structures w.r.t. concept and role alphabets \mathbf{A} and \mathbf{P} .

The *service enactment* set for a state ω and a consistent input variable assignment σ_X , denoted as $S(\omega, \sigma_X)$, contains all the pairs $\langle \omega', \sigma'_Y \rangle$ s.t.: (1) σ'_Y is a consistent instantiation assignment w.r.t. ω ; (2) ω and ω' are in transition relation w.r.t. the assignment and the service effect.

On such foundation, as in the previous cases, we obtain the following result:

Theorem 3. *Given a world specification \mathcal{W} and an accessible and consistently defined simple e-service S , both defined over the same domain $\langle \mathbf{A}, \mathbf{P}, \mathbf{O} \rangle$, there exists a \mathcal{C}^2 sentence KB^U , having a length polynomially bounded by the problem setting size, s.t., every quadruple $\langle \omega, \omega', \sigma_X, \sigma'_Y \rangle$ embedded into a structure $\hat{\omega}$ is an enactment $\langle \omega', \sigma'_Y \rangle \in S(\omega, \sigma_X)$ of S , iff $\hat{\omega} \models KB^U$.*

We are interested in the property of a service that always acts consistently with its specification and the system constraints. In other terms, given any legal state where service preconditions hold, the service is *valid* iff its invocations always result into a legal state where the declared effects are realized.

Example 2. Given the specification of Ex. 1, consider the following e-service that allows a customer to buy a new vehicle (i.e., to become the owner): $\langle \mathbf{X} = \{x\}, \mathbf{Y} = \{y\}, \mathcal{P} = \{\{ \langle +, x_1 \sqcap \text{Citizen} \rangle \}\}, E = \{+\text{Vehicle}(y), +\text{own}(y, x)\}$. Despite the service is accessible and its effects are consistently defined, it is not valid since its enactments violate the domain constraints: a vehicle must be recorded to the town authority. A valid version of the service effects is the following: $E = \{+\text{Vehicle}(y)\} \cup \{+\text{own}(y, x), +\text{Goods}(y), +\text{regIn}(y, \exists \text{resIn}.x)\}$

Theorem 4. *A consistent and accessible simple e-service S is valid w.r.t. a world specification \mathcal{W} iff the following implication holds: $KB^U \wedge \tau(\mathcal{W}) \models \bar{\tau}(\mathcal{W})$, where $\bar{\tau}$ is a translation function defined w.r.t. the name mapping $m(x) \triangleq \bar{x}$.*

Based on the above property, since the complexity of reasoning in \mathcal{C}^2 [4], we are able to provide the following complexity upper bound:

Corollary 1. *Given a world specification \mathcal{W} and an accessible and consistent simple e-service S , the problem of checking if S is also valid w.r.t. \mathcal{W} is in coNEXP.*

4 Incomplete Specifications and Repairs

Now we analyze the properties of e-services keeping into account incomplete specifications and a form of non-determinism effect repair. In the field of update theory, the notion of repair is old at least as the notion of update itself [15]. However, the problem of repairing even a simple update in presence of a complex intensional knowledge base turns out to be very hard, since, given the complexity of the language, non-local repair side-effects may arise. Some authors have addressed the problem limiting the constraint language to a simpler form (e.g., acyclic or definitorial TBox), but in the general case the problem is undecidable both in DL [14] and in relational database schemas [16]. Generally speaking, since we can reduce, using some adjustments, our framework to the proposal of [14], the problem is also undecidable in our framework, but, if we renounce to the completeness of the repair search, limiting to a restricted, and finite, set of possible repairs, we can regain decidability.

Example 3. Given the following world specification of Ex. 1 and the following service specification: $\langle \mathbf{X} = \emptyset, \mathbf{Y} = \{y\}, \mathcal{P} = \emptyset, E = \{+\text{Vehicle}(y)\}$, it is trivial to observe that the service is not valid, since the insertion of a new element in the extension of the concept Vehicle violates various axioms. In order to enforce this constraint, a repair like $\{+\text{Goods}(y), +\text{regIn}(y, \{t_1\})\}$ is enough.

A *simple repair* R for a simple e-service S is an arbitrary set of atomic concept and role repairs, possibly empty, s.t. it does not contain any pair of conflicting atomic repairs (i.e., atomic repair differing only by the sign). An *atomic repair* is a special kind of atomic effect, whose arguments range over any nominal introduced in the specification. The number of different repairs, or, in other words, the size of the search space, is finite and exponentially bounded by the number of alphabet elements (in terms of names), while they are substantially independent of the complexity of the world specification axioms and service effect statements.

Definition 3 (Candidate repaired successor state). *Given a world specification \mathcal{W} and an enactment $\langle \omega', \sigma_Y' \rangle \in S(\omega, \sigma_X)$ of a service S , let $R \in R_S$ be a simple repair. The associated candidate repaired successor state ω'_R is an interpretation structure s.t.:* (1) *its interpretation domain is the same as the interpretation domain of the successor state ($\Delta^{\omega'} = \Delta^{\omega'_R}$);* (2) *the interpretation of each concept or role name $N \in \mathbf{A} \cup \mathbf{P}$ is adjusted accordingly to the repair;* (3) *the interpretation of each object name $O \in \mathbf{O}$ is left unchanged ($O^{\omega'_R} = O^{\omega'}$).*

The provided definition is not complete: in fact, in order to be useful and safe, a repair R should be s.t.: (1) it does not “undo” the effects of the service; (2) it actually updates the world state. Moreover, among multiple candidate repaired successor states, the repairing strategy selects the one closest to the base successor state ω' , in terms of symmetric difference between interpretation structures, in order to enforce a kind of minimal-change repair. According to this approach, which assumes a repair step following the service update, we need to refine the embedding relation in order to keep into account also the intermediate transient state. We can employ a name mapping function to cope with multiple repairs at the same time, using different names for each non-deterministic branch. So, let $R_i \in R_S$ be a repair, a suitable name mapping function in the case of simple e-service is $n_i(x) \triangleq x_i$ and $n_i(\text{Top}) \triangleq \text{Top}$.

Theorem 5. *Given a world specification \mathcal{W} and an accessible and consistently defined simple e-service S , both defined over the same domain $\langle \mathbf{A}, \mathbf{P}, \mathbf{O} \rangle$, and a simple repair R^* for S , there exists a \mathcal{C}^2 sentence $\Delta KB^R(R^*)$, having a length polynomially bounded by the problem setting size, s.t. every quintuple $\langle \omega, \omega', \omega'', \sigma_X, \sigma_Y' \rangle$ embedded into a structure $\hat{\omega}$ is an enactment $\langle \omega', \sigma_Y' \rangle \in S(\omega, \sigma_X)$ of S repaired applying R^* leading to state ω'' , iff $\hat{\omega} \models KB^U \wedge \Delta KB^R(R^*)$.*

An accessible service, with consistent effects, is *repairable* iff for any possible enactment always exists a simple repair that is able to achieve a legal state without retracting service effects. Such a property can be checked using the following result.

Theorem 6. *A consistent and accessible simple e-service S is repairable w.r.t. a world specification \mathcal{W} using a family of repair $R_S = \{R_1, \dots, R_r\}$, iff the following implication holds $\tau(\mathcal{W}) \wedge KB^U \wedge \bigwedge_{i=1}^r \Delta KB^R(R_i) \models \bigvee_{i=1}^r \tau_i(\mathcal{W}) \wedge \Delta KB^C(R_i)$, where τ_i is the translation function defined w.r.t. the name mapping function n_i and $\Delta KB^C(R_i)$ is a suitable \mathcal{C}^2 sentence that encodes the repair consistency constraints for a given repair, having a length polynomially bounded by the problem setting.*

We have also devised in [5] an effective procedure to build both ΔKB^R and ΔKB^C for given service S and simple repair $R \in R_S$. Moreover, since an exponential number of possible repairs must be accordingly encoded, we obtain the following property:

Corollary 2. *Given a world specification \mathcal{W} and an accessible and consistent simple e-service S , the problem of checking if S is also repairable w.r.t. \mathcal{W} is in co2NEXP.*

5 Conclusions

The main contribution of this work is to show how an expressive and suitable operational semantic model for e-service, based upon the enforcing of the service contract, can be traced to a logic characterization as a foundation for the analysis of a number of relevant functional properties in the design of a service-oriented integration solution. Moreover, the model accounts for both complete and incomplete specifications of e-services. To the best of our knowledge, this is the only framework allowing for effective reasoning about action consequences w.r.t. a complex domain specification, without any significant limitation regarding the constraint language, while adopting a semantics of minimal-change and taking into account more interesting, and semantically well-founded, repairing options.

References

1. Alonso, G., Casati, F., Kuno, H., Machiraju, V.: *Web Services: Concepts, Architectures and Applications*. Springer-Verlag (2004)
2. Meyer, B.: *Object-Oriented Software Construction*. Prentice-Hall, Inc. (1988)
3. Baader, F., Calvanese, D., McGuinness, D.L., Nardi, D., Patel-Schneider, P.F., eds.: *The Description Logic Handbook*. Cambridge University Press (2003)
4. Pratt-Hartmann, I.: Complexity of the two-variable fragment with counting quantifiers. *J. of Logic, Lang. and Inf.* **14**(3) (2005) 369–395
5. Dragone, L., Rosati, R.: *Modeling and reasoning about e-services*. Technical report, DIS, Università di Roma “la Sapienza” (2006)
6. Noia, T.D., Sciascio, E.D., Donini, F.M., Mongiello, M.: *A system for principled matchmaking in an electronic marketplace*. In: *Proc. of the WWW 2003 Conference*. (2003)
7. Grimm, S., Motik, B., Preist, C.: *Matching semantic service descriptions with local closed-world reasoning*. In: *Proc. of 3rd Annual European Semantic Web Conference*. (2006)
8. McIlraith, S.A., Son, T.C.: *Adapting Golog for composition of semantic web services*. In: *Proc. of the KR 2002 Conference*. (2002)
9. Carman, M., Serafini, L., Traverso, P.: *Web Service Composition as Planning*. In: *Proc. of ICAPS’03 Workshop on Planning for Web Services*. (2003)
10. Deutsch, A., Sui, L., Vianu, V.: *Specification and verification of data-driven web services*. In: *Proc. of the PODS 2004 Conference*. (2004)
11. Berardi, D., Calvanese, D., De Giacomo, G., Hull, R., Mecella, M.: *Automatic composition of transition-based semantic web services with messaging*. In: *Proc. of the 31st VLDB Conference*. (2005)
12. Martens, A.: *Usability of web services*. In: *Proc. of the 1st Web Service Quality Workshop*. (2003)
13. Immerman, N., Lander, E.: *Describing graphs: A first-order approach to graph canonization*. In: Alan L. Selman, Editor, *Complexity Theory Retrospective*. Volume 1. (1990)
14. Baader, F., Lutz, C., Milicic, M., Sattler, U., Wolter, F.: *Integrating description logics and action formalisms for reasoning about web services*. Technical report, Desden University of Technology (2005)
15. Winslett, M.: *Updating logical databases*. Cambridge University Press (1990)
16. Abiteboul, S., Hull, R., Vianu, V.: *Foundations of Databases*. Addison-Wesley (1995)

L-SME: A Tool for the Efficient Discovery of Loosely Structured Motifs in Biological Data

Fabio Fassetti¹, Gianluigi Greco², and Giorgio Terracina²

¹DEIS and ²DIMAT, Università della Calabria, 87036 Rende (CS), Italy
 fabio.fassetti@deis.unical.it, {ggreco, terracina}@mat.unical.it

Abstract. Motifs discovery is a crucial task in bioinformatics and, in fact, several approaches have already been proposed to deal with motifs exhibiting some kinds of regular structure. In this paper, an efficient approach is instead discussed for extracting *loosely structured* motifs, i.e., motifs whose structure is characterized by a large number of variabilities, such as the number of boxes, their lengths, the gaps between them, and the possibility of skipping and swapping boxes.

1 Introduction

The development of effective knowledge discovery techniques tailored for genomic analysis has become a very active research area in recent years. Genomic data usually come in the form of string databases, where each string represents a DNA or a protein sequence. Interpreting these data comprises quite diverse tasks and associated methods.

In this paper, we consider the task of extracting *biologically relevant* patterns, which are also called *motifs*. In the literature, motifs are often classified either as *simple motifs*, if they represent single portions of conserved regions (i.e., pieces of genomic code approximately repeated across related sequences), or as *structured motifs*, if they represent ordered collections of simple motifs (boxes, in the following) with gap constraints between each pair of adjacent boxes.

More formally, a pattern structure \hat{p} is a tuple $\langle l_1, d_1, l_2, d_2, \dots, d_{r-1}, l_r \rangle$, where l_i indicates the length of the i -th box, d_j indicates the length of the gap separating the j -th and the $(j + 1)$ -th boxes, and r is the number of boxes. Actually, both l_i and d_j can be (possibly degenerating) intervals of the form $l_i = [\min _l_i, \max _l_i]$ and $d_j = [\min _d_j, \max _d_j]$. A *pattern instance* p for a pattern structure \hat{p} is a string $p = b_{l_1} X(d_1) b_{l_2} X(d_2) \dots X(d_{r-1}) b_{l_r}$, where b_{l_i} are strings, and $X(d_j)$ is a sequence of d_j “don’t care” symbols X . In the following, pattern instances are simply referred to as patterns. Figure 1.(b) graphically shows an example of the pattern structure $\hat{p} = \langle 3, 2, 2, 2, 2 \rangle$, whereas Figure 1.(c) highlights the possible pattern instances for \hat{p} that can be extracted from the string database SDB of Figure 1.(a).

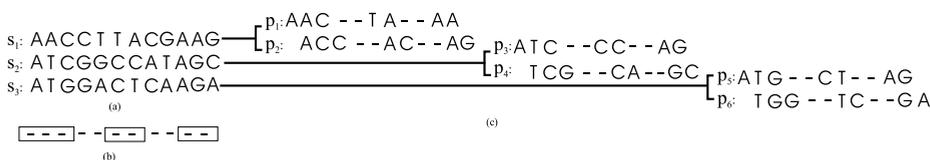


Fig. 1. (a) A String Database SDB , (b) A Pattern Structure \hat{p} , (c) The Candidate Patterns for SDB and \hat{p} .

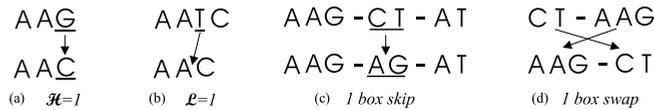


Fig. 2. Box similarity under (a) the Hamming distance, (b) the Levenshtein distance. Pattern similarity if (c) one skip is allowed, (d) one swap is allowed.

In the structured motif mining problem, the pattern structure is assumed to characterize some biological specificities of the organisms. For instance, the most frequently observed prokaryotic promoter regions are in general composed by two boxes positioned approximately 10 and 35 bases upstream from the transcription start; in other words, a pattern structure can be seen as a template for the motifs of interest.

However, dealing with many variabilities in the specification of pattern structures may lead to computationally hard mining tasks. This is also complicated by the fact that pattern occurrences are generally measured according to some concept of similarity (rather than w.r.t. the exact matching), in order to tolerate mismatches that often characterize biologic domains. In fact, current approaches impose quite “rigid” constraints on the pattern structures or they consider simple kinds of pattern similarity.

In this work, we provide a contribution in this setting, by presenting a novel approach for the mining of motifs that improves on the variety of allowed pattern specifications, and that is still scalable in practical application domains, as for it emerges from an extensive experimentation on real datasets. In a nutshell, the peculiarities of the approach are as follows:

- ▶ The number of boxes, their lengths, and the gaps between them can vary in some user-defined intervals.
- ▶ Similarity between single boxes can be specified either with the Hamming distance (\mathcal{H}), which allows only for symbol mismatches and, consequently, allows comparing only equal length boxes, or with the Levenshtein distance (\mathcal{L}) which considers also insertions and deletions and, thus, allows comparing different length boxes.
- ▶ Finally, the basic similarity between pattern instances (obtained when each single box of the first pattern is sufficiently similar with the corresponding box of the second pattern) is extended by allowing some *skips* or *swaps* of boxes in pattern repetitions; many biological studies demonstrated the relevance of these situations.

Due to space constraints, we can not formalize here all the kinds of motif the approach is able to deal with—we call these motifs *loosely structured*. To help the intuition, we just report in Figure 2 some scenarios that can be faced with our approach. Note that, to the best of our knowledge, there is no technique proposed in the literature considering all these variabilities altogether. Details on the approach are reported in the full version of the paper [3], while the system L-SME, integrating all the algorithms discussed in the paper, is available at <http://siloe.deis.unical.it/l-sme>.

2 Overview of the Approach

A crucial role in the proposed approach is played by the concept of \bar{e} -neighborhood of boxes and patterns. Intuitively, given a maximum error \bar{e} that can be tolerated for a box (resp., a pattern), its \bar{e} -neighbor set is a compact representation of all the possible strings

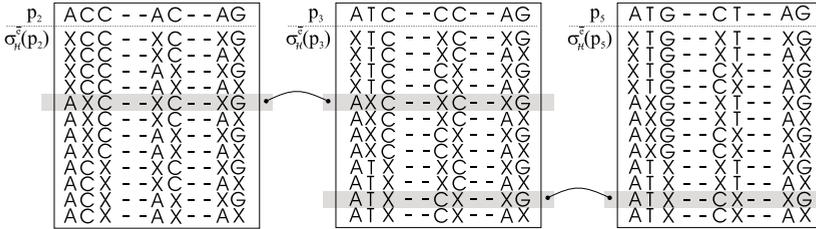


Fig. 3. Examples of \bar{e} -neighbor patterns for $\bar{e} = \langle 1, 1, 1 \rangle$.

(resp., patterns) that can be considered an occurrence of it; compactness is obtained with the exploitation of don't care symbols that represent sets of symbols. As an example, let $b = AAC$ be a box of length 3 and assume to tolerate at most $\bar{e} = \langle 1 \rangle$ substitution (i.e., $\mathcal{H}=1$ at most), the \bar{e} -neighbor set of b is: $\sigma_{\mathcal{H}}^{\bar{e}}(b) = \{AAX, AXC, XAC\}$. Note that, while the number of possible strings at Hamming distance \bar{e} from a box b of length l_j over an alphabet Σ is $\sum_{i=1}^{\bar{e}} \binom{l_j}{i} (|\Sigma| - 1)^i$, the size of $\sigma_{\mathcal{H}}^{\bar{e}}(b)$ is just $\binom{l_j}{\bar{e}}$.

Extending the concept of \bar{e} -neighbor to patterns is straightforward. Figure 3 illustrates the \bar{e} -neighbor sets for some of the candidate patterns shown in Figure 1 when the maximum Hamming distances allowed for each box are $\bar{e} = \langle 1, 1, 1 \rangle$. The figure also evidences an important property of \bar{e} -neighbors: *for any two patterns p' and p'' complying with the same structure \hat{p} , p' can be considered an occurrence of p'' (and vice versa) if and only if $\sigma_{\mathcal{H}}^{\bar{e}}(p') \cap \sigma_{\mathcal{H}}^{\bar{e}}(p'') \neq \emptyset$* . Thus, in order to count the occurrences of a pattern p , we need just to consider its (compact) \bar{e} -neighbor set.

Interestingly, \bar{e} -neighbors enjoy some properties that are even more useful. First, \bar{e} -neighbors can be used to easily handle involved kinds of pattern similarity. As an example, the Levenshtein distance can be handled as follows. To generate all the strings at Levenshtein distance e from b , three edit operations must be considered: symbol substitutions, symbol insertions and symbol deletions. Substitutions are in fact mismatches and can be handled by don't care symbols as done for the Hamming distance; insertions and deletions are complementary, since the deletion of a symbol in one string corresponds to an insertion in the other string. Then, a don't care symbol inserted in the first string can be used to represent the insertion of any possible symbol in that position needed to match the corresponding symbol in the second string. For instance, let $b = TAC$ and $e = 2$. Then, $\sigma_{\mathcal{L}}^e(b) = \{TXX, TXAX, TAXX, TXXC, TXAXC, TAXXC, XAX, XAXC, XXC, XXAC, TXXAC\}$, where, for instance, the e -neighbor $XAXC$ represents all the strings obtainable from b with a substitution in the first position and an insertion in the third position (e.g., with the optimal alignment $TA - C$).

Similarly, \bar{e} -neighbors can be used to handle skips and swaps of boxes, just enriching the base \bar{e} -neighbor set of patterns, as illustrated in Figure 4. Also in this case, the important property of \bar{e} -neighbor outlined above still holds; this allows us to identify the occurrences of a pattern under loose specifications looking just at the \bar{e} -neighbor set of the pattern itself.

2.1 Algorithm and Data Structures

We are now able to highlight our algorithm for mining loosely structured motifs based on \bar{e} -neighbors. The algorithm works in two phases.

2.2 A Randomized Variant

One of the heaviest parts, from a computational point of view, consists in the need of storing the occurrences of \bar{e} -neighbors by means of boolean arrays. This impacts both on the space requirements (to store the arrays) and on time complexity (to compute the bitwise ORs), causing a quadratic dependency from the number n of input sequences.

To further improve our approach, we then designed a randomized variant of the main algorithm (where randomization is limited in the way pattern occurrences are represented) that yet provides a guarantee on the approximation performances. Specifically, the guarantee depends on the additional space used to store patterns occurrences; hence, in the extreme case where $O(n)$ bits are used to this aim, the randomized variant will in fact degenerate to the exhaustive one.

In more detail, we resort to some recent techniques introduced for efficiently processing *data streams* and, in particular, we substitute the boolean array with the *sketch* defined in [2] introduced to compute the Hamming norm of multiple data streams. In fact, such sketches enjoy the following nice property: *Given a stream \mathbf{a} , and two values $\epsilon \in [0, 1]$ and $\delta \in [0, 1]$, a sketch $sk(\mathbf{a})$ can be computed such that:* (1) *It requires space $O(\frac{1}{\epsilon^2} \log \frac{1}{\delta})$;* (2) *It allows the approximation of the Hamming norm of \mathbf{a} within a factor $1 \pm \epsilon$, with probability $1 - \delta$;* (3) *Both updating $sk(\mathbf{a})$ and computing the Hamming norm of \mathbf{a} take time $O(\frac{1}{\epsilon^2} \log \frac{1}{\delta})$;* (4) *The Hamming norm of the union of k multiple streams can be computed within a factor $1 \pm \epsilon$, with probability $1 - \delta$, in time $O(k \times \frac{1}{\epsilon^2} \log \frac{1}{\delta})$.*

Since computing the Hamming norm of the union of k multiple streams corresponds to counting the number of *true* elements in the set of boolean arrays associated with the \bar{e} -neighbor patterns of a pattern p , this kind of structure perfectly fits our application context and allows our overall approach to scale linearly (both in time and space) in the number of input sequences. This result has never been obtained by previous approaches.

3 Experiments

Both the algorithms presented in the paper have been implemented and integrated into a system (L-SME) which has been thoroughly tested on syntectic and real biological data (non-coding regions of *B. subtilis*, *H. Pylori*, and *E. coli*). This section is devoted to discuss some results for this experimental activity. All tests have been carried out on a Pentium IV machine at 3.2 Ghz and equipped with 4Gb RAM under Linux.

3.1 Time and Space Requirements

We first compared the time and space requirements of the two algorithms presented in the paper. To this end, we designed two typologies of tests.

Test (a). For this test, we used syntectic data ranging from 1'000 to 1'000'000 sequences, each one being formed by 100 basis, thus for a total of 100'000'000 basis in the worst case. Sequences have been generated randomly, considering each basis equally probable. In fact, this choice puts L-SME in the worst case scenario, where each word (of a fixed length) is equally probable, thereby reducing the chances of having several candidate patterns sharing the same portions of the Keyword Trees. Moreover,

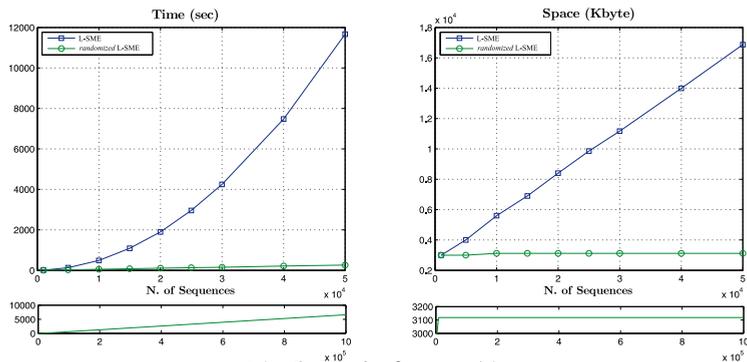


Fig. 6. Results for Test (a).

data sets have been incrementally constructed, in such a way that each new data set was exactly a superset of all the previously generated ones. The pattern structure used has been fixed to $\hat{p} = \langle 3, 8, 3 \rangle, \bar{e} = \langle 1, 1 \rangle$.

Test (b). For this test, we used real biological data, i.e., a database of about 11'000 sequences from non coding regions of B. Subtilis, H. Pilory and E. Coli, with average length of 180 basis, for a total of about 1'900'000 input basis. Moreover, we considered the following pattern structures: (b.1) $\hat{p} = \langle 3, 8, 3 \rangle, \bar{e} = \langle 1, 1 \rangle$; (b.2) $\hat{p} = \langle 4, 8, 4 \rangle, \bar{e} = \langle 1, 1 \rangle$; (b.3) $\hat{p} = \langle 4, 8, 4 \rangle, \bar{e} = \langle 2, 2 \rangle$; (b.4) $\hat{p} = \langle 5, 8, 5 \rangle, \bar{e} = \langle 1, 1 \rangle$; (b.5) $\hat{p} = \langle 5, 8, 5 \rangle, \bar{e} = \langle 2, 2 \rangle$; (b.6) $\hat{p} = \langle 6, 8, 6 \rangle, \bar{e} = \langle 2, 2 \rangle$; (b.7) $\hat{p} = \langle 6, 8, 6 \rangle, \bar{e} = \langle 3, 3 \rangle$.

Due to space constraints, we can not show here all the results of our tests. Figure 6 reports, on the left, the time responses we have measured for Test (a). The quadratic scaling of the algorithm clearly emerges, but with a scale factor fairly acceptable; indeed, more than 50'000 sequences can be handled in reasonable time. Also, we can appreciate the linear dependency of the randomized version in the number of input sequences, which is further stressed in the graph shown below the main picture, where its scaling is reported up to 1'000'000 sequences.

The same figure on the right reports the actual space required by both the algorithms. We note that the basic algorithm scales linearly in the number of sequences, instead of quadratically as predicted by the theoretical (worst case) considerations [3]. Indeed, in practice, the higher is the number of input sequence, the higher is the *expected* number of e -neighbors sharing the same portions of the Keyword Tree. This causes a significant space saving w.r.t. the worst case, since (i) the space required by the use of boolean arrays is counterbalanced by the decreasing number of nodes actually occurring in the tree (the Keyword Tree quickly becomes *full*), and (ii) the number of candidate patterns becomes significantly greater than the number of generated \bar{e} -neighbor patterns. The same line of reasoning justifies the practically constant space required by the randomized version. In fact, in this algorithm, boolean arrays are replaced by sketches of constant dimension; then, since the dimension of the involved tree (quickly) converges to a constant value corresponding to the space needed by the full version, we have that the space required by the randomized variant also tends to converge to a fixed value.

In the light of the above results witnessing the capability of efficiently handling more than 100'000'000 basis, our approach emerges to be suited for dealing with very large data sets, e.g., in genome-wide frequent pattern mining.

Test (a)			Test (b)		
N. of Sequences	Exhaustive vs Theoretical	Randomized vs Exhaustive	Configuration	Exhaustive vs Theoretical	Randomized vs Exhaustive
5000	99.93%	25.00%	(b.1)	99.98%	45.87%
10000	99.96%	44.29%	(b.2)	99.75%	83.38%
20000	99.98%	62.87%	(b.3)	99.98%	70.64%
30000	99.99%	72.07%	(b.4)	96.96%	85.83%
40000	99.99%	77.70%	(b.5)	99.74%	85.15%
50000	99.99%	81.51%	(b.6)	-	-
1000000	99.99%	98.88%	(b.7)	99.72%	85.73%

Table 1. Analysis of the space saving allowed by the two versions of the approach.

3.2 Impact of Keyword Trees

In order to further analyze the benefits obtained by the adoption of Keyword Trees and the use of Sketches in representing candidate patterns and their occurrences, we carried out another set of tests specifically conceived to measure the space saving guaranteed by our approaches. In particular, we measured the relative gain, in terms of needed space, obtained by the basic algorithm w.r.t. the theoretical expectations. To this end, we compared the total number of nodes generated by the algorithm with the total number of generated \bar{e} -neighbor patterns. This allows measuring our improvement w.r.t. a naïve method storing each computed \bar{e} -neighbor pattern in a separate structure. Moreover, we measured the relative gain obtained by the randomized variant w.r.t. our basic algorithm, computed by comparing the actual memory occupied by them. Results for both the two measures are reported in Table 1.

By analyzing this table, we can firstly observe the enormous advantage of using Keyword Trees on naïve approaches. Moreover, for the randomized variant, we can notice the great benefit of using sketches in terms of space saving, in all the tested scenarios; in particular, the benefit is generally higher for problems of higher complexity.

3.3 Comparison with RISO

We also compared our system with RISO [1] for the subclass of patterns it supports. RISO is one of the most recently proposed pattern discovery tools in the bioinformatics context, and significantly outperforms previously proposed systems [1]. Figures 7 and 8 show some results of this comparison.

In particular, Figure 7.(a) plots the time responses of RISO and L-SME for increasing values of the maximum allowed Hamming distance e . It emerges that for low values of e the performance of the two systems are comparable; on the contrary, for medium-high values of e , the adoption of e -neighbors in L-SME allows a significant reduction of the computational efforts, whereas RISO presents an exponentially increasing trend.

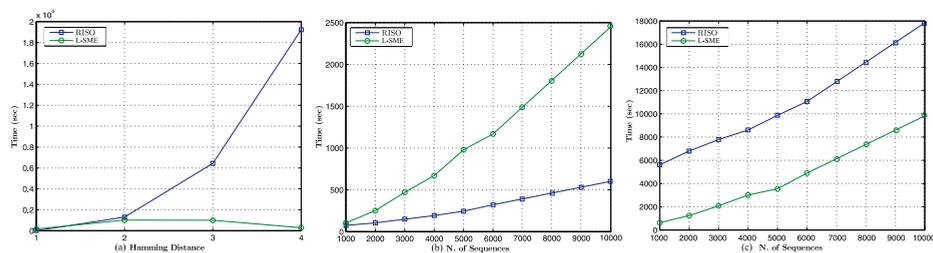


Fig. 7. RISO vs L-SME: (a) Increasing values of e ; (b) $\hat{p}' = \langle 5, 16, 5 \rangle$ and $\bar{e} = \langle 2, 2 \rangle$; and, (c) $\hat{p}'' = \langle 6, 16, 6 \rangle$ and $\bar{e} = \langle 3, 3 \rangle$.

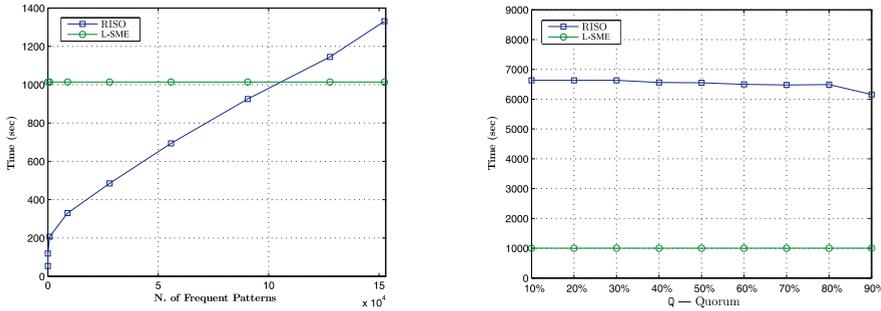


Fig. 8. RISO vs L-SME: (left) increasing number of frequent patterns; (right) increasing quorums.

We also compared the scaling of the two systems w.r.t. the number of input sequences. To this purpose, we exploited the synthetic data sets introduced for *Test (a)* and we considered two kinds of pattern structures: $\hat{p}' = \langle 5, 16, 5 \rangle$ with $\bar{e} = \langle 2, 2 \rangle$ and $\hat{p}'' = \langle 6, 16, 6 \rangle$ with $\bar{e} = \langle 3, 3 \rangle$. Obtained results are shown in Figures 7.(b) and 7.(c), respectively. As far as \hat{p}' is concerned, we may observe that both the systems scale practically linearly, even if RISO presents a lower scaling factor w.r.t. L-SME; moreover, the response times are comparable for a low number of input sequences but significantly increases for big input data sets. Instead, for \hat{p}'' , we observe that even if the scaling factor of the two systems is exactly the same, the response times of L-SME are significantly lower than those of RISO, already for small input data sets. These results (along with Figure 7.(a)) outline that RISO could be more appropriate than L-SME for conserved patterns and that, in turn, L-SME is better suited for longer and more complex motifs.

As a further comparison between the two systems, we measured their time performance w.r.t. the number of patterns turned out to be frequent. Results for two tests, considering different situations, are shown in Figure 8. In particular, on the left we consider a situation in which the number of detected frequent patterns significantly varies w.r.t. the value of the minimum desired quorum Q (the database is relatively sparse w.r.t. \hat{p} and \bar{e}). The picture clearly shows that the time response of L-SME is completely independent of the number of frequent patterns, whereas RISO heavily depends on this parameter. In this specific application context, RISO significantly benefits of this property; in fact, it presents very fast response times when the number of frequent patterns is very low; however, this benefit rapidly decreases with the increase of the number of frequent patterns, until the time response of RISO becomes higher than that of L-SME.

To the contrary, the test case shown on the right of Figure 8 represents a situation in which the database is particularly dense w.r.t. the considered pattern class; in fact, the number of frequent patterns ranges in a short interval. Results obtained in this test show that in presence of a “dense” database L-SME performs much better than RISO.

References

1. A.M. Carvalho, A.T. Freitas, A.L. Oliveira, and M.F. Sagot. An Efficient Algorithm for the Identification of Structured Motifs in DNA Promoter Sequences. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, 3(2):126–140, 2006.
2. G. Cormode, M. Datar, P. Indyk, and S. Muthukrishnan. Comparing data streams using hamming norms (how to zero in). *IEEE TKDE*, 15(3):529–540, 2003.
3. F. Fassetti, G. Greco, and G. Terracina. Discovering loosely structured motifs from biological data sets. In *Proc. of SAC 2006*, pp. 151–155, Dijon, France, 2006. ACM Press.

An Information-Theoretic Framework for High-Order Co-Clustering of Heterogeneous Objects ^{*}

Gianluigi Greco^a, Antonella Guzzo^b, Luigi Pontieri^c

Dept. of Mathematics^a, UNICAL, Via P. Bucci 30B, 87036, Rende, Italy

Dept. DEIS^b, UNICAL, Via P. Bucci 30B, 87036, Rende, Italy

ICAR, CNR^c, Via Pietro Bucci 41C, 87036 Rende, Italy

ggreco@mat.unical.it, guzzo@deis.unical.it, pontieri@icar.cnr.it

Abstract. The *high-order co-clustering* problem, i.e., the problem of simultaneously clustering several heterogeneous types of domains, is usually faced by minimizing a linear combination of some optimization functions evaluated over pairs of correlated domains, where each weight expresses the reliability/relevance of the associated contingency table. Clearly enough, accurately choosing these weights is crucial to the effectiveness of the co-clustering, and techniques for their automatic tuning are particularly desirable, which are instead missing in the literature. This paper faces this issue by proposing an information-theoretic framework where the co-clustering problem does not need any explicit weighting scheme for combining pairwise objective functions, while a suitable notion of *agreement* among these functions is exploited. Based on this notion, an algorithm for co-clustering a “star-structured” collection of domains is defined.

1 Introduction

The problem of clustering heterogeneous objects has become an active research area recently. In particular, a great deal of literature addresses the clustering of two different types of objects, hereinafter called *domains* or *dimensions*, such as documents and terms in text corpora (e.g., [7, 4]), or genes and conditions in micro-array data.

In this context, a bi-dimensional contingency table is typically used to represent correlations between the two domains (e.g., frequency/relevance of each term for each document), and the problem amounts to simultaneously cluster both domains by exploiting the clear duality between rows and columns. This task, usually known as (bi-dimensional) *co-clustering* or *bi-clustering*, was faced by using different strategies, including spectral [7, 4] and information-theoretic approaches [5, 3, 1].

Some recent works have generalized the bi-clustering problem to the case of more than two domains (short: *high-order co-clustering* problem) [2, 6]. In particular, [6] considered a co-clustering problem for “star”-structured domains of the form $D_X, D_{Y^1}, \dots, D_{Y^N}$ (where $N > 1$ and D_X is the central domain), by defining an objective function f_{X, Y^i} , for each “auxiliary” domain D_{Y^i} , whose optimization should lead to the isolation of the best co-clusters over D_X and D_{Y^i} . In order to integrate all such (bi-dimensional) co-clustering subproblems, a linear combination of all pairwise objective

^{*} This paper appeared in Proc. of the 17th European Conf. on Machine Learning, Berlin, 2006

functions is optimized, subject to the constraint that consistent clusters are found for the central (shared) domain. More precisely, for each domain D_{Y^i} , the objective function for co-clustering D_X and D_{Y^i} is weighted with a factor β_i , such that $\sum_{i=1}^N \beta_i = 1$. Extending this setting to arbitrary pairwise interactions mainly requires to equip with a weight, say $\beta_{A,B}$, each pair of (arbitrarily) correlated domains A and B (cf. [2]). Clearly, these weights can strongly affect the quality of the resulting co-clustering.

However, in general, there may be not enough knowledge on the reliability/relevance of pairwise correlation data to set the weights precisely. Hence, some method for their automatic tuning should be defined, as already stated in [6]. In fact, we mainly aim at facing such an open issue, for the specific case of star-structured domains.

In more details, in Section 2, we introduce an information-theoretic framework which generalizes that in [5] and allows to co-cluster an arbitrary number of domains forming a star structure. Notably, this setting fits a wide range of relevant real-world data, like those describing relationships among authors, conferences, papers and keywords in academic publications (with publications constituting the central domain).

In order to address such a problem without using any arbitrary weighting scheme, in Section 3, we propose and study an algorithm that solves the High-Order Co-Clustering by computing Agreements for contrasting Domain objective functions (short: AD-HOCC algorithm). The basic idea is to consider a notion of *agreement*, such that a clustering of the central domain is found guaranteeing that each partial objective function is “close enough” to its optimal value.

Results from test on both synthetic and real data are finally illustrated in Section 4.

2 Formal Framework

Let $D_X = \{x_1, \dots, x_m\}$, $D_{Y^1} = \{y_1^1, \dots, y_{n^1}^1\}, \dots, D_{Y^N} = \{y_1^N, \dots, y_{n^N}^N\}$ be $N + 1$ domains, i.e., sets of values, and let X, Y^1, \dots, Y^N be discrete random variables taking values in $D_X, D_{Y^1}, \dots, D_{Y^N}$, resp. The domains are assumed to form a *star* structure, i.e., the auxiliary domains D_{Y^i} , for $i = 1..N$ are pairwise independent, while each of them is correlated with the central domain D_X . Let then $p_i(X, Y^i)$, with $1 \leq i \leq N$, denote the joint probability distribution between X and Y^i , i.e., $p_i(x, y^i)$ is the probability that X takes the value $x \in D_X$ and Y^i takes the value $y^i \in D_{Y^i}$.

Assume that the values of D_X are to be clustered into k clusters, say $\widehat{D}_X = \{\widehat{x}_1, \widehat{x}_2, \dots, \widehat{x}_k\}$, and those of D_{Y^i} in l_i clusters, say $\widehat{D}_{Y^i} = \{\widehat{y}_1, \widehat{y}_2, \dots, \widehat{y}_{l_i}\}$, for each i in $1..N$. Then, a *high-order co-clustering* for Y^1, \dots, Y^N w.r.t. X is a tuple $C = \langle C_X, C_{Y^1}, \dots, C_{Y^N} \rangle$, such that $C_X : D_X \mapsto \widehat{D}_X$, and $C_{Y^i} : D_{Y^i} \mapsto \widehat{D}_{Y^i}$, for $i = 1 \dots N$. For brevity, for each random variable W and its associated domain D_W , the set of all possible mappings from D_W to its clusters is denoted by $\mathcal{P}(W)$. Moreover $\widehat{W}_{C_W} = C_W(W)$ is the random variable denoting the cluster assigned to W , through the function C_W , defined on D_W and ranging over $\mathcal{P}(W)$. Like in [5], we use lower-case letters for domain elements, and upper-case letters for the random variable ranging over them; in addition, hatted letters are reserved for clusters, and clustered random variables. Also, \widehat{W}_{C_W} will be shortened as \widehat{W} whenever C_W is clear from the context.

Example 1. Let d_1, \dots, d_6 be documents (e.g., academic papers), and t_1, \dots, t_8 be terms. In Fig. 1.(a) the occurrences of terms in documents are represented as edges, while

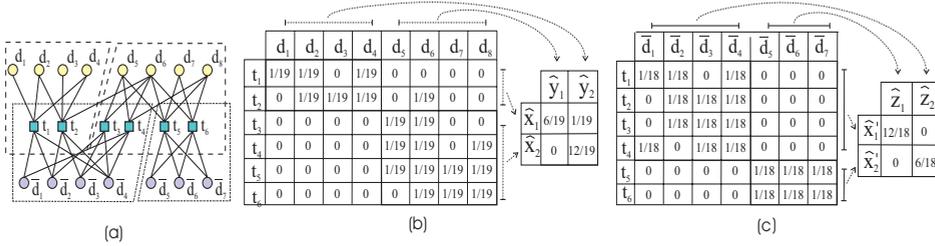


Fig. 1. Co-clustering a text corpus: (a) Spectral and (b-c) Information-theoretic approach.

terms and documents are depicted as nodes (of two different types). The problem can be easily modelled in an information-theoretic framework by defining X and Y to be two random variables taking values in $\{d_1, \dots, d_6\}$ and $\{t_1, \dots, t_8\}$, respectively. Let $p(X, Y)$ be the joint probability distribution between X and Y represented in a tabular form in Fig. 1(b)¹. E.g., $p(d_1, t_2) = \frac{1}{19}$ is the *frequency* of the event of having t_2 occur in document d_1 , while $p(d_1, t_3) = 0$ indicates that t_3 does not occur in d_1 . In this setting we can consider the problem of co-clustering both documents and terms in two clusters, say $\{\hat{x}_1, \hat{x}_2\}$ and, resp., $\{\hat{y}_1, \hat{y}_2\}$. An example co-clustering $\langle C_X, C_Y \rangle$ is shown in the same figure, where C_X is the function mapping d_1 and d_2 to \hat{x}_1 and every other document to \hat{x}_2 , while C_Y maps t_1, \dots, t_4 to \hat{y}_1 and all the other terms to \hat{y}_2 .

Assume that some information is available on document authors, say a_1, \dots, a_7 . Based on authorship data, reported again in Fig. 1(a), we can consider the problem of co-clustering all of the three domains. This is just a *high-order* co-clustering problem over star-structured domains, which essentially amounts to finding a tuple $\langle C_X, C_Y, C_Z \rangle$, where Z is a random variable taking values in $\{a_1, \dots, a_7\}$. If one looks at authors independently of terms, a natural co-clustering is $\langle C'_X, C_Z \rangle$, where C'_X is the function mapping d_1, \dots, d_4 to \hat{x}_1 and every other document to \hat{x}_2 , and C_Z is the function mapping a_1, \dots, a_4 to \hat{z}_1 and every other author to \hat{z}_2 — see Fig. 1(c). \triangleleft

As the effect of co-clustering can be viewed as a sort of information compression, the co-clustering problem can be turned in the search for a fixed-size compression scheme preserving as much as possible of the original mutual information. To this aim, for any auxiliary domain D_{Y^i} , one can compute the *mutual information* $I(X; Y^i)$ between the random variables X and Y^i , ranging over D_X and D_{Y^i} , respectively. Then, the quality of a multi-dimensional co-clustering can be assessed by taking into account the loss of mutual information that occurs when replacing the original variables X, Y^1, \dots, Y^N with their clustered versions $\hat{X}_{C_X}, \hat{Y}^1_{C_{Y^1}}, \dots, \hat{Y}^N_{C_{Y^N}}$. Hereinafter, for brevity, the loss of mutual information pertaining the i -th auxiliary dimension will be denoted by $\Delta I_i(C_X, C_{Y^i}) = I(X; Y^i) - I(\hat{X}_{C_X}; \hat{Y}^i_{C_{Y^i}})$ (or, shortly, ΔI_i).

For the base case of $N = 1$, an algorithm that computes a (locally) optimal co-clustering has been presented in [5], where it was shown that the mutual information loss caused by clustering X and Y^i can be expressed as a dissimilarity between the original joint distribution $p_i(\cdot, \cdot)$ and a function $q_i(\cdot, \cdot)$ that approximates it. More pre-

¹ For simplicity, the joint distribution shown here just is a binary association matrix over terms and documents (where 1's represent the presence of terms), normalized to have 1 as total sum.

cisely: $\Delta I_i = \mathcal{D}(p_i(X, Y^i) || q_i(X, Y^i))$, where $\mathcal{D}(\cdot || \cdot)$ denotes the Kullback-Leibler (KL) divergence, and $q_i(X, Y^i)$ is a function, preserving all marginals of p_i , of the form $q_i(X, Y) = p_i(\widehat{X}, \widehat{Y}^i) \cdot p_i(X | \widehat{X}) \cdot p_i(Y | \widehat{Y}^i)$.

Thus, the pairwise co-clustering problem can be solved by searching for the function q_i that is most similar to p_i , according to \mathcal{D} . To this purpose, an alternate minimization scheme is used in [5] which considers only one dimension per time. In the rest of the paper, we investigate the case of $N > 1$, and define a co-clustering approach that ensures as low as possible values for all the information loss functions ΔI_i .

3 The proposed agreement method (algorithm AD-HOCC)

As discussed previously, a way to jointly optimize the losses of mutual information ΔI_i , for each $1 \leq i \leq N$, is to linearly combine these individual functions into a global one. Thus, one can try to minimize the quantity $\sum_{i=1}^N \beta_i \cdot \Delta I_i$, where β_1, \dots, β_N are suitable weights such that $\sum_{i=1}^N \beta_i = 1$. Since, in general, there may be no knowledge enough to set the coefficients in a precise manner, we abandon the idea of using a pre-fixed weighting scheme and restate the co-clustering problem as a multi-objective optimization of all functions $\Delta I_i = I(X; Y^i) - I(\widehat{X}; \widehat{Y}^i)$.

On the other hand, a major problem occurring while optimizing the mutual information losses ΔI_i , for each $1 \leq i \leq N$, is that the best co-clustering for D_X and D_{Y^i} may not comply with the best co-clustering for D_X and D_{Y^j} for $j \neq i$, as concerns the partition of D_X . For example, this is the case of the data set depicted in Fig. 1. Indeed, when considering the auxiliary dimensions (i.e., terms and authors) independently, the co-clusterings $\langle C_X, C_Y \rangle$ and $\langle C'_X, C_Z \rangle$ introduced in Example 1 seem to be good candidate solutions – this is supported by the graphical representation in Fig. 1(a), where these clusters induce, in fact, an “optimal” cut over the nodes. However, these two optimal bi-clusterings do not conform with each other, since $C_X \neq C'_X$ and, therefore, there is no immediate way for extending them into a global high-order co-clustering. \triangleleft

Therefore, we introduce a notion of agreement to represent a sort of optimal “compromise” among the different (and potentially discordant) goals, which are autonomously pursued by all the bi-dimensional co-clustering subproblems. Actually, it can be shown that computing an *optimal agreement* is NP-hard. Indeed, it requires the computation of the optimal clustering over each dimension alone, which is NP-hard of its own. Accordingly, the following definition states, in a pragmatic way, the notion of agreement under a “local” perspective, only.

Definition 1. Let α be a real number, such that $0 < \alpha \leq 1$. Then a high-order co-clustering $C = \langle C_X, C_{Y^1}, \dots, C_{Y^N} \rangle$ is said to be an α -agreement for Y^1, \dots, Y^N w.r.t. X if, for each Y^i with $1 \leq i \leq N$, the following conditions hold:

- (a) $\forall C' \in \mathcal{P}(Y^i)$, $\Delta I_i(C_X, C') \geq \Delta I_i(C_X, C_{Y^i})$, and
- (b) $\forall C'' \in \mathcal{P}(X)$, $\Delta I_i(C'', C_{Y^i}) \geq \alpha \times [\Delta I_i(C_X, C_{Y^i})]$.

If there exists no $\alpha' > \alpha$ satisfying condition (b), the agreement is said maximal. \square

Notably, α is a sort of quality measure assessing the ability of approximating some local optimum of each function ΔI_i . As an extreme case, when $\alpha = 1$ and $N = 1$, an α -agreement is a local optimum for the two-dimensional co-clustering problem.

<p>Input: Domains $D_X, D_{Y^1}, \dots, D_{Y^N}$, cluster sets $\widehat{D}_X, \widehat{D}_{Y^1}, \dots, \widehat{D}_{Y^N}$, a real number ε, joint distributions $p_1(X, Y^1), \dots, p_N(X, Y^N)$, and the numbers k, l_1, \dots, l_N of clusters required for $D_X, D_{Y^1}, \dots, D_{Y^N}$, resp.;</p> <p>Output: An α-agreement for Y^1, \dots, Y^N w.r.t. X;</p>
<pre> 1 Define an arbitrary co-clustering $\langle C_X^0, C_{Y^1}^0, \dots, C_{Y^N}^0 \rangle$; 2 set $\alpha^{(0)} = 0, t = t_i^* = 0, \Delta I_i^{(0)} = +\infty, \varepsilon_i = \varepsilon, \forall i \in \{1..N\}$; 3 repeat 4 Compute $q_i^{(t)}$, for $i = 1 \dots N$, and set $t = t + 1$; 5 for each Y^i and $y \in D_{Y^i}$ do 6 $C_{Y^i}^{(t)}(y) = \arg \min_{\widehat{g} \in \widehat{D}_{Y^i}} \mathcal{D}(p_i(X y) q_i^{(t-1)}(X \widehat{y}))$; 7 for each Y^i do 8 if $\Delta I_i^{(t)} < \Delta I_i^{(t_i^*)}$ then $t_i^* = t$ else $\varepsilon_i = \varepsilon_i/2$; 9 Compute $q_i^{(t)}$, for $i = 1 \dots N$, and set $t = t + 1$; 10 for each $x \in D_X$ do 11 let $\delta I_i(x, \widehat{x}_j) = \mathcal{D}(p(Y^i x) q_i^{(t-1)}(Y^i \widehat{x}_j)), \forall \widehat{x}_j \in \widehat{D}_X$; 12 let $\alpha(x, \widehat{x}_j) = \min_{Y^i} \frac{\min_{\widehat{x}' \in \widehat{D}_X} \delta I_i(x, \widehat{x}')}{\delta I_i(x, \widehat{x}_j)}, \forall \widehat{x}_j \in \widehat{D}_X$; 13 $C_X^{(t)}(x) = \arg \max_{\widehat{x} \in \widehat{D}_X} \alpha(x, \widehat{x})$; 14 end for 15 let $\alpha^{(t)} = \min_{x \in D_X} \max_{\widehat{x} \in \widehat{D}_X} \alpha(x, \widehat{x})$; 16 let $\Delta I_i^{(t)} = \mathcal{D}(p_i(X^{(t)}, Y^{i(t-1)}) q_i(X^{(t)}, Y^{i(t-1)})), \forall Y^i$; 17 while $(\alpha^{(t)} \simeq \alpha^{(t-2)})$ and $\exists Y^i$ s.t. $(1 - \varepsilon_i) \cdot \Delta I_i^{(t)} > \Delta I_i^{(t_i^*)}$ or $\alpha^{(t)} > \alpha^{(t-2)}$; 18 return $\langle C_X^{(t-2)}, C_{Y^1}^{(t-3)}, \dots, C_{Y^N}^{(t-3)} \rangle$; </pre>

Fig. 2. Algorithm AD-HOCC.

In Fig. 2 algorithm AD-HOCC is shown, which finds a maximal α -agreement, based on a local, alternate, optimization scheme. First an arbitrary co-clustering $\langle C_X^0, C_{Y^1}^0, \dots, C_{Y^N}^0 \rangle$ is computed, which is eventually refined in the main loop. At each repetition t , the optimal clustering $C_{Y^i}^{(t)}$ is computed for each auxiliary domain Y^i ($1 \leq i \leq N$), based on the current clustering for the central domain X . Intuitively, this is carried out with the aim of assigning y to the cluster in \widehat{D}_{Y^i} leading to the minimization of the loss of mutual information. Subsequently, the iteration continues by computing the optimal clustering $C_X^{(t)}$ for the central domain, based on the just computed clusterings for the auxiliary domains. This step is crucial for getting an agreement and is, thus, discussed in more details below.

Preliminary, for each value $x \in D_X$ and each cluster $\widehat{x}_j \in \widehat{D}_X$, we compute the contribution $\delta I_i(x, \widehat{x}_j)$ that would be given to the information loss over each domain Y^i by assigning x to \widehat{x}_j . All these values are normalized w.r.t. the best possible assignment for x , i.e., $\min_{\widehat{x}' \in \widehat{D}_X} \delta I_i(x, \widehat{x}')$, and the worst possible mapping $\alpha(x, \widehat{x}_j)$ is computed.

The algorithm eventually chooses the best over such worst mappings, i.e., an element x is mapped to the cluster $\widehat{x} \in \widehat{D}_X$ maximizing the value $\alpha(x, \widehat{x})$, according to the formula: $C_X^{(t)}(x) = \arg \max_{\widehat{x} \in \widehat{D}_X} \alpha(x, \widehat{x})$.

In addition, the value for $\alpha^{(t)}$ characterizing the guarantee for the agreement is computed according to the formula: $\alpha^{(t)} = \min_{x \in D_X} \max_{\hat{x} \in \hat{D}_X} \alpha(x, \hat{x})$.

The algorithm keeps on iterating till the value $\alpha^{(t)}$ increases, i.e., as long as a better agreement is discovered. Actually, it tolerates that some bi-clustering objective function temporarily get worse, provided that it remains close enough to its best value found so far. To this aim a real number ε is also required in input to denote the range of tolerance admitted. Such a behavior is meant to reduce the risk of underestimating some partial optimum, as the strategy adopted here does not guarantee that every objective function monotonically decreases. Notice, however, that the width of these tolerance ranges is progressively reduced during the search, thus ensuring termination.

Theorem 1. *Let I be the number of iterations, N be the number of auxiliary domains, and M be an upper bound for both the size of any domain and the number of nonzero elements in any joint distribution matrix. Then algorithm AD-HOCC converges, at a step $t = O(N \cdot M \cdot (k + \sum_i l_i) \cdot I)$, to a high-order co-clustering for Y^1, \dots, Y^N w.r.t. X such that $\forall C'_X \in \mathcal{P}(X), C'_{Y^1} \in \mathcal{P}(Y^1), \dots, C'_{Y^N} \in \mathcal{P}(Y^N)$, and for each i in $1..N$:*

- (a) $\Delta I_i(C'_X, C'_{Y^i}) \geq \Delta I_i(C_X^{(t-2)}, C_{Y^i}^{(t-3)})$, and
- (b) $\Delta I_i(C'_X, C'_{Y^i}) \geq \alpha^{(t)} \Delta I_i(C_X^{(t-2)}, C_{Y^i}^{(t-3)})$.

Moreover, there is no $\bar{\alpha} > \alpha^t$ satisfying condition (b).

4 Experiments

This section presents an empirical study of the proposed approach, based on tests on both synthetic and real data, we performed with a Java implementation of algorithm AD-HOCC on a 1600MHz/512MB Pentium IV machine. Synthetic data were produced through an ad-hoc Java generator with the following parameters: (i) the number $N > 1$ of auxiliary domains, (ii) the size of the domains $D_X, D_{Y^1}, \dots, D_{Y^N}$, (iii) the number of required clusters along each domain, (iv) a noise factor θ , and (v) a “disagreement” factor γ . Roughly speaking, the latter basically expresses the maximum percentage of values in D_X that would be assigned to different clusters when considering two different contingency tables. In order to reduce the statistical bias of initial clusterings, all measures have been averaged over 10 runs.

A first series of experiments were conducted to evaluate the capability of algorithm AD-HOCC to achieve a satisfactorily low loss of mutual information for each auxiliary domain, and to contrast it with other co-clustering methods aimed at optimizing an overall objective function, which combine all the bi-clustering sub-problems in a weighted way. Unfortunately, however, the experimental results of such methods [6, 2] available in the literature were always computed by chosen some fixed value for the weights used in the linear combination. Therefore, we decided to implement a “prototypical” co-clustering algorithm, named `Linear-CC`, which adopts such an approach, by trying to minimize the function $\sum_{i=1}^N \beta_i \times \Delta I_i$. To this aim, we simply modified AD-HOCC, in the way it assigns each element of the central domain to the clusters. More precisely, rather than using the formula at line 13 of Figure 2, the optimal clustering $C_X^{(t)}$ is computed as: $C_X^{(t)}(x) = \arg \min_{\hat{x} \in \hat{D}_X} \sum_{i=1}^N \beta_i \cdot \mathcal{D}(p_i(Y^i|x) || q_i^{(t-1)}(Y^i|\hat{x}))$.

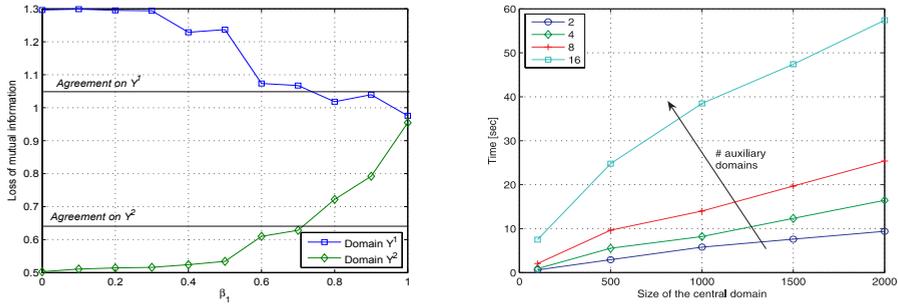


Fig. 3. Loss of mutual information (left) and computation time (right) on synthetic data

On the left side of Fig. 3 the mutual information loss is reported at different values of β_1 ($\beta_2 = 1 - \beta_1$), for a dataset with 2 auxiliary domains, which was built by using a fixed size (1000) and a fixed number of clusters (2) for every domain, and by setting $\gamma = \theta = 0.2$. Note that, for each domain, the information loss produced by AD-HOCC is always (“slightly”) higher than the one found by the linear-combination approach when it just considers that domain (i.e., when $\beta_1 = 1$ for Y^1 , and $\beta_1 = 0$ for Y^2).

In order to assess the scalability of the approach, we computed the time spent against several data sets, all having a fixed size of 100 for auxiliary domains, and generated with $\gamma = 0.1$ and $\theta = 0.05$. Actually, different values have been considered for the size of the central domain (up to 2000 values) and for the total number of domains (up to 16). Results shown in the right side of Fig. 3, confirm the linear dependence of the computation time on both parameters (cf. Theorem 1).

Finally, for validating the proposed approach against real data, we selected two directories, namely *lokay-m* and *williams-w3*, from the preprocessed email datasets available at http://www.cs.umass.edu/~ronb/enron_dataset.html. The resulting data set consists of 275 documents organized in 11 and 18 sub-folders, respectively. We then built the document-by-term and document-by-category matrices, with categories corresponding to the sub-folders. Totally, 1074 terms were selected. Precision results obtained when partitioning the emails in 2 clusters are shown in Fig. 4. More precisely, we report the *micro-averaged precision* measures (also used in, e.g., [2]) computed by assuming that the main directories *lokay-m* and *williams-w3* are “ground truth” emails’ classes. The table also shows results for similar experiments conducted on the directories *kitchen-l* and *sanders-r*. Note that, in both cases, AD-HOCC finds an accurate solution corresponding to some intermediate value of β_t . Indeed, the precision with AD-HOCC is higher than that linear combination get in both extreme scenarios ($\beta_t=0$ and $\beta_t=1$).

	AD-HOCC	Linear-CC	
		$\beta_t=1$	$\beta_t=0$
<i>lokay-m/williams-w3</i>	0.82	0.62	0.62
<i>kitchen-l/sanders-r</i>	0.77	0.75	0.76

Fig. 4. Micro-averaged precision on real data.

5 Conclusions

In this paper we proposed a novel approach to the high-order co-clustering problem (i.e., the problem of simultaneously clustering several heterogeneous domains), in contexts where the domains are correlated in a pairwise way only – in particular, like in [6], we assume that a star-shaped structure of relationships exists.

Previous proposals [2,6] mainly decompose this problem into a series of bi-dimensional (bi-clustering) sub-problems, and try to optimize a linear combination of the respective objective functions, disregarding the issue of properly setting the involved weights. In contrast, based on a novel notion of *agreement* among bi-clustering sub-problems, we have devised an algorithm (AD-HOCC) which does not need any arbitrary weighting scheme and discovers a co-clustering solution, that guarantees low mutual information losses for all the pairs of correlated domains – as it is witnessed by the empirical study presented in the paper. By the way, we also overcame some limitations of [6], where, e.g., only a *bi-partite* co-clustering problem is approached explicitly, whereas our approach can partition each domain in whatever number of clusters.

On the other hand, we observe that our approach does provide for the automatic selection of the number of clusters, which is instead performed by the algorithm proposed in [2], interleaving the top-down clustering of some domains and bottom-up clustering of the others, and adopting a local correction routine. However, this generality comes with a cost: the algorithm in [2] runs in $O(\max_W \{ \log |\widehat{D}_W|, \log(|D_W|/|\widehat{D}_W|) \} \cdot \max_W \{|D_W|^3\})$, where W denotes any input domain, while a quadratic dependence on D_W is ensured only when two domains are to be co-clustered. Conversely, the total computation time required by AD-HOCC only depends linearly on the sizes of the domains and contingency matrices taken as input (cf. Theorem 1), which makes our approach suitable even to large data sets.

References

1. A. Banerjee, I. Dhillon, J. Ghosh, S. Merugu, and D. S. Modha. A generalized maximum entropy approach to bregman co-clustering and matrix approximation. In *Proc. 10th ACM SIGKDD Conf. on Knowledge Discovery and Data Mining (KDD04)*, 509–514, 2004.
2. R. Bekkerman, R. El-Yaniv, and A. McCallum. Multi-way distributional clustering via pairwise interactions. In *Proc. 22nd Intl. Conf. on Machine Learning (ICML05)*, 41–48, 2005.
3. P. Berkhin and J. D. Becher. Learning simple relations: Theory and applications. In *Proc. 2nd SIAM Conf. on Data Mining (SDM02)*, 2002.
4. I. S. Dhillon. Co-clustering documents and words using bipartite spectral graph partitioning. In *Proc. 7th ACM SIGKDD Conf. on Knowledge Discovery and Data Mining (KDD01)*, 269–274, 2001.
5. I. S. Dhillon, S. Mallela, and D. S. Modha. Information-theoretic co-clustering. In *Proc. 9th ACM SIGKDD Conf. on Knowledge Discovery and Data Mining (KDD03)*, 89–98, 2003.
6. B. Gao, T.Y. Liu, X. Zheng, Q.-S. Cheng, and W.-Y. Ma. Consistent bipartite graph co-partitioning for star-structured high-order heterogeneous data co-clustering. In *Proc. 11th ACM SIGKDD Conf. on Knowledge Discovery and Data Mining (KDD05)*, 41–50, 2005.
7. H. Zha, X. He, C. Ding, H. Simon, and M. Gu. Bipartite graph partitioning and data clustering. In *Proc. 10th ACM Intl. Conf. on Information and Knowledge Management (CIKM01)*, 25–32, 2001.

Exploiting Preference Rules for Querying Databases

Sergio Greco, Cristian Molinaro, and Francesco Parisi

DEIS, University of Calabria,
87036 Rende, Italy,
{greco, cmolinaro, fparisi}@deis.unical.it

Abstract. The paper proposes the use of preferences for querying databases. In expressing queries it is natural to express preferences among tuples belonging to the answer. This can be done in commercial DBMS, for instance, by ordering the tuples in the result. The paper presents a different proposal, based on similar approaches deeply investigated in the artificial intelligence field, where preferences are used to restrict the result of queries posed over databases. In our proposal a query over a database \mathcal{DB} is a triplet $\langle q, \mathcal{P}, \Phi \rangle$, where q denotes the output relation, \mathcal{P} a Datalog program (or an SQL query) used to compute the result and Φ is a set of preference rules used to introduce preferences on the computed tuples. In our proposal tuples which are “dominated” by other tuples do not belong to the result and cannot be used to infer other tuples. A new stratified semantics is presented where the program \mathcal{P} is partitioned into strata and the preference rules associated to each stratum of \mathcal{P} are divided into layers; the result of a query is carried out by computing one stratum at time and by applying the preference rules, one layer at time. The proposed technique is sound and the complexity of computing queries with preference rules is still polynomial.

1 Introduction

The growing volume of available information poses new challenges to the database and artificial intelligence communities. Recent researches have investigated new techniques in accessing large volumes of data such as user-centered access to information, information filtering and extraction and policies to reduce data presented to users. An interesting direction deeply studied in the artificial intelligence and nonmonotonic reasoning fields consists in the use of preferences to express priorities on the alternative scenarios.

The paper presents a logical framework wherein preferences are used to restrict the result of queries posed over a database. This is an important aspect in querying large databases such as those used by search engines. In this context, the result of a query contains only tuples which are not *dominated* by other tuples and dominated tuples cannot be used to infer new information. The novelty of the presented approach is that preferences are stratified and applied one stratum at time. A second innovative aspect of this proposal is that preferences on both base and derived atoms are considered as well as general (recursive) queries which can be expressed by means of stratified Datalog.

Example 1. Consider a database $\mathcal{DB} = \{\text{fish}, \text{beef}\}$ and a program \mathcal{P} consisting of the two rules:

```
red-wine ← beef
white-wine ← fish
```

Assume now to have a query defined by the rules in \mathcal{P} and the preference

$$\rho_1 = \text{red-wine} \succ \text{white-wine} \leftarrow \text{beef}$$

stating that if there is beef, we prefer red-wine to white-wine. The set of preferred atoms contains the base atoms fish and beef and the derived atom red-wine (the atom white-wine is not preferred). Assume now to also have the preference $\rho_2 = \text{fish} \succ \text{beef}$ stating that we prefer fish to beef. In this case, first the preference rule ρ_2 , and next the preference rule ρ_1 , are considered. However, ρ_1 cannot be applied as beef is not in the preferred set of atoms. Consequently, the set of preferred atoms, with respect to the preference rules ρ_2 and ρ_1 , is $\{\text{fish}, \text{white-wine}\}$. \square

Contributions. In this paper we study the use of preferences in querying databases. We consider general (stratified) Datalog queries and general preferences: the head of preference rules may contain atoms belonging to different relations and the body consists of a conjunction of literals. A semantics where both query and preferences are partitioned into strata is defined. Under such a semantics, the query is computed one stratum at time and for each stratum (of the query), the preferences are applied one stratum at time.

Related Work. The increased interest in preferences in logic programs is reflected by an extensive number of proposals and systems for preference handling. Most of the approaches propose an extension of logic programming by adding preference information. The most common form of preference consists in specifying a strict partial order on rules [7, 8, 17, 20], whereas more sophisticated forms of preferences also allow priorities to be specified between conjunctive (disjunctive) knowledge with preconditions [4, 17] and numerical penalties for suboptimal options [3].

Considering the use of preferences in querying databases, an extension of relational calculus expressing preferences for tuples in terms of logical conditions has been proposed in [15]. Preferences requiring non-deterministic choice among atoms which minimize or maximize the value of some attribute has been proposed in [10]. An extension of Datalog with preference relations, subsuming the approach proposed in [15], has been proposed in [14], whereas an extension of SQL including preferences has been proposed in [11, 12]. In the last proposal several built-in operators and a formal definition of their combinations (i.e. intersection, union, Pareto composition, etc.) have been considered. Borzsonyi et al. proposed the *skyline* operator [2], to filter out a set of “interesting” point (i.e. not dominated by any other point) from a potential large set of points. An extension of SQL with a skyline operator has been also proposed. A framework for specifying preferences using logical formulas and its embedding into relational algebra has been introduced in [5]. The paper also introduces the *winnow* operator which generalizes the skyline operator. The implementation of winnow and ranking is also studied in [18]. Algorithms for computing skyline operators are also studied in [13, 16, 6]. In [1] the use of quantitative preferences (scoring functions) in queries is proposed. In this work, in contrast with previous proposals, general preferences and a different (stratified) semantics are considered.

2 Background

Familiarity with disjunctive logic programs and disjunctive deductive databases is assumed [19].

Datalog programs. A *term* is either a constant or a variable. An *atom* is of the form $p(t_1, \dots, t_h)$, where p is a *predicate symbol* of arity h and t_1, \dots, t_h are terms. A *literal* is either an atom A or its negation $\text{not } A$. A (*Datalog*) *rule* r is a clause of the form

$$A \leftarrow B_1, \dots, B_m, \text{not } B_{m+1}, \dots, \text{not } B_n, \varphi \quad n \geq 0$$

where A, B_1, \dots, B_n are atoms, whereas φ is a conjunction of built-in atoms of the form $u \theta v$ where u and v are terms and θ is a comparison predicate. A is the *head* of r (denoted by $\text{Head}(r)$), whereas the conjunction $B_1, \dots, B_m, \text{not } B_{m+1}, \dots, \text{not } B_n, \varphi$ is the *body* of r (denoted by $\text{Body}(r)$). It is assumed that each rule is safe, i.e. a variable appearing in the head or in a negative literal also appears in a positive body literal.

A (*Datalog*) *program* is a finite set of rules. A *not-free* program is called *positive*. The *Herbrand Universe* $\mathcal{U}_{\mathcal{P}}$ of a program \mathcal{P} is the set of all constants appearing in \mathcal{P} , and its *Herbrand Base* $\mathcal{B}_{\mathcal{P}}$ is the set of all ground atoms constructed from the predicates appearing in \mathcal{P} and the constants from $\mathcal{U}_{\mathcal{P}}$. A term (resp. an atom, a literal, a rule or a program) is *ground* if no variable occurs in it. A rule r' is a *ground instance* of a rule r if r' is obtained from r by replacing every variable in r with some constant in $\mathcal{U}_{\mathcal{P}}$; $\text{ground}(\mathcal{P})$ denotes the set of all ground instances of the rules in \mathcal{P} .

An *interpretation* M for a Datalog program \mathcal{P} is any subset of $\mathcal{B}_{\mathcal{P}}$; M is a *model* of \mathcal{P} if it satisfies all rules in $\text{ground}(\mathcal{P})$. The (model-theoretic) semantics for positive \mathcal{P} assigns to \mathcal{P} the set of its *minimal models* $\mathcal{MM}(\mathcal{P})$, where a model M for \mathcal{P} is minimal if no proper subset of M is a model for \mathcal{P} . For any interpretation M , \mathcal{P}^M is the ground positive program derived from $\text{ground}(\mathcal{P})$ by 1) removing all rules that contain a negative literal $\text{not } A$ in the body and $A \in M$, and 2) removing all negative literals from the remaining rules. An interpretation M is a *stable model* of \mathcal{P} if and only if $M \in \mathcal{MM}(\mathcal{P}^M)$ [9]. For general \mathcal{P} , the stable model semantics assigns to \mathcal{P} the set $\mathcal{SM}(\mathcal{P})$ of its stable models. It is well-known that stable models are minimal models (i.e. $\mathcal{SM}(\mathcal{P}) \subseteq \mathcal{MM}(\mathcal{P})$) and that for negation free programs minimal and stable model semantics coincide (i.e. $\mathcal{SM}(\mathcal{P}) = \mathcal{MM}(\mathcal{P})$).

Given a Datalog program \mathcal{P} , $\mathcal{G}_g(\mathcal{P}) = (V_g, E_g)$ denotes the dependency graph associated with $\text{ground}(\mathcal{P})$ where V_g consists of all ground atoms appearing in $\text{ground}(\mathcal{P})$, whereas there is an arc from B to A in E_g if there is a rule r in $\text{ground}(\mathcal{P})$ such that $\text{Head}(r) = A$ and $B \in \text{Body}(r)$; the arc is said to be marked negatively if B appears negated in the body of r . The dependency graph $\mathcal{G}(\mathcal{P}) = (V, E)$ associated with \mathcal{P} is built by considering the ground program derived from \mathcal{P} by eliminating all terms (i.e. every atom $p(t)$ is replaced by p). A ground atom $p(t)$ *depends* on a ground atom $q(u)$ if there is a path in $\mathcal{G}_g(\mathcal{P})$ from $q(u)$ to $p(t)$. Analogously, a predicate symbol p *depends* on a predicate symbol q if there is a path in $\mathcal{G}(\mathcal{P})$ from q to p . The dependency is negated if there is an arc marked negatively in the path.

A partition π_0, \dots, π_k of the set of all predicate symbols of a Datalog program \mathcal{P} , where each π_i is called *stratum*, is a *stratification* of \mathcal{P} if for each rule r in \mathcal{P} the predicates that appear only positively in the body of r are in strata lower than or equal to the

stratum of the predicate in the head of r , and the predicates that appear negatively are in strata lower than the stratum of the predicate in the head of r . The stratification of the predicates defines a stratification of the rules of \mathcal{P} into strata $\langle \mathcal{P}_1, \dots, \mathcal{P}_k \rangle$ where a stratum \mathcal{P}_i contains rules which define predicates in π_i . A Datalog program is called *stratified* if it has a stratification. Stratified (normal) programs have a unique stable model which coincides with the *stratified model*, obtained by computing the fixpoints of every stratum in their order.

Queries. Predicate symbols are partitioned into two distinct sets: *base predicates* and *derived predicates*. Base predicates correspond to database relations defined over a given domain and they do not appear in the head of any rule, whereas derived predicates are defined by means of rules. Given a set of ground atoms \mathcal{DB} , a predicate symbol p and a stratified program \mathcal{P} , $\mathcal{DB}[p]$ denotes the set of p -tuples in \mathcal{DB} , while $\mathcal{P}_{\mathcal{DB}}$ denotes the program derived from the union of \mathcal{P} with the facts in \mathcal{DB} , i.e. $\mathcal{P}_{\mathcal{DB}} = \mathcal{P} \cup \mathcal{DB}$. The semantics of $\mathcal{P}_{\mathcal{DB}}$ is given by the stratified model (which coincide with the unique stable model) of $\mathcal{P}_{\mathcal{DB}}$. The answer to a query $Q = (g, \mathcal{P})$ over a database \mathcal{DB} , denoted by $Q(\mathcal{DB})$, is given by $\mathcal{M}[g]$ where $\mathcal{M} = \mathcal{SM}(\mathcal{P}_{\mathcal{DB}})$. In the following we also denote with $\mathcal{P}(\mathcal{DB}) = \mathcal{SM}(\mathcal{P}_{\mathcal{DB}})$ the application of \mathcal{P} to \mathcal{DB} ; therefore $Q(\mathcal{DB}) = \mathcal{P}(\mathcal{DB})[g]$.

3 Preference Rules and Queries

This section presents a framework for expressing preferences in the evaluation of queries posed on a given database. The framework is based on the introduction of *preference rules*, whose syntax is inspired to the management of priorities in the artificial intelligence field, logic programming and database querying [4, 7, 8, 17, 20].

3.1 Syntax

A *prioritized program* consists of a set of standard rules (Datalog program) and a set of preference rules. As rules expressing preferences eliminate tuples which are derived by means of standard rules (Datalog program) we first introduce a standard stratification of the Datalog program to fix the order in which standard rules are applied. Preference rules are associated to each subprogram (stratum) and applied after the subprogram has been evaluated. Let start by introducing the concept of standard stratification.

Definition 1. The *standard stratification* of a stratified program \mathcal{P} consists of k strata $\langle \mathcal{P}_1, \dots, \mathcal{P}_k \rangle$ where k is the minimal value such that for each \mathcal{P}_i and for each pair of predicates p and q defined in \mathcal{P}_i either they are mutually recursive or they are independent (i.e. p does not depend on q and q does not depend on p). \square

In the following, given an atom $p(t)$, $str(p(t))$ denotes the stratum of the predicate symbol p (or equivalently of the subprogram in which p is defined) in the standard stratification.

Definition 2. A *preference rule* ρ is of the form:

$$A \succ C \leftarrow B_1, \dots, B_m, \text{not } B_{m+1}, \dots, \text{not } B_n, \varphi \quad (1)$$

where A, C, B_1, \dots, B_n are atoms, and φ is a conjunction of built-in atoms. \square

Also in this case we assume that rules are safe. In the above definition $A \succ C$ is called head of the preference rule (denoted as $Head(\rho)$), whereas the conjunction $B_1, \dots, B_m, \text{not } B_{m+1}, \dots, \text{not } B_n, \varphi$ is called body (denoted as $Body(\rho)$). Moreover, we denote with $Head_1(\rho)$ and $Head_2(\rho)$ the first and the second atom in the head of ρ , respectively (i.e. $Head_1(\rho) = A$ and $Head_2(\rho) = C$).

The intuitive meaning of a ground preference rule ρ is that if the body of ρ is true, then the atom A is *preferable* to C (we also say that the atom C is *dominated* by the atom A). This means that in the evaluation of a prioritized program $\langle \mathcal{P}, \Phi \rangle$ the model defining its semantics cannot contain the atom C if it contains the atom A and the body of the preference rule is true.

Let Φ be a *preference program*, i.e. a set of preference rules. The transitive closure of $ground(\Phi)$ is $\Phi_g^* = ground(\Phi) \cup \{(A \succ C \leftarrow body_1, body_2 \mid \exists A \succ B \leftarrow body_1 \in \Phi_g^* \wedge \exists B \succ C \leftarrow body_2 \in \Phi_g^*)\}$. Analogously, we define Φ^* as the closure of the set of ground preference rules derived from Φ by replacing every atom $p(t)$ with p and deleting built-in atoms.

Definition 3. A (ground) preference program Φ_g^* is *layered* if it is possible to partition it into n layers $\langle \Phi_g^*[1], \dots, \Phi_g^*[n] \rangle$ as follows:

- For each ground atom A such that there is no ground rule $\rho \in \Phi_g^*$ such that $Head_2(\rho) = A$, $layer(A) = 0$;
- For every ground atom C such that there is a rule ρ of the form (1) (i.e. such that $Head_2(\rho) = C$), $layer(C) > \max\{layer(B_1), \dots, layer(B_n), 0\}$ and $layer(C) \geq layer(A)$;
- The layer of a preference rule $\rho \in \Phi_g^*$, denoted as $layer(\rho)$, is equal to $layer(Head_2(\rho))$;
- $\Phi_g^*[i]$ consists of all preference rules associated with the layer i . \square

Example 2. Consider the set of preference rules $\Phi = \{\rho_1 : \text{fish} \succ \text{beef} \leftarrow, \rho_2 : \text{red-wine} \succ \text{white-wine} \leftarrow \text{beef}, \rho_3 : \text{white-wine} \succ \text{red-wine} \leftarrow \text{fish}\}$. The transitive closure Φ^* consists of the rules ρ_1, ρ_2, ρ_3 plus the rules $\{\rho_4 : \text{red-wine} \succ \text{red-wine} \leftarrow \text{beef}, \text{fish}, \rho_5 : \text{white-wine} \succ \text{white-wine} \leftarrow \text{fish}, \text{beef}\}$. Then Φ^* is partitioned into the two layers $\Phi^*[1] = \{\rho_1\}$ and $\Phi^*[2] = \{\rho_2, \rho_3, \rho_4, \rho_5\}$. \square

As it will be clear in the next subsection, preference rules of the form $A \succ A \leftarrow body$ are useless and can be deleted. Therefore, in the above example $\Phi^*[2] = \{\rho_2, \rho_3\}$.

Example 3. Consider the following set Φ of preference rules:

$$\rho_1 : \text{fish} \succ \text{beef} \leftarrow \text{white-wine} \quad \text{and} \quad \rho_2 : \text{red-wine} \succ \text{white-wine} \leftarrow \text{beef}$$

According to ρ_1 the layer of beef must be greater than the layer of white-wine , whereas according to ρ_2 the layer of white-wine must be greater than the layer of beef . Thus, the set of preference rules is not layered. \square

Observe that in the above definition, in order to compute the closure of the ground instantiation of Φ , we need to know the database \mathcal{DB} containing all constants in the database domain. Therefore, checking whether Φ_g^* can be partitioned into layers cannot be done at compile-time. It is possible to define sufficient conditions which guarantee

that the set of preference rules can be partitioned into layers by considering the (ground) program Φ^* instead of the program Φ_g^* . This means that if Φ^* can be partitioned into layers, the set Φ_g^* can be partitioned into layers as well, although the layers of Φ_g^* may be different from the layers of Φ^* (the layers of Φ_g^* define a “refinement” of the layers of Φ^*).

Definition 4. A *prioritized query* is of the form $\langle q, \mathcal{P}, \Phi \rangle$ where q is a predicate symbol denoting the output relation, \mathcal{P} is a (stratified) Datalog program and Φ is a set of preference rules. \square

As said before, the intuitive meaning of a prioritized query $\langle q, \mathcal{P}, \Phi \rangle$ over a database \mathcal{DB} is that the atoms derived from \mathcal{P} and \mathcal{DB} must satisfy the preference conditions defined in Φ .

Definition 5. A prioritized query $Q = \langle q, \mathcal{P}, \Phi \rangle$ is said to be *well formed* if Φ_g^* is layered and for every ground atom C such that there is a rule ρ of the form (1) (i.e. such that $Head_2(\rho) = C$) it holds that

1. $str(C) \geq \max\{str(A), str(B_1), \dots, str(B_n)\}$, and
2. A, B_1, \dots, B_m do not depend on C in \mathcal{P} . \square

In the following we assume that our queries are well formed. Sufficient conditions can be defined on the base of the dependency graph $\mathcal{G}(\mathcal{P})$.

3.2 Semantics

First we analyze the case where Φ defines preferences on databases atoms and next we consider the case where Φ expresses preferences on base and derived atoms, i.e. also on atoms defined in \mathcal{P} .

Preferences on Base Atoms. It is assumed here to have a query $Q = \langle q, \mathcal{P}, \Phi \rangle$ and that the preference rules in Φ express preferences only among base atoms. As said before, Φ_g^* can be partitioned into n layers $\widehat{\Phi}_g^* = \langle \Phi_g^*[1], \dots, \Phi_g^*[n] \rangle$.

Definition 6. Let \mathcal{DB} be a set of ground atoms, Φ a set of preference rules such that $\widehat{\Phi}_g^* = \langle \Phi_g^*[1], \dots, \Phi_g^*[n] \rangle$, and t, u two atoms in \mathcal{DB} . We say that t is *preferable* to u with respect to $\Phi_g^*[i]$ (denoted as $t \sqsupset_{\Phi[i]} u$) if

- $\exists (t \succ u \leftarrow body_1) \in \Phi_g^*[i]$ s.t. $\mathcal{DB} \models body_1$, and
- $\nexists (u \succ t \leftarrow body_2) \in \Phi_g^*[i]$ s.t. $\mathcal{DB} \models body_2$.

The set of tuples in \mathcal{DB} which are *preferred* with respect to $\Phi_g^*[i]$ is $\Phi_g^*[i](\mathcal{DB}) = \{t \mid t \in \mathcal{DB} \wedge \nexists u \in \mathcal{DB} \text{ s.t. } u \sqsupset_{\Phi[i]} t\}$. \square

Observe that Φ_g^* could contain preference rules of the form $A \succ A \leftarrow body$. Such preferences are useless as they are not used to infer preferences among ground atoms and can be deleted from Φ_g^* .

Example 4. Consider the database $\mathcal{DB} = \{\text{fish, beef, red-wine, white-wine, pie, ice-cream}\}$ and the set of preference rules $\Phi = \{\rho_1 : \text{pie} \succ \text{ice-cream} \leftarrow, \rho_2 : \text{red-wine} \succ \text{white-wine} \leftarrow \text{fish}, \rho_3 : \text{white-wine} \succ \text{red-wine} \leftarrow \text{beef}\}$. The set Φ_g^* consists, without considering useless rules, of a unique layer $\Phi_g^*[1] = \{\rho_1, \rho_2, \rho_3\}$. The application of $\Phi_g^*[1]$ to \mathcal{DB} gives the set $\Phi_g^*[1](\mathcal{DB}) = \{\text{fish, beef, red-wine, white-wine, pie}\}$. \square

Definition 7. Let \mathcal{DB} be a database and $Q = \langle q, \mathcal{P}, \Phi \rangle$ be a query such that Φ expresses preferences only on base atoms and the set of ground preference rules Φ_g^* is layered into $\widehat{\Phi}_g^* = \langle \Phi_g^*[1], \dots, \Phi_g^*[n] \rangle$. Then the set of preferred tuples with respect to $\widehat{\Phi}_g^*$ is

$$\mathcal{M} = \mathcal{P}(\widehat{\Phi}_g^*(\mathcal{DB})) = \mathcal{P}(\Phi_g^*[n](\Phi_g^*[n-1] \dots (\Phi_g^*[1](\mathcal{DB})) \dots)).$$

The answer to the query Q is given by $\mathcal{M}[q]$. \square

Example 5. Consider the database $\mathcal{DB} = \{\text{fish, beef, red-wine, white-wine, pie}\}$ and the preference rules Φ of Example 2. Φ_g^* is equal to Φ and it is layered into $\widehat{\Phi}_g^* = \langle \Phi_g^*[1], \Phi_g^*[2] \rangle = \langle \{\rho_1\}, \{\rho_2, \rho_3\} \rangle$. The application of $\Phi_g^*[1]$ to \mathcal{DB} gives the set $\mathcal{M}_1 = \Phi_g^*[1](\mathcal{DB}) = \{\text{fish, red-wine, white-wine, pie}\}$. The application of $\Phi_g^*[2]$ to \mathcal{M}_1 gives the set $\mathcal{M}_2 = \Phi_g^*[2](\mathcal{M}_1) = \{\text{fish, white-wine, pie}\}$. \square

General Preferences. We consider now general prioritized queries $Q = \langle q, \mathcal{P}, \Phi \rangle$ where \mathcal{P} is a stratified Datalog program and Φ expresses preferences also on derived atoms.

Let $\langle q, \mathcal{P}, \Phi \rangle$ be a prioritized query and \mathcal{DB} a database. Let $\langle \mathcal{P}_1, \dots, \mathcal{P}_k \rangle$ be the standard stratification of $\text{ground}(\mathcal{P})$ and let $\mathcal{P}_0 = \{A \leftarrow \mid A \in \mathcal{DB}\}$. Then, $\Phi_g^*[\mathcal{P}_i]$, for $i \in [0..k]$, denotes the following set of preference rules in Φ_g^* :

$$\Phi_g^*[\mathcal{P}_i] = \{A \succ C \leftarrow \text{body} \mid \exists (C \leftarrow \text{body}') \in \mathcal{P}_i\}$$

Definition 8. Let \mathcal{DB} be a database and let $Q = \langle q, \mathcal{P}, \Phi \rangle$ be a prioritized query and $\langle \mathcal{P}_1, \dots, \mathcal{P}_k \rangle$ the standard stratification of \mathcal{P} . The application of \mathcal{P} and Φ to \mathcal{DB} is defined as follows:

$$\begin{aligned} \mathcal{M}_0 &= \widehat{\Phi}_g^*[\mathcal{P}_0](\mathcal{DB}) \\ \mathcal{M}_i &= \widehat{\Phi}_g^*[\mathcal{P}_i](\mathcal{P}_i(\mathcal{M}_{i-1})) \quad \text{for } i \in [1..k] \end{aligned}$$

In the following \mathcal{M}_k will be denoted as $\langle \mathcal{P}, \Phi \rangle(\mathcal{DB})$. The answer to the query Q over the database \mathcal{DB} , denoted as $Q(\mathcal{DB})$, is given by $\mathcal{M}_k[q]$. \square

Before providing a comprehensive example, we present two theorems regarding the soundness and complexity of our proposal.

Theorem 1. Let \mathcal{DB} be a database, $Q = \langle q, \mathcal{P}, \Phi \rangle$ a prioritized query and let $M = \langle \mathcal{P}, \Phi \rangle(\mathcal{DB})$. For each ground preference rule $A \succ C \leftarrow \text{body}$ in Φ_g^* , if $M \models (\text{body} \wedge A)$ then $M \not\models C$. \square

Theorem 2. Let \mathcal{DB} be a database and $Q = \langle q, \mathcal{P}, \Phi \rangle$ a prioritized query. The computational complexity of $Q(\mathcal{DB})$ is polynomial time. \square

Example 6. Consider the following database \mathcal{DB} consisting of the single relation `menu`:

```

menu(tuna, fish),          menu(meat_stew, meat),
menu(sole, fish),         menu(red_wine, drink),
menu(sword_fish, fish),   menu(white_wine, drink)
menu(sausage, meat),

```

Let \mathcal{P} be the program containing the following rules:

```

r1: lunch(red_wine)   ← menu(X, meat), menu(red_wine, drink)
r2: lunch(white_wine) ← menu(X, fish), menu(white_wine, drink)
r3: lunch(sausage)    ← menu(sausage, meat)
r4: lunch(meat_stew) ← menu(meat_stew, meat)
r5: lunch(tuna)       ← menu(tuna, fish)
r6: eaten(T)          ← lunch(X), menu(X, T), T ≠ drink
r7: dinner(X)         ← menu(X, T), not eaten(T)

```

Let Φ be the preference program containing the following preference rules:

```

ρ1: lunch(sausage) > lunch(tuna)           ← lunch(meat_stew)
ρ2: lunch(red_wine) > lunch(white_wine)    ← lunch(sausage)
ρ3: lunch(white_wine) > lunch(red_wine)    ← lunch(tuna)
ρ4: dinner(tuna) > dinner(sole)            ←
ρ5: dinner(red_wine) > dinner(white_wine) ← dinner(sausage)
ρ6: dinner(white_wine) > dinner(red_wine) ← dinner(tuna)

```

The standard stratification of \mathcal{P} is $\langle \mathcal{P}_1, \mathcal{P}_2, \mathcal{P}_3 \rangle$, where $\mathcal{P}_1 = \{r_1, r_2, r_3, r_4, r_5\}$, $\mathcal{P}_2 = \{r_6\}$ and $\mathcal{P}_3 = \{r_7\}$. Therefore $\Phi_g^*[\mathcal{P}_1] = \{\rho_1, \rho_2, \rho_3\}$, $\Phi_g^*[\mathcal{P}_2] = \emptyset$ and $\Phi_g^*[\mathcal{P}_3] = \{\rho_4, \rho_5, \rho_6\}$. Moreover, we obtain: $\widehat{\Phi}_g^*[\mathcal{P}_1] = \langle \Phi_g^*[\mathcal{P}_1][1], \Phi_g^*[\mathcal{P}_1][2] \rangle = \langle \{\rho_1\}, \{\rho_2, \rho_3\} \rangle$, $\widehat{\Phi}_g^*[\mathcal{P}_2] = \langle \emptyset \rangle$ and $\widehat{\Phi}_g^*[\mathcal{P}_3] = \langle \Phi_g^*[\mathcal{P}_3][1], \Phi_g^*[\mathcal{P}_3][2] \rangle = \langle \{\rho_4\}, \{\rho_5, \rho_6\} \rangle$.

Let $Q = \langle \text{dinner}, \mathcal{P}, \Phi \rangle$ be a prioritized query. The answer to Q posed on \mathcal{DB} is obtained as follows.

```

P1(DB) = DB ∪ { lunch(red_wine), lunch(white_wine), lunch(sausage),
                lunch(meat_stew), lunch(tuna) };
M1 = Φg*[P1](P1(DB)) = DB ∪ { lunch(red_wine), lunch(sausage), lunch(meat_stew) };
P2(M1) = M1 ∪ { eaten(meat) };
M2 = Φg*[P2](P2(M1)) = P2(M1);
P3(M2) = M2 ∪ {                dinner(tuna), dinner(sole), dinner(sword_fish),
                dinner(red_wine), dinner(white_wine) };
M3 = Φg*[P3](P3(M2)) = M2 ∪ { dinner(tuna), dinner(sword_fish),
                dinner(white_wine) }

```

Thus the answer to Q over \mathcal{DB} is:

```

Q(DB) = M3[dinner] = { dinner(tuna), dinner(sword_fish), dinner(white_wine) }.

```

4 Conclusions

This paper has introduced *prioritized queries*, a form of queries well-suited for expressing preferences among tuples either belonging to the source database or derived by

means of the program specified in the query. It has been shown that prioritized queries are well-suited to express queries wherein we are interested only in *preferred tuples*. A stratified semantics for computing prioritized queries has been presented where the program \mathcal{P} is partitioned into strata and the preference rules associated to each stratum of \mathcal{P} are divided into layers; a query is evaluated by computing one stratum at time and by applying the preference rules, one layer at time. The computational complexity of computing prioritized queries remains polynomial.

References

1. Agrawal, R., Wimmers, E. L., A framework for expressing and combining preferences, *Proc. ACM SIGMOD International Conference on Management of Data*, 297–306, 2002.
2. Borzsonyi, S., Kossmann, D., Stocker, K., The skyline operator, *Proc. International Conference on Data Engineering (ICDE)*, 421–430, 2001.
3. Brewka, G., Complex Preferences for Answer Set Optimization, *Proc. Principles of Knowledge Representation and Reasoning (KR)*, 213–223, 2004.
4. Brewka, G., Niemela, I., Truszczynski, M., Answer Set Optimization, *Proc. International Joint Conference on Artificial Intelligence (IJCAI)*, 867–872, 2003.
5. Chomicki, J., Preference Formulas in Relational Queries, *ACM Transactions on Database Systems (TODS)*, Vol. 28(4), 1–40, 2003.
6. Chomicki, J., Godfrey, P., Gryz, J., Liang, D., Skyline with presorting, *Proc. International Conference on Data Engineering (ICDE)*, 717–816, 2003.
7. Delgrande, J., P., Schaub, T., Tompits, H., A Framework for Compiling Preferences in Logic Programs, *Theory and Practice of Logic Programming (TPLP)*, Vol. 3(2), 129–187, 2003.
8. Gelfond, M., Son, T.C., Reasoning with prioritized defaults, *Proc. Logic Programming and Knowledge Representation (LPKR)*, 164–223, 1997.
9. Gelfond, M., Lifschitz, V., The Stable Model Semantics for Logic Programming, *Proc. International Conference Logic Programming (ICLP)*, 1070–1080, 1988.
10. Greco, S., Zaniolo, C., Ganguly, S., Greedy by Choice, *Proc. Symposium on Principles of Database Systems (PODS)*, 105–113, 1992.
11. Kießling, W., Foundations of preferences in database systems, *Proc. International Conference on Very Large Data Bases (VLDB)*, 311–322, 2002.
12. Kießling, W., Kostler, G., Preference SQL - Design, Implementation, experience, *Proc. International Conference on Very Large Data Bases (VLDB)*, 990–1001, 2002.
13. Kossmann, D., Ramsak, F., Rost, S., Shooting stars in the sky: An online algorithm for skyline queries. *Proc. International Conference on Very Large Data Bases*, 275–286, 2002.
14. Kostler, G., Kießling, W., Thone, H., Guntzer, U., Fixpoint iteration with subsumption in deductive databases. *Journal of Intelligent Information Systems*, Vol. 4, 123–148, 1995.
15. Lacroix, M., Lavency, P., Prefences: Putting More Knowledge Into Queries, *Proc. International Conference on Very Large Data Bases (VLDB)*, 217–225, 1987
16. Papadias, D., Tao, Y., Fu, G., Seeger, B., An optimal and progressive algorithm for skyline queries, *Proc. ACM SIGMOD International Conf. on Management of Data*, 467–478, 2003.
17. Sakama, C., Inoue, K., Prioritized logic programming and its application to commonsense reasoning, *Artificial Intelligence*, Vol. 123(1-2), 185–222, 2000.
18. Torlone, R., Ciaccia, P., Finding the Best when it's a Matter of Preference, *Proc. Italian Symposium on Advanced Database Systems (SEBD)*, 347–360, 2002.
19. Ullman, J. K., *Principles of Database and Knowledge-Base Systems*, Vol. 1, Computer Science Press, 1988.
20. Zhang, Y., Foo, N., Answer sets for prioritized logic programs, *Proc. International Symposium on Logic Programming (ILPS)*, 69–83, 1997.

Semantic Web Service Composition in the NeP4B Project: Challenges and Architectural Issues^{*}

Federica Mandreoli¹, Wilma Penzo², and Antonio Massimiliano Perdichizzi¹

¹ DII

Università di Modena e Reggio Emilia
Modena, Italy

{mandreoli.federica, antonio.perdichizzi}@unimo.it

² DEIS

Università di Bologna
Bologna, Italy

wpenzo@deis.unibo.it

Abstract. Semantic Web service discovery and composition frameworks proposed so far assume for the most part a centralized registry that holds information of all the Web services available at any given time. This solution does not well cope with the scalability and flexibility requirements of dynamic, fast changing contexts. As part of the NeP4B project, in this paper we propose an alternative peer to peer architecture based on the Goal concept.

1 Introduction

Service Oriented Architectures (SOA) and Web services (WS) as a way to realize the formers have been in both the industrial and scientific focus for many years. They provide a new perspective to look at the Internet and at its potentials for supporting business.

Currently the Internet is mainly a collection of information but does not yet support processing this information. Recent effort around the Web services try to lift the Internet to a new level of service enabling full cooperation and integration between users. The ultimate vision is to discover, invoke and compose Web services to create new complex services fully automatically.

Proposed standardization of basic WS capabilities, i.e. communication (SOAP), description (WSDL) and discovery (UDDI)[17], only address part of the overall stack that needs to be available in order to eventually achieve large scale interoperation of Web services. Fundamental to cope with this issue is the need to make such services computer interpretable, that is to create a Semantic Web of services whose properties, capabilities, interfaces and effects are encoded in an unambiguous form [15]. Semantic Web Services (SWS) combine Semantic

^{*} This work has been partially supported by the Italian Council co-founded FIRB Project: “NeP4B: Networked Peers for Business”.

Web research efforts with the Web services world. Their strength lie in being machine-accessible, as they are founded on the Web Service technology, and machine-understandable, as part of the Semantic Web vision.

The NeP4B project is an Italian Council co-founded FIRB project whose main aim is to investigate and design a technology as the basis for creating a network of semantic peers, providing advanced data-driven semantic services for B2B applications, where companies can classify their own profiles, offers, services and other features so as to gain public visibility to potential customers and partners. These semantic peers will cooperate on a P2P basis to deliver the targeted semantic services to all the users of the semantic peer infrastructure. As part of the NeP4B project, we are interested in defining a SWS scalable and flexible architecture to support large scale service interoperation, and particularly composition, in a broad and heterogeneous environment, where each user may as well be a provider of his own services and a consumer.

Starting from this context definition (Section 2), we aim to propose a possible architectural solution to face the identified challenges in the NeP4B project (Section 3). Finally, we provide conclusions and future work (Section 4).

2 Motivations

In this Section, we first introduce the architectural challenges to be faced in the NeP4B Project for supporting large scale service interoperation. We then present the core concepts of our proposal.

2.1 The NeP4B Scenario

To compete in the global market context, ICTs are a key asset to gain and sustain competitive advantage. Yet, for small and medium-sized enterprises (SMEs), the high cost level of an IT project and the high risk rate involved in such projects [2], represent a quite insuperable barrier to technological innovation.

Taking this into account, NeP4B aims to develop an advanced technological infrastructure to support SMEs by enabling them to search for partners, exchange data, negotiate and collaborate without limitations and constraints, regardlessly of nature, size and geographic location. In order to do this, NeP4B relies on the concept of semantic peers constituting a virtual network (see Fig. 1) of intelligent, trusted and distributed data-driven services, with high added value.

In this context, peers can be single SMEs as well as mediators representing groups of companies. Each semantic peer may as well be a provider of its own services and a consumer. It is fully autonomous of participating to the network and exposing Web services on the basis of its own internal and external business needs.

Semantic Web Services are powered by the underlying data, which are collections of structure, semi-structured, unstructured and multimedia data and which are described by the peer's schema (PS). In such a cooperative context, data can

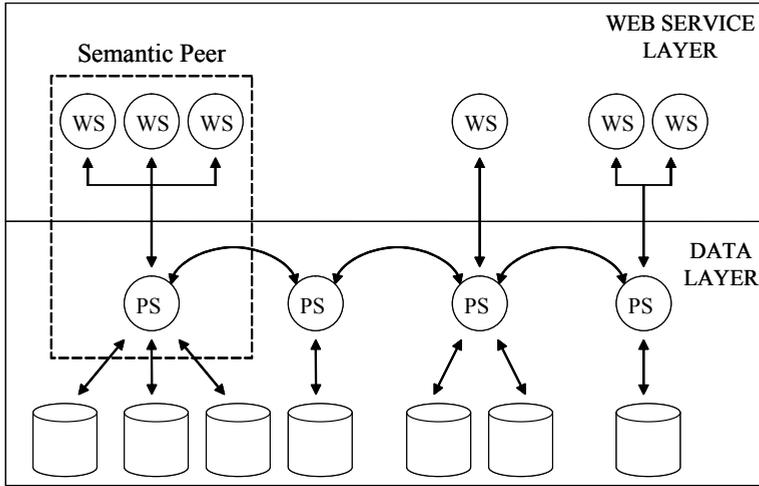


Fig. 1. NeP4B Architectural Layers

also be shared among peers in order to allow services, particularly information providing services, to collect information by spanning over the network. To this end, mappings between schemas are provided locally between pairs of peers. In Figure 1, this layer is referred as the data layer. On top of this layer, there is the Web service layer consisting in the data-driven SWSs that a given peer offers. Each SWS is marked up by adopting an ontology language such as DAML/OWL-S [8] and refers to the underlying data by using the local dictionary of the peer's schema.

In this distributed and heterogeneous context, services have to be searchable to meet users' requests and invocable both for a stand alone execution or an automatic composition process [13, 3]. The composition issue has a crucial role: Being able to automatically re-use existing services to generate new services would allow scalability, flexibility and effectiveness of the network, reducing human effort otherwise needed. To support it, in this paper we provide a broad framework to overcome the heterogeneity of dictionaries and the lack of shared service knowledge due to the autonomy of peers in our dynamic environment.

2.2 Our contributions

Much of the work done on Web service architectures proposes solutions based on centralized registries, such as UDDI [1], where every Web service coming on line advertises its capabilities and functionalities with the registry. Centralized control of published services allows to ease discovery and composition of services but it suffers from the traditional weaknesses of centralized systems, namely single point of failure, and performance bottlenecks.

An alternative to this approach is provided by P2P computing, where Web services interact with each other dynamically, without any centralized control.

In such a context, there have been several proposals for Semantic Web service discovery (e.g. [10, 18]). However, this alternative does not well support automatic composition of services because it does not provide a known and definite service space. There have been several proposals to overcome this problem by either trolling both the construction of the overlay network and the location of data within the system, i.e. structured systems as Chord (e.g. [7]), or only defining its topology ex-ante (e.g. [12]).

Finally, it should be noted that both approaches rely on a common service dictionary for composition to take place.

Considering all of the above, we propose a hybrid approach which exploits the advantages of centralized registries for service discovery and composition, as well as the dynamism of non-structured P2P networks that ensure the peers' full autonomy. The main idea is that, while not centralizing the knowledge of what specific services are available in the system, we keep centralized knowledge of what objectives may be satisfied within the network, namely *Goals*. A Goal is the conceptualization of a domain of services whose ultimate aims are identical or similar. For example, all services that, in the same geographical area provide driving directions from a departure point to a destination. Some could allow the requester to choose among different route options, for example the shortest or the most economical one, while others would only provide the fastest. Even if they may have different reference dictionaries, specific requirements or functionalities, they do answer the same requestor's need: To provide driving directions within a certain geographical area. Therefore they would be well represented by the same goal. Each Goal specifies therefore a subnetwork of specific services, and it is stored in an appropriate repository, called *Goal Repository*.

Using a repository of Goals has several advantages that do not come with loss of flexibility or scalability. Firstly, Goals constitute the domain for the composition purposes. Now its dimension is greatly reduced w.r.t. the underlying service level as goals only represent the objectives the network is able to satisfy rather than how they are satisfied. In this way, we move the issues of discovery and defining a composition pattern from service level to Goal level, namely *Goal discovery* and *Goal composition* respectively. Once Goals have been identified, it is possible to limit the search of the most suitable candidates within their own service networks, e.g. for the composition synthesis process [4, 16].

Secondly, the Goal layer acts as a "semantic service integrator" reconciling peers' service models and dictionaries. Goals are indeed described in a common language and consistency of concepts within the Goal space is ensured by referring to a domain ontology. In this way the inherent heterogeneity of an open P2P system is reconciled within a homogenous space allowing the communication of different services through the Goal layer. In the literature, there are several works dealing with the issue of building semantic service integration systems for composition purposes (e.g. [4, 6, 11]). However, they mainly focus on the translation of service descriptions into an internal form to apply specific techniques for automatic composition. In this paper, rather than focusing on this aspect, we create

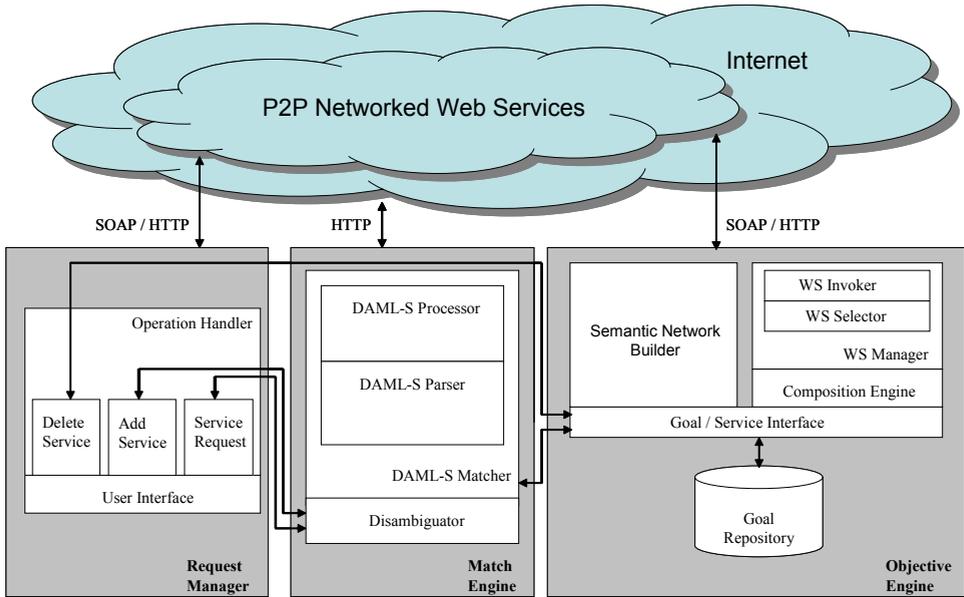


Fig. 2. The FLO²WER Architecture

a proper interoperable environment for such kind of proposals be applicable in our context, regardless of their specificity.

3 The FLO²WER Architecture

A schema of the FLeXible GOal-Oriented Web SeRvice (FLO²WER) architecture we propose is shown in Figure 2. It is a high level description which is meant to show the basic processes and data flows needed to support automatic semantic Web service handling. It is composed of three main modules, the *Request Manager*, the *Match Engine* and the *Objective Engine*, and it is founded on the community ontology [5] described in the Goal Repository.

The user interacts with the system through the Request Manager. Activating the other modules, which are described in the following subsections, it allows to perform three basic operations: Look for, add, or delete a Web service.

3.1 The Goal Repository

The Goal Repository comprises several aspects: a DAML-S description of Goals and a DAML domain ontology which represents the semantics of the information and data of the application domain. It may also include a service ontology specifying the meaning of the offered operations. To this extent, we adopt two semantic integration approaches [5]: A service-oriented approach for Goal descriptions, as they are built taking into account the service available in the underlying P2P

network; a client-tailored approach for the domain ontology which is independent from the services available and that, if needed, can be downloaded off the Web.

Here, DAML-S is used to describe capabilities of a Goal as the Service Profile of a Web service. It describes a Goal as an atomic process thus specifying what the Goal is for, the inputs it requires, the outputs it produces and the pre-conditions that must hold for the Goal to take effect. DAML-S Service Profile also describes the post-conditions, i.e. the service execution effects on the real world. However, for Goal discovery and composition, we are concerned with the knowledge effects (outputs) rather than the physical effects (post-conditions) of executing Web services, which might in turn be relevant for invocation. Inputs, outputs, and pre-conditions refer to the domain ontology and are mapped towards the underlying Goal's subnetwork of services. Thus, we do not maintain mappings from the domain ontology to the peers' schemas, but they can be derived from the mappings between Goals and services.

3.2 The Match Engine

It is invoked by the Request Manager when the user adds or looks for a service and it is composed by the *Disambiguator* and the *DAML-S Matcher*. The Disambiguator takes non-DAML-S user's request and outputs a corresponding DAML-S description disambiguating the information managed against the domain ontology. A disambiguation process is described in [14]: Future research on this approach aims to also disambiguate natural language requests within an accuracy range. The DAML-S Matcher takes a DAML-S user's request to match it with the Goal's descriptions. [10] proposes a possible implementation of the matcher in [9] in an unstructured P2P context. It is composed by the *DAML-S Parser* and the *DAML-S Processor*. The former translates DAML ontologies and DAML-S specifications in a set of predicates, whereas the latter implements a DAML inference engine.

3.3 The Objective Engine

The central component is the *Goal/Service Interface*. Besides mediating between the other components, it allows the automatic creation, activation, deactivation and deletion of a Goal. A Goal is deactivated when its subnetwork is empty, and it is activated again if it matches a new service to be added to its network. There may be several possible policies to adopt for deletion, e.g. timeouts: A deactivated Goal could be deleted after a certain time interval during which it did not match any new added service.

The *Semantic Network Builder* is activated to accomplish a delete or a add service request and drives the integration process necessary to add or delete a service. It maps each Goal to only one service, called the *entry point* of the Goal

subnetwork³. While a structured network is supposed not to be appropriate in the NeP4B context, we can still control how each subnetwork should evolve, optimizing a trade-off between the cost of establishing mappings and the cost of navigating the network in the search of proper services to be chosen. There are several topologies to compare, such as ring, bus, or Cayley graphs networks like hypercubes or star graph [12]. When a new service joins the network, it is matched against existing Goals. If a matching Goal is found, the Semantic Network Builder maps the new service semantically to a service of the subnetwork, namely the *service integrator*, chosen depending on the policies adopted for the network topology. When a service leaves the network, the same process takes place to replace canceled semantic mappings with new ones. Such new semantic mappings are derived by the Goal matching information, along the path of mappings from the entry point to the service integrator, assuming the mapping function to be transitive. At last, when a matching Goal is not found for a new service, or when the last service of a Goal subnetwork leaves the system, the Semantic Network Builder passes this information to the Service/Goal Interface, that provides to create or deactivate the Goal, respectively.

The Composition Engine implements the Goal discovery and composition, identifying Goals suitable to answer a user's request. Once one Goal or a pattern of Goals have been found, the Composition Engine invokes the WS Manager which handles the automatic service composition process. It is constituted by the *Web Service Selector* and the *Web Service Invoker*. The Web Service Selector enters, through the entry point, each of the Goal subnetwork identified, and selects the most suitable service based on different ranking criteria such as reliability, cost, quality of service, trust and reputation and, if available, on its process description⁴. Once Web services have been selected, the Web Service Invoker manages their execution and communication. Successful Goal composition patterns are then stored in the Goal Repository for future use.

4 Conclusions

In this paper, we described a broad framework to achieve a flexible and scalable Web service interoperating environment in an open, dynamic and heterogeneous P2P context, such as it is the NeP4B's. The FLO²WER architecture we propose aims to overcome the heterogeneity of dictionaries and the lack of shared service knowledge due to the autonomy of peers in our network, and allows to create the proper conditions for the application of specific techniques for automatic composition of semantic Web services. In order to do so, we have adopted a hybrid approach which exploits the advantages of centralized registries for service discovery and composition, as well as the dynamism of non-structured P2P

³ There may be also more entry points to one subnetwork to avoid single point of failure. What matters is that the Goal does not have to know the whole set of services which constitute its subnetwork.

⁴ DAML-S Web service description includes a Process Model, where it is defined what is needed for a proper interaction of services

networks that ensure the peers' full autonomy. This is performed by introducing Goals as centralized knowledge of what objectives may be satisfied within the network, which allows to greatly reduce the domain for the composition purposes and to create a semantic service integrator layer.

Due to the peculiarities of the NeP4B project, we believe that to face large scale interoperation of semantic Web services, we first had to address the architectural issues involved. For these reasons, this work might constitute a solid starting point for the NeP4B project as it defines the basic building blocks and execution flows to enable automatic service discovery and composition. In our future works we will focus on the development of the FLO²WER components.

References

1. UDDI: *The UDDI Technical White Paper*. <http://www.uddi.org/>, 2000.
2. B. Van Ark et al. ICT Investments and Growth Accounts for the European Union. Tech. Rep., European Commission, 2002.
3. D. Berardi et al. Automatic Composition of E-Services that Export their Behaviour. In *Proc. of ICSOC*, 2003.
4. D. Berardi et al. Automatic Composition of Transition-based Semantic Web Services with Messaging. In *Proc. of VLDB*, 2005.
5. D. Berardi et al. Automatic Web Service Composition: Service-tailored vs. Client-tailored Approaches. In *Proc. of AISC Workshop (in conj. with ECAI)*, 2006.
6. D. Wu et al. Automating DAML-S Web Services Composition Using SHOP2. In *Proc. of ISWC*, 2003.
7. F. Emekçi et al. A Peer-to-Peer Framework for Web Service Discovery with Ranking. In *Proc. of ICWS*, 2004.
8. M. H. Burstein et al. DAML-S: Web Service Description for the Semantic Web. In *Proc. of ISWC*, 2002.
9. M. Paolucci et al. Semantic Matching of Web Services Capabilities. In *Proc. of ISWC*, 2002.
10. M. Paolucci et al. Using DAML-S for P2P Discovery. In *Proc. of ICWS*, 2003.
11. M. Pistore et al. Automated Synthesis of Composite BPEL4WS Web Services. In *Proc. of ICWS*, 2005.
12. M. Schlosser et al. A Scalable and Ontology-Based P2P Infrastructure for Semantic Web Services. In *Proc. of P2P*, 2002.
13. R. Hamadi and B. Benatallah. A Petri Net-based Model for Web Service Composition. In *Proc. of ADC*, 2003.
14. F. Mandreoli, R. Martoglia, and E. Ronchetti. Versatile Structural Disambiguation for Semantic-aware Applications. In *Proc. of CIKM*, 2005.
15. S. A. McIlraith, T. Cao Son, and H. Zeng. Semantic web services. *IEEE Intelligent Systems*, 16(2):46–53, 2001.
16. S. Narayanan and S. McIlraith. Simulation, Verification and Automated Composition of Web Services. In *Proc. of WWW*, 2002.
17. S. M. P. Papazoglou and D. Georgakopoulos. Service Oriented Computing. *CACM*, 46(10), 2003.
18. C. Schmidt and M. Parashar. A Peer-to-Peer Approach to Web Service Discovery. *WWW*, 7(2), 2004.

Clustering Web and Desktop Searches

Giansalvatore Mecca, Salvatore Raunich, Alessandro Pappalardo

Dipartimento di Matematica e Informatica,
Università della Basilicata, Potenza, Italy

1 Introduction

Web and desktop search services are nowadays essential tools for any Internet user. However, keyword-based, boolean-style search engines like Google usually fall short when asked to answer rather broad queries – those that are often posed by less-experienced users – like, for example, to find documents about the term “*power*” or the term “*amazon*”. The poor quality of results in these cases is mainly due to two different factors: (a) polysemy and/or synonymity in search terms (b) excessively high number of results returned to the user. As a consequence, less skilled users are often frustrated in their research efforts.

The so-called Semantic Web [2] promises to solve most of these problems by adding semantics to Web resources. Although there have been some proposals in the literature towards a semantic Web search engine [6], the transition from the current “syntactic Web” to the next generation Web appears to be a slow process and these proposals can hardly be considered as a ready-made solution to today's search problems.

At the moment, the most promising improvement over traditional search techniques are the so-called *clustering engines*. The idea of clustering search results is not new, and has been investigated quite deeply in Information Retrieval, based on the so called *cluster hypothesis* [13] according to which clustering may be beneficial to users of an information retrieval system since it is likely that results that are relevant to the user are close to each other in the document space, and therefore tend to fall into relatively few clusters.

Several commercial clustering engines have recently emerged on the market. In fact, even Google has experienced for a while with forms of clustering of their search results, and has recently introduced a “Refine Your Query” feature¹ that essentially allows one to select some of a few topics to narrow a search. Similar experiments are also being conducted by Microsoft².

Other well known examples of commercial clustering engines are Vivisimo³ and Grokker⁴. Although the underlying clustering techniques are not fully disclosed to the public, based on the available documentation it is possible to say that these systems share a number of common features with

1 <http://www.google.com/help/features.html#refine>

2 <http://rwsn.directtaps.net/>

3 <http://www.vivisimo.com>

4 <http://www.grokker.com>

research systems introduced in literature, mainly the Grouper system [15]. We summarize these features in the following.

First, these tools are usually not search engines by themselves. On the contrary, when a user poses a query, the clustering engine uses one or more traditional search engines to gather a number of results; then, it does a form of post-processing on these results in order to cluster them into meaningful groups. The cluster tree is then presented to the user so that s/he can browse it in order to explore the result set. It can be seen that such a technique may be helpful to users, since they can quickly grasp the different meanings and articulations of the search terms, and more easily select a subset of relevant clusters.

Being based on a post-processing step, all of these clustering engines work by analyzing *snippets*, i.e., short document abstracts returned by the search engine, usually containing words around query term occurrences. The reason for this is performance: each snippet contains from 0 to 40 words, and therefore can be analyzed very quickly, so that users do not experience excessive delays due to the clustering step. However, snippets are often hardly representative of the whole document content, and this may in some cases seriously worsen the quality of the clusters.

Noodles [9] aims at reconsidering some of the techniques for clustering search results. More specifically, we investigate the trade-off between performance and quality of the clustering when choosing snippets versus whole documents. This is particularly relevant if we consider that in some emerging contexts, like for example, desktop search engines, (a) it is reasonable to assume that the document-term vectors are available to the clustering engine; (b) snippets may not be available at all, based on the different nature of documents.

The main goal of the system is to evaluate the quality of the clustering in terms of its ability to correctly *classify* documents, i.e., to dynamically build a bunch of clusters that correctly reflect the different categories in the document collection returned by the search engine. Similarly to [3], we believe that this ability may assist less-skilled users in browsing the document set and finding relevant results.

The system is based on a new document clustering algorithm called *Dynamic SVD Clustering (DSC)*, based on Latent Semantic Indexing [4]; the novelty of the algorithm is twofold: (a) first, it is based on an incremental computation of singular values, and does not require to compute the whole SVD of the original matrix; (b) second, it uses an original strategy to select k , i.e., the number of singular values used to represent the “concepts” in the document space; differently from other proposals in the literature, like for example [12] or [10], our strategy does not assume a fixed value of k , neither a fixed approximation threshold. Experimental results [9] show that the algorithm has very good classification power; in many cases it is able to cluster pre-classified documents collections with 100% accuracy; it is worth noting that the quality of the classification severely degrades when snippets are used in place of the whole document content, thus providing further evidence that snippets are often too poor and not sufficiently informative.

In [9] we show that the algorithm has good performance, since it has low computational complexity and lends to a very natural clustering strategy based on the minimum spanning tree of the projected document space.

To the best of our knowledge, this is the first research to propose a dynamic strategy to discover the optimal number of singular values to be used in a classification task. This strategy represents the main contribution of this work.

2 Preliminaries: Latent Semantic Indexing

Latent Semantic Indexing (LSI) [4] is a document projection technique based on *Singular Value Decomposition (SVD)*. Suppose we are given d documents and t terms; let us represent each document as a term vector of t coordinates. We call A the $t \times d$ matrix whose columns are the term vectors; let r be the rank of A . SVD decomposes matrix A into the product of three new matrices, as follows:

$$A = U \Sigma V^T = \sum_{i=0}^r \sigma_i u_i v_i^T$$

where:

- Σ is a diagonal matrix of size $r \times r$ made of the *singular values* of A in decreasing order: $\Sigma = \text{diag}(\sigma_1, \sigma_2, \dots, \sigma_r)$, $\sigma_1 \geq \sigma_2 \geq \dots \geq \sigma_r > 0$
- U and V are of size $t \times r$ and $d \times r$ respectively; vectors u_i are called the *left singular vectors* and v_i^T are the *right singular vectors* of A

A singular value and its left and right singular vectors are also called a *singular triplet*. In order to obtain an approximation of A let us fix $k \leq r$ and call $\Sigma_k = \text{diag}(\sigma_1, \sigma_2, \dots, \sigma_k)$, i.e., the $k \times k$ head minor of Σ . Similarly, let us call U_k the restriction of U to the first k left singular vectors; similarly for V_k^T ; we define:

$$A_k = U_k \Sigma_k V_k^T$$

It is possible to prove [4] that A_k is the best k -rank approximation of the original matrix A . Informally speaking, by appropriately choosing a value for k we are selecting the largest singular values of the original matrix, and ignoring the smallest ones, therefore somehow preserving the main features of the original vector space while filtering out some “noise”. It can be seen that the smaller is k with respect to r , the less the new space resembles the original one. In traditional information retrieval applications, in which LSI is used as a means to improve retrieval performance, it is crucial that the essential topological properties of the vector space are preserved; as a consequence, the value of k is usually quite high (empirical studies [12] show that for a typical information retrieval application a value between 150 and 300 is usually the best choice).

3 Overview of the Clustering Algorithm

Our algorithm is heavily based on Latent Semantic Indexing. LSI has a natural application in clustering and classification applications. It is often said that, by means of SVD, LSI does perform a transformation of the original vector space – in which each coordinate is a term – into a new vector space, in which each coordinate is some relevant “concept” in the document collection, i.e., some topic that is common to several documents. Note that the coordinates of the original documents in this space are given by matrix $V_k \Sigma_k$. We shall call this $d \times k$ space the *projected document space*.

In this respect, a possible approach to clustering would be the following: (a) compute SVD over the original matrix, for some value k , to obtain a representation of the original documents in the new “concept” space, $V_k \Sigma_k$, in which each coordinate represents some “topic” in the original document collection, and therefore some cluster of documents; (b) run a clustering algorithm in this space to cluster documents with respect to their topics. This method has been used for clustering purposes for example in [10].

A critical step, in this approach, is the selection of k . A natural intuition suggests that, assuming the document collection contains x hidden clusters, the natural value of k to be used for SVD is exactly x . This is a consequence of the fact that one of the property of SVD is that of producing in $V_k \Sigma_k$ an optimal alignment of the original documents along the k axes [11]. However, such a value is unknown and must be discovered by the clustering engine. There are a few interesting observations with respect to this point:

- first, such a value of k can be significantly lower than the number of documents, d , and the number of terms, t , since it is unlikely that there are more than a dozen relevant clusters among the results of a search; this means that the projected document space does not preserve much of the features of the original space; this is, however, not a concern, since in our case we are using this space only as a means to discover clusters, and not for retrieval purposes;
- it is therefore apparent that fixing the value of k , for example assuming k in the order of 150-300, as it is often done in the literature, would not give good classification performance; the fact that lower values of k are to be preferred in clustering applications is confirmed for example in [12]; nevertheless, in the latter work fixed values of $k=20$ and $k=50$ are compared, whereas the optimal value should ideally be discovered dynamically;
- in light of these considerations, we also discard the other typical approach used to establish the value of k , i.e., that of fixing a threshold for the approximation that A_k gives of the original space A , as it is done, for example, in [10].

In fact, the strategy used to dynamically select the optimal value of k is one of the main originality of our algorithm.

The main intuition behind the selection of k is that – assuming the original

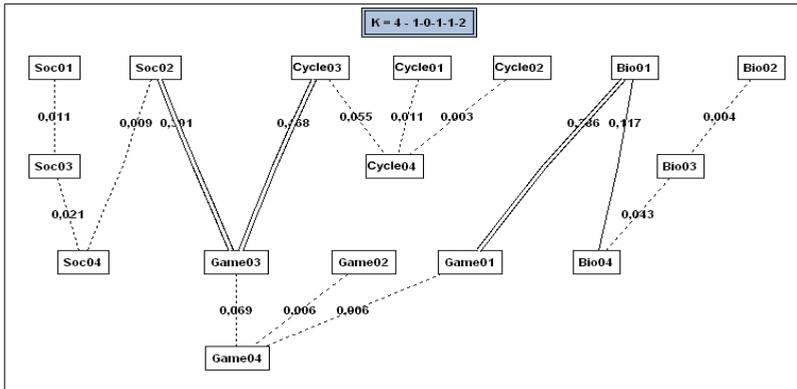


Figure 1: Example of a Minimum Spanning Tree for the Life Example

collection contains x clusters, and that each cluster informally corresponds to some clearly defined “topic” – then the points in the projected document space should naturally fall into x clusters that are reasonably compact and well separated.

To see this, consider for example Figure 1, which refers to one of our test document collection, corresponding to search results for the query term “life”. We have identified four natural clusters in the collection, namely: (a) documents about biological life; (b) documents about life cycles of a system or a living entity; (c) documents about social life and related events; (d) documents about the game of life. The figure shows the minimum spanning tree of document distances in space $V_4 \Sigma_4$. Edges are drawn differently based on their length; more specifically, the longest ones are drawn as a double line, the shortest ones as a dashed line. It can be seen that, on the one side SVD has clearly brought documents belonging to the same cluster close to each other, and on the other side that an accurate clustering can be obtained simply by removing the $k-1$ longest edges, that is, by applying a variant of a spanning-tree based clustering algorithm [14] [8].

Based on this observation, our algorithm can be informally sketched as follows:

- given the original term-document space A we incrementally compute SVD, starting with a low number of singular values, and, for each value of k , generate the projected space $V_k \Sigma_k$;
- we find the minimum spanning tree of points in the projected space; assuming we have found the optimal value for k , we stop the algorithm and obtain our clusters simply by removing the $k-1$ longest edges from the minimum spanning tree, and considering each connected component as a cluster;
- to discover such optimal value of k , we define a quality function for the

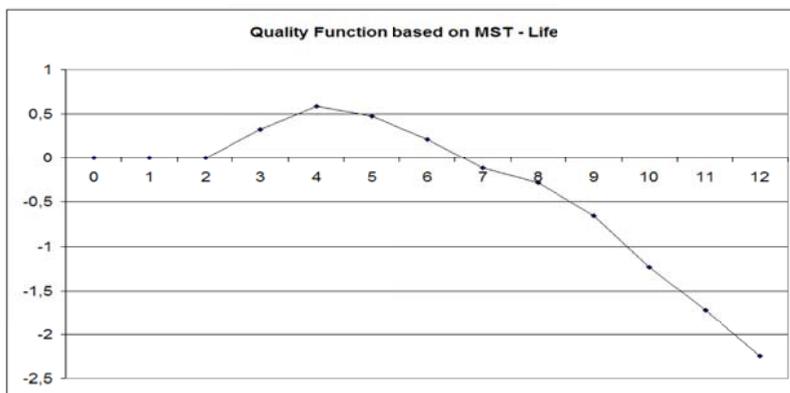


Figure 2: *Quality Function for the Life Example*

clusters obtained from the minimum spanning tree; intuitively, this function rates higher those trees in which the $k - 1$ longest edges are significantly longer than the average; based on this, we stop as soon as we find a local maximum of the quality function; this is a hint that we have found a “natural” distribution of documents in the projected space, that would be worsened if we choose higher values of k , i.e., more clusters.

A plot of the quality function for the life example discussed above is reported in Figure 2. It can be seen that the quality function has a maximum for $k=4$, that is exactly the number of clusters we are seeking in this example. It is also worth noting that such a strategy nicely solves a typical problem of clustering algorithms, i.e., that of dynamically choosing the right number of clusters for a given dataset [8].

A detailed description of the algorithm is given in the full paper [9].

4 Implementation and Experiments

The clustering algorithm has been implemented in the Noodles desktop search engine. The system is written in Java, using Apache Lucene⁵ as an indexing engine, and the Spring Rich Client Platform⁶ as a desktop application framework. It has been conceived as a general-purpose search tool, that can run both Web and desktop searches. In order to perform desktop searches, it incorporates a crawler module that runs as a daemon and incrementally indexes portions of the local disks that have been specified by the user.

It also incorporates a testbed for the clustering engine, which we have used to conduct a number of experiments. Overall, we have run several hundred experiments, combining different datasets, different summarization schemes, different weighting schemes, different criteria for the selection of k in SVD,

⁵ <http://lucene.apache.org>

⁶ <http://www.springframework.org>

and different clustering algorithms. The most interesting experimental evidences are described in [9]. All experimental data – including document collections and logs of the experiments – are available on the project Web site⁷.

5 Related Works

The idea of clustering search results as a means to improve retrieval performance has been investigated quite deeply in Information Retrieval. A seminal work in this respect is the Scatter/Gather project [7]. After the user has posed her/his query, s/he can decide to “scatter” the results into a fixed number of clusters; then, s/he can “gather” the most promising clusters, possibly to scatter them again in order to further refine the search. One limitation in Scatter-Gather is the fact that the system is not able to infer the optimal number of clusters for a given query, and requires that this is specified by the user in advance.

The Paraphrase system [1] introduces Latent Semantic Indexing as a means to cluster search results. The authors follow a rather typical approach with respect to the selection of the number k of singular values. More specifically, k is fixed and equals 200; in fact, values in the range 100-200 were considered as optimal in retrieval applications, competitive or even superior to term-based similarity search.

In [12] the authors further explore the use of projection techniques for document clustering. Their results show that LSI performs better than document truncation. Also in this case, the number k of singular values is fixed. The authors compare the quality of the clustering with $k=20$, $k=50$ and $k=150$. An interesting point is that, although $k=150$ was considered a more typical value, the paper concludes that in clustering applications $k=20$ gives better performance and clustering quality. This conclusion is coherent with our idea that the optimal value of k for clustering documents is equal to the number of classes (or ideal clusters), and therefore usually much lower than the number of documents.

Web clustering engines like Grouper [15] have been discussed in Section 1. Grouper has inspired a number of other proposals along the same lines. Examples are SHOC [16] and Lingo/Carrot Search [10]. Both extends the STC algorithm with the use of SVD in order to filter some “noise” in the snippets and improve the quality of the produced clusters. SHOC's clustering is based on two steps: during the first step phrase analysis is used to generate a snippet-topic matrix – in which a topic is either a term or a phrase; then, as a second step, SVD is performed on the matrix in order to identify the most relevant topics. A key difference with our work is that the stop criterion for SVD is based on a fixed approximation threshold. No experimental results are reported in [16] to assess the quality of the clustering algorithm.

Lingo/Carrot Search is similar in spirit, but it uses a different strategy. A primary concern is to produce meaningful descriptions for the clusters. To do this, first SVD is performed on a snippet-term matrix to identify a number of relevant topics. Also in this case, the selection of k is based on a fixed

⁷ <http://www.db.unibas.it/projects/noodles>

approximation threshold specified by the user. Then, phrase analysis is done to identify, for each of the selected topics, a phrase that represents a good description. Finally, documents are assigned to clusters based on the contained phrases.

SnakeT [5] introduces advanced cluster labeling with variable-length sentences, based on the use of “gapped sentences”, i.e., sentences made of terms that may not appear contiguously in the original snippet. The main focus of the system is on personalization: the authors show how to plug the clustering engine into a search engine to obtain a form of personalization.

Bibliography

1. P. Anick, S. Vaithyanathan. Exploiting Clustering and Phrases for Context-Based Information Retrieval. In *ACM SIGIR*, 1997. 314 - 323
2. T. Berners-Lee, J. Hendler, O. Lassila. The Semantic Web. *Scientific American* (2001) 284(5):34-43.
3. C. Chekuri, P. Raghavan. Web Search Using Automatic Classification. In *Proceedings of the World Wide Web Conference*, 1997.
4. S. Deerwester, S. T. Dumais, G. W. Furnas, T. K. Landauer, R. Harshman. Indexing by Latent Semantic Analysis. *Journal of the American Society for Information Sciences* (1990) 41(6):391-407.
5. P. Ferragina, A. Gulli. A Personalized Search Engine Based on Web Snippet Hierarchical Clustering. In *Proceedings of the World Wide Web Conference*, 2005.
6. R. Guha, R. Mc Cool, E. Miller. Semantic Search. In *Proceedings of the World Wide Web Conference*, 2003.
7. M. A. Hearst, J. O. Pedersen. Re-examining the Cluster Hypothesis: Scatter/Gather on Retrieval Results. In *Proceedings of the ACM SIGIR Conference*, 1996.
8. A. K. Jain, M. N. Murty, P. J. Flynn. Data Clustering: A Review. *ACM Computing Surveys* (1999) 31(3):265-323.
9. G. Mecca, S. Raunich, A. Pappalardo. A New Algorithm to Cluster Search Results. *Data and Knowledge Engineering* (2007) To appear:- Available as Noodles WR-01-2006 at <http://www.db.unibas.it/projects/noodles>.
10. S. Osinski, J. Stefanowski, D. Weiss. Lingo: Search Results Clustering Algorithm Based on Singular Value Decomposition. In *Proceedings of the International Conference on Intelligent Information Systems (IIPWM)*, 2004.
11. C. Papadimitriou, P. Raghavan, H. Tamaki, S. Vempala. Latent Semantic Indexing: a Probabilistic Analysis. *Journal of Computer and System Sciences* (2000) 61:217 - 235 .
12. H. Schutze, C. Silverstein. Projections for Efficient Document Clustering. In *Proceedings of ACM SIGIR*, 1997.
13. C. J. van Rijsbergen. *Information Retrieval, Second Edition*. London, Butterworths, 1979.
14. C. T. Zahn. Graph-Theoretical Methods for Detecting and Describing Gestalt Clusters. *IEEE Transactions on Computers* (1971) C-20:68-86.
15. O. Zamir, O. Etzioni. Web Document Clustering: A Feasibility Demonstration. In *Proceedings of the ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR)*, 1998.
16. D. Zhang, Y. Dong. Semantic, Hierarchical, Online Clustering of Web Search Results. In *Proceedings of the Asia-Pacific Web Conference*, 2004.

Online Distribution Estimation for Streaming Data: Framework and Applications

Themis Palpanas¹, Vana Kalogeraki², and Dimitrios Gunopulos²

¹ University of Trento

themis@dit.unitn.it

² University of California, Riverside

{vana, dg}@cs.ucr.edu

Abstract. In the last few years, we have been witnessing an evergrowing need for continuous observation and monitoring applications. This need is driven by recent technological advances that have made streaming applications possible, and by the fact that analysts in various domains have realized the value that such applications can provide.

In this paper, we propose a general framework for computing efficiently an approximation of multi-dimensional distributions of streaming data. This framework enables the development of a wide variety of complex streaming applications. In addition, we demonstrate how our framework can operate in a distributed fashion, thus, making better use of the available resources.

We motivate our techniques using two concrete problems, both in the challenging context of resource-constrained sensor networks. The first problem is outlier detection, while the second is detection and tracking of homogeneous regions. Experiments with synthetic and real data show that our method is efficient and accurate, and compares favorably to other proposed techniques for both the problems that we studied.

1 Introduction

Advances in processor technologies and wireless communications have enabled the deployment of small, low cost and power efficient sensor nodes. Such sensor deployments enable the observation of physical phenomena at a time and space granularity that was never before possible. The same is also true for monitoring the operation of machinery, and the structural integrity of buildings and vehicles.

In all these cases, the applications deal with several data streams and require the ability to efficiently process this type of data in real-time. Applications in several other domains have the same requirements as well. For example, in an e-business scenario, we are interested in continuously monitoring the execution of the various processes and the corresponding services, and in network management, we have to continuously analyze traffic statistics coming from multiple sources.

The way that streaming applications are able to efficiently process continuous data arriving at high rates is by computing succinct summaries of the data, and operating on these summaries [8, 7]. In this paper, we describe a framework for efficient, online approximation of multi-dimensional streaming data distributions, based on *kernel density*

estimators [18]. The proposed framework does not require a priori knowledge about the input distribution, and is adaptive, in the sense that it automatically recognizes changes in the streaming data distributions, and updates the approximations accordingly. Moreover, the framework can operate in a distributed fashion, thus, making use of all the available resources, and reusing any processing that has already taken place.

The framework we propose is general, and enables the development of a wide variety of complex streaming applications. In this study, we demonstrate the usefulness and versatility of the framework using two concrete applications from the area of sensor networks. Sensor networks represent a challenging domain, because they combine the requirements of streaming algorithms (i.e., online processing, and efficient use of main memory) with the restrictions of sensor network deployments (i.e., limited available resources and processing power, and distributed operation).

The first application that we examine is distributed deviation detection in a sensor network. The goal is to identify, among all the sensor readings in a sliding window, those values that have very few near neighbors. Note that this is a challenging problem, even for static datasets [14, 17]. This problem is especially important in the sensor network setting because it can be used to identify faulty sensors, and to filter spurious reports from different sensors.

The second application is identification and tracking of *homogeneous regions* [1, 12], which are defined as spatial divisions of the field under observation that exhibit similar measured values over time. For example, an oil spill detected in the ocean is a homogeneous region (Figure 1). The sensors deployed around the origin of the spill can organize themselves into a network and communicate the measurements, to detect regions of varying oil concentrations. In this work, we address the problems of detecting and tracking such homogeneous regions in *real-time* when the definition of the phenomenon is *not* known in advance.

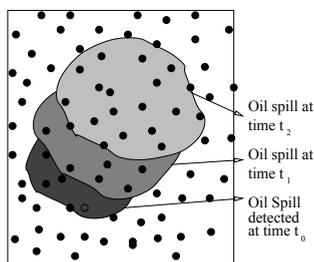


Fig. 1. Spread of an oil spill detected in the ocean over time.

Below we summarize our main contributions.

- We describe a non-parametric, online technique for approximating unknown multi-dimensional streaming data distributions. We then extend this technique in order to efficiently model distributions that evolve over time, and to operate in a distributed fashion.
- We demonstrate the versatility of our approach by describing how it facilitates the implementation of two different applications, namely outlier detection, and detection and tracking of homogeneous regions.

2 Data Distribution Approximation Framework

In this section, we describe a general framework for estimating the underlying distribution of a data stream. We discuss the case where we are interested in the data that falls within a sliding time window W (this is a more general case than unrestricted windows; we omit the presentation of the latter for brevity).

Estimating the Probability Density Function: There are several model estimation techniques that have been proposed in the literature, such as histograms [10], wavelets [8], kernel density estimators [18], and others. In our framework, we choose to estimate the unknown distribution using the kernel density estimators, because of the following desirable properties: (i) they are efficient to compute and maintain in a streaming environment; (ii) they can very accurately approximate an unknown data distribution, with no a priori knowledge and (effectively) no parameters; (iii) they can easily be combined and (iv) they scale well in multiple dimensions. In general, it is computationally more expensive to apply the above operations in histograms or wavelets, and their performance is not better than that of kernels [11].

Kernel Estimators: The *kernel estimator* [18] is a generalized form of sampling, whose basic step is to produce a uniform random sample. In kernel estimation each point distributes its weight in the space around it. A *kernel function* describes the form of this weight distribution, generally distributing most of the weight in the area near the point. Summing up all the kernel functions we obtain a density function for the dataset.

More formally, assume that we have a static relation, T , that stores the d -dimensional values \mathbf{t} , $\mathbf{t} = (t_1, \dots, t_d)$, whose distribution we want to approximate. The recorded values must fall in the interval $[0, 1]^d$. This requirement is not restrictive, since we can map the domain of the input values to the interval $[0, 1]^d$. Let R be a random sample of T , and $k(\mathbf{x})$ a d -dimensional function of $\mathbf{x} = (x_1, \dots, x_d)$, such that $\int_{[0,1]^d} k(\mathbf{x}) d\mathbf{x} = 1$, for all tuples in R . We call $k(\mathbf{x})$ the *kernel function*. We can now approximate the underlying distribution $f(\mathbf{x})$, according to which the values in T were generated, using the following function

$$f(\mathbf{x}) = \frac{1}{|T|} \sum_{\mathbf{t}_i \in R} k(x_1 - t_{i_1}, \dots, x_d - t_{i_d}). \quad (1)$$

The choice of the kernel function is not significant for the results of the approximation [18]. Hence, we choose the Epanechnikov kernel that is easy to integrate:

$$k(\mathbf{x}) = \begin{cases} \left(\frac{3}{4}\right)^d \frac{1}{B_1 \dots B_d} \prod_{1 \leq i \leq d} \left(1 - \left(\frac{x_i}{B_i}\right)^2\right) \\ \quad , \text{ if } \forall i, 1 \leq i \leq d, \left|\frac{x_i}{B_i}\right| < 1 \end{cases} \quad (2)$$

where $\mathbf{B} = (B_1, \dots, B_d)$ is the bandwidth of the kernel function. We use Scott's rule to set \mathbf{B} [18]: $B_i = \sqrt{5} \sigma_i |R|^{-\frac{1}{d+4}}$, where σ_i is the standard deviation of the values in T in dimension i .

Online approximation in a sliding window: For the discussion that follows, and for ease of presentation, we assume the case of a sensor network model (though, our framework is applicable to any environment with multiple streams of data), consisting of a set of sensors, each having a location on a 2-d plane and producing a stream of values. The

sensor network may be organized in a hierarchical way for scalability reasons. The idea is to organize the network using several tiers of overlapping virtual grids with different levels of granularity, ranging from small local areas at the lowest tier, to the entire network area at the highest tier (see Figure 2). We assume that each cell of the grid elects a leader, or *parent* node, that is responsible for processing the measurements of all the sensors in the cell, and for collecting values from the leader nodes of all its sub-cells in the lower level. In such an online setting, we require that each sensor maintains a model

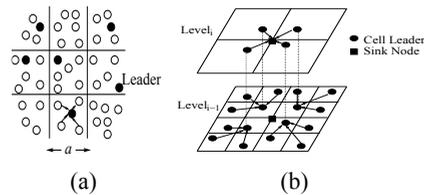


Fig. 2. (a) Sensor field organization (b) *Leader* node

for the distribution of values it generates within a sliding window W of size N (see Figure 3). The first step in creating this model is to maintain online a random sample of size $|R|$ of the set of the values in the most recent window W . The other quantity we need for the kernel estimator is the standard deviation σ of the values in the sliding window W . Both of these operations can be efficiently supported in a data streaming environment [2, 4, 13, 3].

Theorem 1. *The memory requirement for a sensor to maintain an estimate of its distribution is $O(|R| + \frac{1}{\epsilon^2} \log|W|)$, where $|R|$ is the size of the sample, ϵ is the maximum error for the estimation of the standard deviation, and $|W|$ is the size of the sliding window.*

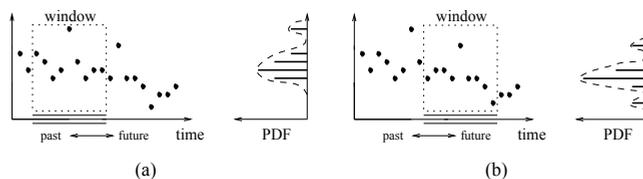


Fig. 3. Estimation of data distribution in sliding window for two time instances (1-d data).

Distributed Computation of Estimators In our framework, we define a mechanism for model composition. This allows us to take the data distribution models of two streams, and construct a single model that describes the behavior of both data streams. Our kernel estimators can be easily combined, and thus are well suited for our framework. There are two quantities that we have to combine, the sample set, R , and the bandwidth of the kernel function, B . We can combine the sample sets just by taking their union. We may then reduce the size of the resulting set by re-sampling, if necessary. In order to

combine the bandwidths of two kernel functions, we only need to combine the two standard deviations upon which the bandwidths depend. This is accomplished using the same techniques as the ones for computing the standard deviation in a sliding window of streaming data [20].

Propagating Estimator Updates in the Network Hierarchy: An interesting question is how often a node should send its model to the leader of the cell it belongs to (assuming a hierarchical organization).

The simplest approach is to have the children transmit updates to the parent as these updates take place in their own estimators. Assume that the parent has l children, each having a kernel estimator of size $|R|$, and that the kernel estimator of the parent has size $|R_p|$. Then, with probability $f = \frac{|R_p|}{l|R|}$, when a child updates its kernel estimator by adding a new kernel, it also propagates this update to its parent (i.e., it transmits the new kernel and the new standard deviation(s)). This simple approach can be used to efficiently maintain a sample with expected size $|R_p|$ at the parent, and has the important advantage that the parent's distribution quickly adjusts to changes in the distribution of the observed data.

Comparing Distributions We now discuss a method for computing the difference between two distributions, which will be useful for the algorithms we describe. Several methods have been proposed to quantify the difference between probability density distributions [15]. One widely used measure is the *Kullback-Liebler divergence* $D(p \parallel q)$ [6], which is defined as

$$D(p \parallel q) = \int_{\mathbf{y}} p(\mathbf{y})(\log p(\mathbf{y}) - \log q(\mathbf{y})), \quad (3)$$

where $p(\mathbf{y})$ and $q(\mathbf{y})$ are probability distribution functions over \mathbf{y} , and \mathbf{y} is drawn from a finite set \mathbf{Y} . However, the measure is undefined when $p(\mathbf{y}) > 0$ but $q(\mathbf{y}) = 0$ for some $\mathbf{y} \in \mathbf{Y}$. The *KL-divergence* is therefore not applicable to the density distributions derived by kernel density estimation method, because this method may assign probability of zero for regions in the domain of the values. We use a variation of the KL-divergence, called the *Jensen-Shannon divergence* [16] which is defined as follows

$$JS(p, q) = \frac{1}{2} [D(p \parallel avg(p, q)) + D(q \parallel avg(p, q))] \quad (4)$$

where $avg(p, q)$ is the average distribution $(p(\mathbf{y}) + q(\mathbf{y}))/2$.

We estimate the JS-distance between two kernel estimator models $p(\mathbf{x})$ and $q(\mathbf{x})$ as follows. We approximate the estimated distribution with the values of the function with a finite set of grid points $\mathbf{b}_1, \mathbf{b}_2, \dots, \mathbf{b}_k$. Thus, we approximate the term $D(p \parallel avg(p, q))$ in Equation 4 as

$$D(p \parallel avg(p, q)) = \sum_{i=1 \dots k} P_p(\mathbf{b}_i, bs/2) \times \left[\log(P_p(\mathbf{b}_i, bs/2)) - \log\left(\frac{P_p(\mathbf{b}_i, bs/2) + P_q(\mathbf{b}_i, bs/2)}{2}\right) \right] \quad (5)$$

where bs is the grid interval and P_p and P_q are the probabilities estimated with respect to the estimator models $p(\mathbf{x})$ and $q(\mathbf{x})$ respectively. We approximate the term $D(q \parallel avg(p, q))$ in Equation 4 in a similar way, and estimate the JS-divergence between the kernel estimator models. The time complexity for the above procedure is $O(dk|R|)$.

3 Applications of Framework

3.1 Outlier Detection

There exist several formal definitions of an outlier. In our work, we follow two of the commonly-used definitions.

Distance-based outliers [14]: A point \mathbf{p} in a dataset T is a (D, r) -outlier if *at most* D of the points in T lie within distance r from \mathbf{p} . The approach to detect such outliers does not require any prior knowledge of the underlying data distribution, but rather uses the intuitive explanation that an outlier is an observation that is sufficiently far from most other observations in the dataset.

Local metrics-based outliers [17]: This method detects outliers based on the metric Multi Granularity Deviation Factor (MDEF). For any given value \mathbf{p} , MDEF is a measure of how the neighborhood count of \mathbf{p} (in its *counting* neighborhood) compares with that of the values in its *sampling* neighborhood. A value is flagged as outlier, if its MDEF is (statistically) significantly different from that of the local averages. The parameters r , the *sampling* neighborhood and αr , the *counting* neighborhood, determine the range over which the neighborhood counts are estimated. This method takes into consideration the local density variations in the feature space, and provides an automatic cut-off for the outliers, based on the characteristics of the data.

We make two observations. The first observation is that both definitions of outliers require the estimation of the number of points within specified regions of the data space. As we discussed in Section 2, our framework for estimating the probability density function of streaming data can efficiently provide the answers to such questions.

The second observation (summarized by the following theorem) is specific to the density-based outliers, and leads to an even more efficient implementation.

Theorem 2. *Assume nodes n_1, \dots, n_l children of node n_p , data streams S_1, \dots, S_l referring to the l children nodes, and corresponding sliding windows W_1, \dots, W_l . The sliding window of node n_p is defined as $W_p = \bigcup_{i=1}^l W_i$. Let, at some point in time, O_1, \dots, O_l be the sets of distance-based outliers corresponding to the l sliding windows. Then, for the set O_p of outliers in W_p it holds that $O_p \subseteq \bigcup_{i=1}^l O_i$.*

This theorem is important for two reasons: (a) it results in significant computation savings for the parent node, because it only needs to examine a very small subset of the streaming values, i.e., the outliers identified by its children; (b) it limits the necessary communication messages from the children nodes to their parents.

The experimental results show that our approach can result in approximately 95% precision and recall for identifying outliers when compared to the offline, centralized approach, while at the same time being orders of magnitude more efficient [20].

3.2 Detection and Tracking of Homogeneous Regions

The problems we are considering in this application are the following.

Problem 1. (Homogeneous Region Discovery) Given L sensors monitoring an area of interest, find a group of sensors (corresponding to a region in space) such that the observations of the sensors belonging to the same group are *similar*, and are different from those of other sensors.

Problem 2. (Homogeneous Region Tracking) For a given region R defined by a set of *similar* sensors, track its movement over time.

In order to solve these problems, we need to address the following issues. First, to detect homogeneous regions, we need an efficient technique to identify sensors with similar readings. Second, we need to provide a formulation that allows us to define and discover homogeneous regions that differ from the surrounding area.

Once again, the basis for our solution is the proposed framework that allows the efficient estimation of data densities in a distributed manner. In our solution we first estimate the data distributions within each cell, and then cluster together cells with similar distributions. During this process, we define the homogeneous regions, and approximate their boundaries, which we subsequently track as they evolve over time.

The experimental evaluation shows that we can effectively detect and track homogeneous regions, with very small processing and communication costs [19].

3.3 Other Applications

An accurate online approximation of the probability density function allows us to solve a number of problems in a sensor network.

Online Query Processing: One category of problems is to provide approximate answers to range queries with both spatial and temporal constraints. These are queries of the following form: “What is the average temperature in region (X, Y) during the time interval $[t_1, t_2]$?”. In such cases, the sensors can estimate the density model for the observations during the specified time interval and answer the queries based on the estimated model.

Finding Faulty Sensors: Another important application is online detection of faulty sensors. Examples include queries of the form: “Give a warning when the values of a given sensor are significantly different from the values of its neighbors over the most recent time window W ”, or queries of the form: “Give a warning if the number of outliers in a given region exceeds a given threshold T over the most recent time window W ”. With our approach, a parent sensor can compute the difference between the estimator models received from its children, to determine if any of them is faulty.

4 Related Work

A recent study [7] proposes a sensor data acquisition technique, based on models that approximate the data with probabilistic confidences. However, any special characteristics of the data distribution, such as periodic drifts, have to be explicitly encoded in the space of models considered, which requires domain knowledge. In our work, we describe a more general technique, which can efficiently overcome this limitation.

A framework for modeling sensor network data is proposed by Guestrin et al. [9]. The goal is that the nodes in the network collaborate in order to fit a global function to each of their local measurements. Being a parametric approximation technique, it has more parameters to fit than our approach, where we only have to estimate a single parameter. A new study proposes a methodology for approximate data collection that

also exploits spatial correlations [5]. This approach significantly reduces the communication costs, but it is not clear how it can support distributed, in-network processing, or dynamically adapt to changes in the spatial correlations.

Ali et al. [1] propose an interesting approach to detect and track discrete phenomena (PDT) in sensor networks. Detecting phenomena with PDT is a centralized approach, and therefore, has high communication energy cost. In our work, we consider a more general problem and we employ a distributed approach. Hellerstein et al. [12] propose algorithms to partition the sensors into *isobars*, i.e., groups of neighboring sensors with approximately equal values during an epoch. In our case, we are partitioning the sensors according to the summary of their values over a time interval that spans several epochs, and make no a priori decisions as to how to group sensors based on their value ranges.

5 Conclusions

In this paper, we propose a general and flexible framework for approximating the distribution of streaming data. The techniques we describe operate efficiently in an online fashion. Moreover, they distribute the computation effort among the nodes available in the network, thus better exploiting the available resources and cutting back on the processing and communication costs. We evaluated our approaches by applying them to different problems, which demonstrates the versatility of the proposed approach.

References

1. M. H. Ali, M. F. Mokbel, W. G. Aref, and I. Kamel. Detection and tracking of discrete phenomena in sensor-network databases. In *SSDBM*, pages 163–172, Santa Barbara, CA, 2005.
2. B. Babcock, M. Datar, and R. Motwani. Sampling From a Moving Window Over Streaming Data. In *SODA*, 2002.
3. B. Babcock, M. Datar, R. Motwani, and L. O’Callaghan. Maintaining Variance And k-medians Over Data Stream Windows. In *PODS*, pages 234–243, San Diego, CA, USA, 2003.
4. B. Bash, J. Byers, and J. Considine. Approximately uniform random sampling in sensor networks. *DMSN*, 2004.
5. D. Chu, A. Deshpande, J. Hellerstein, and W. Hong. Approximate data collection in sensor networks using probabilistic models. *ICDE*, 2006.
6. T. M. Cover and J. A. Thomas. *Elements of Information Theory*. John Wiley & sons, 1991.
7. A. Deshpande, C. Guestrin, S. R. Madden, J. M. Hellerstein, and W. Hong. Model-Driven Data Acquisition in Sensor Networks. In *VLDB*, Toronto, ON, Canada, 2004.
8. A. C. Gilbert, Y. Kotidis, S. Muthukrishnan, and M. Strauss. Surfing Wavelets on Streams: One-Pass Summaries for Approximate Aggregate Queries. In *VLDB*, Rome, Italy, 2001.
9. C. Guestrin, P. Bodik, R. Thibaux, M. Paskin, and S. Madden. Distributed Regression: an Efficient Framework for Modeling Sensor Network Data. In *IPSN*, Berkeley, CA, 2004.
10. S. Guha and N. Koudas. Approximating a Data Stream for Querying and Estimation: Algorithms and Performance Evaluation. In *ICDE*, pages 567–576, San Jose, CA, 2002.
11. D. Gunopulos, G. Kollios, V. J. Tsotras, and C. Domeniconi. Approximating Multi-Dimensional Aggregate Range Queries over Real Attributes. In *SIGMOD*, Dallas, TX, USA, 2000.

12. J. M. Hellerstein, W. Hong, S. Madden, and K. Stanek. Beyond average: Toward sophisticated sensing with queries. In *IPSN*, pages 63–79, 2003.
13. A. Jain and E. Chang. Adaptive sampling for sensor networks. *DMSN*, 2004.
14. E. M. Knorr and R. T. Ng. Algorithms for Mining Distance-Based Outliers in Large Datasets. In *VLDB*, NY, NY, 1998.
15. L. Lee. On the effectiveness of the skew divergence for statistical language analysis. In *Artificial Intelligence and Statistics 2001*, pages 65–72, 2001.
16. J. Lin. Divergence measures based on the shannon entropy. *IEEE Trans. Infor. Theory*, 37:145–151, 1991.
17. S. Papadimitriou, H. Kitagawa, P. Gibbons, and C. Faloutsos. Loci: Fast outlier detection using the local correlation integral, 2003.
18. D. Scott. *Multivariate Density Estimation: Theory, Practice and Visualization*. Wiley & Sons, 1992.
19. S. Subramaniam, V. Kalogeraki, and T. Palpanas. Distributed Real-Time Detection and Tracking of Homogeneous Regions in Sensor Networks. In *RTSS*, Rio de Janeiro, Brazil, 2006.
20. S. Subramaniam, T. Palpanas, D. Papadopoulos, V. Kalogeraki, and D. Gunopulos. Online Outlier Detection in Sensor Data Using Non-Parametric Models. In *VLDB*, Seoul, Korea, 2006.

On the Schema Exchange Problem

Paolo Papotti and Riccardo Torlone

Università Roma Tre
 {papotti,torlone}@dia.uniroma3.it

1 Introduction

In this paper, we introduce the novel notion of *schema exchange*: a natural extension of the data exchange problem to an intensional level. Data exchange is a well-know problem: given a set of correspondences between a source and a target schema, the goal is the automatic generation of queries able to translate data over the source into a format conforming to the target [2, 5]. Schema exchange extends this problem to *sets* of similar schemas. To this aim, we first introduce the notion of *schema template*, which represents a class of database schemas sharing the same structure. Then, given a set of correspondences between a source and a target template, we provide a method for translating any schema conforming to the source template into a schema conforming to the target one. This framework allows the definition, once for all, of “generic” transformations that works for different but similar schemas, such as the denormalization of a pair of relation tables based on a foreign key between them.

In this scenario, given a source template T_1 describing a set of schemas, the user can: (i) impose the structure of the corresponding target schemas through a target template T_2 , and (ii) define how to exchange information between T_1 and T_2 by means of simple correspondences, graphically represented by lines.

We introduce a formal notion of *solution* for the schema exchange problem and propose a technique for the automatic generation of solutions. This is done by first representing constraints over templates and correspondences between them with a special class of first order formulas, and then using them to chase [3] the source schema. Also, we show how it is possible to automatically generate a data exchange setting from a schema exchange solution. This allows the definition of a set of queries to migrate data from a source database into the one obtained as a result of the schema exchange.

2 Schema exchange semantics

2.1 Schema templates

We fix a finite set \mathcal{C} of *construct names* and a countably set \mathcal{I} of *construct identifiers*. A *construct* $C(p_0, p_1, \dots, p_k)$ has a name C in \mathcal{C} and a finite set p_1, \dots, p_k of distinct *properties*, each of which is associated with a set of values called the *domain* of the property. In principle, the set \mathcal{C} can contain construct names from different data models so that we can define transformations between

schemes of different models. In this paper however, for sake of simplicity, we focus on schema exchange between schema templates of relational schemas with a small subset of properties; the approach can be extended to other types of templates, but challenging issues already arise in the relational case.

Therefore, we fix the following relational construct names and properties:

Construct Names	Properties (domain)
Relation (or R)	cid (\mathcal{I}), name (strings)
Attribute (or A)	name (strings), nullable (booleans), in (\mathcal{I})
AttributeKey (or AK)	name (strings), in (\mathcal{I})
AttributeFKKey (or AFK)	name (strings), in (\mathcal{I}), refer (\mathcal{I})

Note that the Relation construct is associated with a special property called cid whose domain consists of construct identifiers. This domain is also associated with the property in of the constructs Attribute, AttributeKey and AttributeFKKey, and the property refer of the construct AttributeFKKey: these properties are used to specify references between constructs.

A template is basically a set of constructs with a set of dependencies associated with them, which are used to specify constraints over single constructs and semantic associations between different constructs. A *dependency* is a first order formula of the form: $\forall \mathbf{x}(\phi(\mathbf{x}) \rightarrow \chi(\mathbf{x}))$ where $\phi(\mathbf{x})$ and $\chi(\mathbf{x})$ are formulas over the template, and \mathbf{x} are the free variables of the formula, ranging over the domains of the attributes occurring in the template. We focus on two special kinds of dependencies: the tuple generating dependencies (tgd) and the equality generating dependencies (egd), as it is widely accepted that they include all of the naturally-occurring constraints on relational databases. A tgd has the form: $\forall \mathbf{x}(\phi(\mathbf{x}) \rightarrow \exists \mathbf{y}(\psi(\mathbf{x}, \mathbf{y})))$ where $\phi(\mathbf{x})$ and $\psi(\mathbf{x}, \mathbf{y})$ are conjunction of atomic formulas, whereas an egd has the form: $\forall \mathbf{x}(\phi(\mathbf{x}) \rightarrow (x_1 = x_2))$ where $\phi(\mathbf{x})$ is a conjunction of atomic formulas and x_1, x_2 are variables in \mathbf{x} .

Definition 1 (Template). A (schema) template is a pair $(\mathbf{C}, \Sigma_{\mathbf{C}})$, where \mathbf{C} is a finite collection of constructs and $\Sigma_{\mathbf{C}}$ is a set of dependencies over \mathbf{C} .

Example 1. An example of a template $\mathcal{T} = (\mathbf{C}, \Sigma_{\mathbf{C}})$ contains the following set of constructs:

$$\mathbf{C} = \{ \text{Relation}(\text{cid}, \text{name}), \text{AttributeKey}(\text{name}, \text{in}), \text{Attribute}(\text{name}, \text{nullable}, \text{in}), \text{AttributeFKKey}(\text{name}, \text{in}, \text{refer}) \}$$

and the dependencies:

$$\begin{aligned} \Sigma_{\mathbf{C}} = \{ & d_1 = \text{Relation}(c, n_1) \wedge \text{Relation}(c, n_2) \rightarrow (n_1 = n_2), \\ & d_2 = \text{AttributeKey}(n_K, c_R) \rightarrow \text{Relation}(c_R, n_R), \\ & d_3 = \text{Attribute}(n_A, c_R) \rightarrow \text{Relation}(c_R, n_R), \\ & d_4 = \text{AttributeFKKey}(n_F, c_R, c'_R) \rightarrow \text{Relation}(c_R, n_R), \text{Relation}(c'_R, n'_R) \} \end{aligned}$$

The egd d_1 states the uniqueness of the cid of the Relation construct. The tgds d_2 and d_3 state the membership of keys and attributes to relations, respectively. Finally, the dependency d_4 states the membership of a foreign key to a relation and its reference to another relation.

For simplicity, in the following we will often omit the egds stating the uniqueness of the cid property (like d_1 in Example 1) and the membership dependencies between constructs (like d_2, d_3 and d_4 in Example 1), assuming that they belong to $\Sigma_{\mathbf{C}}$. Let us now introduce the notion of e-schemas. Basically, an e-schema corresponds to the *encoding* of a (relational) schema and is obtained by instantiating a template.

Definition 2 (E-schema). An e-schema component S over a given construct $C(p_1, \dots, p_k)$ is a function that associates with each property p_i ($1 \leq i \leq k$) a value a_i taken from its domain. A e-schema S over a template $(\mathbf{C}, \Sigma_{\mathbf{C}})$ is a finite set of e-schema components over constructs in \mathbf{C} that satisfy $\Sigma_{\mathbf{C}}$.

Example 2. A valid e-schema for the template of Example 1 is the following:

Relation	AttributeKey	Attribute	AttributeFKey																											
<table border="1"> <tr><td>cid</td><td>name</td></tr> <tr><td>r₁</td><td>Employee</td></tr> <tr><td>r₂</td><td>Department</td></tr> </table>	cid	name	r ₁	Employee	r ₂	Department	<table border="1"> <tr><td>name</td><td>in</td></tr> <tr><td>EmpName</td><td>r₁</td></tr> <tr><td>DeptNo</td><td>r₂</td></tr> </table>	name	in	EmpName	r ₁	DeptNo	r ₂	<table border="1"> <tr><td>name</td><td>nullable</td><td>in</td></tr> <tr><td>Salary</td><td>true</td><td>r₁</td></tr> <tr><td>Building</td><td>false</td><td>r₂</td></tr> </table>	name	nullable	in	Salary	true	r ₁	Building	false	r ₂	<table border="1"> <tr><td>name</td><td>in</td><td>refer</td></tr> <tr><td>Dept</td><td>r₁</td><td>r₂</td></tr> </table>	name	in	refer	Dept	r ₁	r ₂
cid	name																													
r ₁	Employee																													
r ₂	Department																													
name	in																													
EmpName	r ₁																													
DeptNo	r ₂																													
name	nullable	in																												
Salary	true	r ₁																												
Building	false	r ₂																												
name	in	refer																												
Dept	r ₁	r ₂																												

It is easy to see that this e-schema represents a relational table **Employee** with **EmpName** as key, **Salary** as attribute and **Dept** as foreign key, and a relational table **Department** with **DeptNo** as key and **Building** as attribute.

Note that e-schemas in Example 2 remind the common way commercial databases use to store metadata in catalogs. These relations have no special semantics, thus, with this approach any piece of information can be viewed either as data or as metadata and can be queried using a relational query language.

An e-schema S can be converted into a “standard” relational schema. Basically, this task requires the definition of formulas $\mathbf{R}\text{-DEC}(S) = (\mathbf{S}, \Sigma_{\mathbf{S}})$ that describe the semantics of the various components of an e-schema, according to the intended meaning of corresponding constructs.

Example 3. Let us consider the e-schema S of Example 2. The relational representation of S is: $\mathbf{R}\text{-DEC}(S) = (\mathbf{R}, \Sigma_{\mathbf{R}})$ where:

$$\mathbf{R} = \{\text{Employee}(\text{EmpName}, \text{Salary}, \text{Dept}), \text{Department}(\text{DeptNo}, \text{Building})\}$$

$$\Sigma_{\mathbf{R}} = \{ \text{Employee}(x_1, x_2, x_3), \text{Employee}(x_1, x'_2, x'_3) \rightarrow (x_2 = x'_2, x_3 = x'_3), \\ \text{Department}(x_1, x_2), \text{Department}(x_1, x'_2) \rightarrow (x_2 = x'_2), \\ \text{Employee}(x_1, x_2, x_3) \rightarrow \text{Department}(x_3, x'_2) \}$$

In the same line, a procedure for the *encoding* of a relational schema, that is for the transformation of a relational schema $(\mathbf{R}, \Sigma_{\mathbf{R}})$ into an e-schema $S = \mathbf{R}\text{-ENC}(\mathbf{R}, \Sigma_{\mathbf{R}})$, can also be defined.

2.2 Schema exchange

Given a source template $\mathcal{T}_1 = (\mathbf{C}_1, \Sigma_{\mathbf{C}_1})$, a target template $\mathcal{T}_2 = (\mathbf{C}_2, \Sigma_{\mathbf{C}_2})$, and a set $\Sigma_{\mathbf{C}_1\mathbf{C}_2}$ of *source-to-target dependencies*, that is, a set of tgds on $\mathbf{C}_1 \cup \mathbf{C}_2$, we denote a *schema exchange setting* by the triple $(\mathcal{T}_1, \mathcal{T}_2, \Sigma_{\mathbf{C}_1\mathbf{C}_2})$.

Definition 3 (Schema exchange problem). *Given a schema exchange setting $(\mathcal{T}_1, \mathcal{T}_2, \Sigma_{\mathbf{C}_1\mathbf{C}_2})$ and a source e-schema S_1 over $(\mathbf{C}_1, \Sigma_{\mathbf{C}_1})$, the schema exchange problem consists in finding a finite target e-schema S_2 over $(\mathbf{C}_2, \Sigma_{\mathbf{C}_2})$ such that $S_1 \cup S_2$ satisfies $\Sigma_{\mathbf{C}_1\mathbf{C}_2}$. In this case S_2 is called a solution for S_1 .*

Example 4. Consider a schema exchange problem in which the source template $\mathcal{T}_1 = (\mathbf{C}_1, \Sigma_{\mathbf{C}_1})$ and the target template $\mathcal{T}_2 = (\mathbf{C}_2, \Sigma_{\mathbf{C}_2})$ are the following:

$$\begin{aligned} \mathbf{C}_1 &= \{ \text{Relation}(\text{cid}, \text{name}), \text{AttributeKey}(\text{name}, \text{in}), \text{Attribute}(\text{name}, \text{in}) \} \\ \mathbf{C}_2 &= \{ \text{Relation}(\text{cid}, \text{name}), \text{AttributeKey}(\text{name}, \text{in}), \text{Attribute}(\text{name}, \text{in}), \\ &\quad \text{AttributeFKKey}(\text{name}, \text{in}, \text{refer}) \} \end{aligned}$$

with the corresponding membership constraints in $\Sigma_{\mathbf{C}_1}$ and in $\Sigma_{\mathbf{C}_2}$.

Assume now that we would like to split relations over \mathcal{T}_1 into a pair of relations over \mathcal{T}_2 related by a foreign key. This scenario is graphically shown (informally) in Figure 1 and is precisely captured by the following set of tgds $\Sigma_{\mathbf{C}_1, \mathbf{C}_2}$:

$$\Sigma_{\mathbf{C}_1, \mathbf{C}_2} = \{ \text{Relation}(c_R, n_R), \text{AttributeKey}(n_K, c_R), \text{Attribute}(n_A, c_R) \rightarrow \text{Relation}(c_R, n_R), \text{AttributeKey}(n_K, c_R), \text{AttributeFKKey}(n_F, c_R, c'_R), \text{Relation}(c'_R, n'_R), \text{AttributeKey}(n_F, c'_R), \text{Attribute}(n_A, c'_R) \}$$

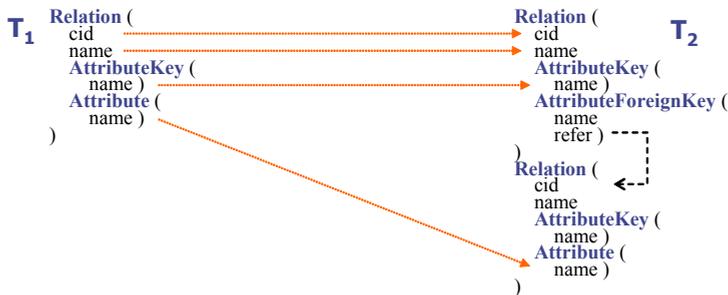


Fig. 1. Schema exchange scenario for Example 4

Consider now the following e-schema valid for \mathcal{T}_1 :

Relation	
cid name	
r ₁ Emp	

AttributeKey	
name in	
EmpName r ₁	

Attribute	
name in	
DeptName r ₁	
Floor r ₁	

This e-schema has one relation called Emp with EmpName as key and two attributes: DeptName and Floor. A possible solution S'_1 for this setting is:

Relation		AttributeKey		Attribute		AttributeFKKey		
cid	name	name	in	name	in	name	in	refer
r ₁	Emp	EmpName	r ₁	DeptName	R ₀	N ₁	r ₁	R ₀
R ₀	N ₀	N ₁	R ₀	Floor	R ₁	N ₃	r ₁	R ₁
R ₁	N ₂	N ₃	R ₁					

with $R_0, R_1, N_0, \dots, N_3$ as labelled nulls. Another solution is the following:

Relation		AttributeKey		Attribute		AttributeFKKey		
cid	name	name	in	name	in	name	in	refer
r ₁	Emp	EmpName	r ₁	DeptName	R ₀	N ₁	r ₁	R ₀
R ₀	N ₀	N ₁	R ₀	Floor	R ₀			

with R_0, N_0 , and N_1 as labelled nulls.

From Example 4 two issues arise: which solution to choose and how to generate it. Solution S'_2 in the example seems to be less general than S'_1 . To formally define this difference we first remind the *homomorphism* definition. An homomorphism from an instance I to an instance J is a function h from constant values and nulls occurring in I to constant values and nulls occurring in J such that: (i) it is the identity on constants, and (ii) (with some abuse of notation) $h(I) \subseteq J$.

It is now easy to see that, while there is an homomorphisms from S'_1 to S'_2 , there is no homomorphism from S'_2 to S'_1 . Actually, it can be shown that S'_1 has homomorphisms to all other solutions. It follows that S'_2 contains “extra” information whereas S'_1 is the “universal” solution. As in data exchange [1, 2], we argue that S'_1 is the “correct” solution and it is possible to show that it can be obtained by chasing the source e-schema using $\Sigma_{C_1 C_2} \cup \Sigma_{C_2}$.

3 The transformation process

Given a relational database over a schema \mathbf{S}_1 and schema exchange setting $(\mathcal{T}_1, \mathcal{T}_2, \Sigma_{C_1 C_2})$ such that the encoding S_1 of \mathbf{S}_1 is an instance of \mathcal{T}_1 , we aim at generating a target database over a schema \mathbf{S}_2 such that the encoding S_2 of \mathbf{S}_2 is a universal solution for S_1 .

Before discussing the transformation process, two preliminary notions are needed. First of all, in order to convert the schema exchange setting into a *data* exchange setting, we need to keep track of the correspondences between the source schema and the solution of the schema exchange problem. This can be seen as an application of the data provenance problem to schema exchange. To this end, by extending to our context a notion introduced in [4], we make use of *metaroutes* to describe the relationships between source and target metadata

Definition 4. Given an e-schema S and a set of dependencies Σ , a metaroute for S is an expression of the form $I_0 \rightarrow_{\sigma_1, h_1} I_1 \dots I_{n-1} \rightarrow_{\sigma_n, h_n} I_n$, where $I_0 \subseteq S$ and, for each $I_{i-1} \rightarrow_{\sigma_i, h_i} I_i$ ($1 \leq i \leq n$), it is the case that I_i is the result of the application of a chase step on I_{i-1} based on the dependency $\sigma_i \in \Sigma$ and the homomorphism h_i .

Note that since a small number of elements are involved in schema exchange, we can store all the metaroutes and we do not need to compute them partially and incrementally as in [4].

Metaroutes and homomorphisms are then used to derive *value correspondences* between source and target schemas

Definition 5. A value correspondence over two schemes \mathbf{S} and \mathbf{S}' is a triple $v = (t \in R, t' \in R', t.A_i = t'.A_j)$ where $R \in \mathbf{S}$, $R' \in \mathbf{S}'$, and $t.A_i = t'.A_j$ is a set of equalities over the attributes of R and R' respectively.

The generation of a target database from a data source according to a schema exchange setting can be summarized (informally) as follows.

1. \mathbf{S}_1 is encoded into an e-schema \mathbf{S}_1 ;
2. the chase procedure [3] is applied to \mathbf{S}_1 using $\Sigma_{\mathbf{C}_1, \mathbf{C}_2}$ and metaroutes are generated during the execution of the procedure: each chase step based on the dependency $\sigma_i \in \Sigma$ and the homomorphism h_i adds an element $I_{i-1} \rightarrow_{\sigma_i, h_i} I_i$ to the metaroute;
3. the result \mathbf{S}_2 of the chase procedure is decoded into a schema \mathbf{S}_2 ;
4. for each attribute A in \mathbf{S}_2 : (i) the metaroute $I_0 \rightarrow_{\sigma_1, h_1} I_1 \dots I_{n-1} \rightarrow_{\sigma_n, h_n} I_n$ such that A occur in I_n is selected, and (ii) A is annotated in \mathbf{S}_1 and \mathbf{S}_2 with the variable $h^{-1}(A)$, where $h = h_1 \circ \dots \circ h_n$;
5. the annotations of the attributes in \mathbf{S}_1 and \mathbf{S}_2 are used to derive value correspondences between them;
6. a data exchange process is applied to \mathbf{S}_1 and \mathbf{S}_2 using the generated value correspondences, on the basis of the methods presented in [5].

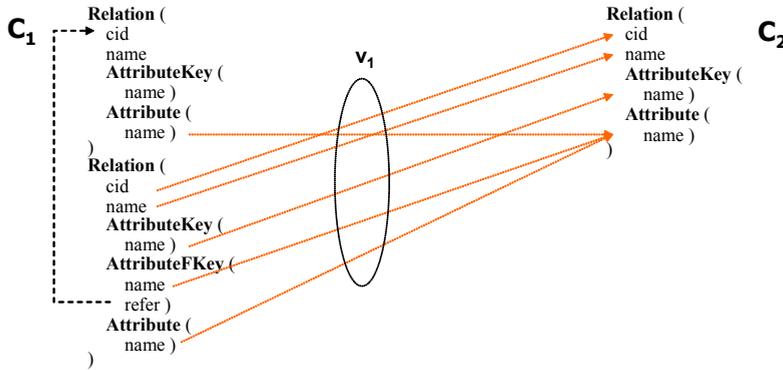


Fig. 2. Schema exchange scenario for Example 5.

Example 5. Let us consider the schema exchange setting described graphically in Figure 2 and represented by the following set of tgds $\Sigma_{\mathbf{C}_1, \mathbf{C}_2}$:

$$\{ v_1 = \text{Relation}(r, n_r), \text{AttributeKey}(n_k, r), \text{Attribute}(n_a, r), \text{Relation}(r', n'_r), \text{AttributeKey}(n'_k, r'), \text{Attribute}(n'_a, r'), \text{AttributeFKKey}(n_f, r', r) \rightarrow \text{Relation}(r', n'_r), \text{AttributeKey}(n'_k, r'), \text{Attribute}(n'_a, r'), \text{Attribute}(n_f, r'), \text{Attribute}(n_a, r') \}$$

Intuitively, the only constraint occurring in $\Sigma_{\mathbf{C}_1, \mathbf{C}_2}$ specifies that the target is obtained by joining two source relations according to a foreign key defined between them. Now consider the following source schema:

$$\begin{aligned} \mathbf{S} &= \{\text{Dept}(\text{id}, \text{dname}), \text{Emp}(\text{id}, \text{ename}, \text{dep})\} \\ \Sigma_{\mathbf{S}} &= \{ \text{Dept}(x_1, x_2), \text{Dept}(x_1, x'_2) \rightarrow (x_2 = x'_2), \\ &\quad \text{Emp}(x_1, x_2, x_3), \text{Emp}(x_1, x'_2, x'_3) \rightarrow (x_2 = x'_2, x_3 = x'_3), \\ &\quad \text{Emp}(x_1, x_2, x_3) \rightarrow \text{Dept}(x_3, x'_1) \} \end{aligned}$$

The encoding of \mathbf{S} is the e-schema \mathbf{S} that follows:

Relation	AttributeKey	Attribute	AttributeFKKey																		
s_1 <table border="1"><tr><td>cid</td><td>name</td></tr><tr><td>r₁</td><td>Dept</td></tr></table>	cid	name	r ₁	Dept	s_3 <table border="1"><tr><td>name</td><td>in</td></tr><tr><td>id</td><td>r₁</td></tr></table>	name	in	id	r ₁	s_5 <table border="1"><tr><td>name</td><td>in</td></tr><tr><td>dname</td><td>r₁</td></tr></table>	name	in	dname	r ₁	s_7 <table border="1"><tr><td>name</td><td>in</td><td>refer</td></tr><tr><td>dep</td><td>r₂</td><td>r₁</td></tr></table>	name	in	refer	dep	r ₂	r ₁
cid	name																				
r ₁	Dept																				
name	in																				
id	r ₁																				
name	in																				
dname	r ₁																				
name	in	refer																			
dep	r ₂	r ₁																			
s_2 <table border="1"><tr><td>r₂</td><td>Emp</td></tr></table>	r ₂	Emp	s_4 <table border="1"><tr><td>id</td><td>r₂</td></tr></table>	id	r ₂	s_6 <table border="1"><tr><td>ename</td><td>r₂</td></tr></table>	ename	r ₂													
r ₂	Emp																				
id	r ₂																				
ename	r ₂																				

Let $\{s_1, \dots, s_7\}$ be the e-components of \mathbf{S} . The application of the chase based on the given tgd produces the set of e-schema components $\{t_1, \dots, t_5\}$:

Relation	AttributeKey	Attribute												
t_1 <table border="1"><tr><td>cid</td><td>name</td></tr><tr><td>r₂</td><td>Emp</td></tr></table>	cid	name	r ₂	Emp	t_2 <table border="1"><tr><td>name</td><td>in</td></tr><tr><td>id</td><td>r₂</td></tr></table>	name	in	id	r ₂	t_3 <table border="1"><tr><td>name</td><td>in</td></tr><tr><td>ename</td><td>r₂</td></tr></table>	name	in	ename	r ₂
cid	name													
r ₂	Emp													
name	in													
id	r ₂													
name	in													
ename	r ₂													
		t_4 <table border="1"><tr><td>dep</td><td>r₂</td></tr></table>	dep	r ₂										
dep	r ₂													
		t_5 <table border="1"><tr><td>dname</td><td>r₂</td></tr></table>	dname	r ₂										
dname	r ₂													

The metaroute generated by this chase step is: $\{s_1, \dots, s_7\} \rightarrow_{v_1, h_1} \{t_1, \dots, t_5\}$, where h_1 is the homomorphism:

$$\begin{aligned} \{ r \mapsto r_1, n_r \mapsto \text{Dept}, n_k \mapsto \text{id}, n_a \mapsto \text{dname}, r' \mapsto r_2, \\ n'_r \mapsto \text{Emp}, n'_k \mapsto \text{id}, n'_a \mapsto \text{ename}, n_f \mapsto \text{dep} \} \end{aligned}$$

The chase ends successfully and produces an e-schema \mathbf{S}' whose decoding is the schema $(\mathbf{S}', \Sigma_{\mathbf{S}'})$ where:

$$\begin{aligned} \mathbf{S}' &= \{\text{Emp}(\text{id}, \text{ename}, \text{dep}, \text{dname})\} \\ \Sigma_{\mathbf{S}'} &= \{ \text{Dept}(x_1, x_2, x_3, x_4), \text{Dept}(x_1, x'_2, x'_3, x'_4) \rightarrow (x_2 = x'_2, x_3 = x'_3, x_4 = x'_4) \} \end{aligned}$$

Now, on the basis of the above metaroute, source and target schema can be annotated as follows:

$$\begin{aligned} \mathbf{S} &= \{\text{Dept}(\text{id}[n_k], \text{dname}[n_a]), \text{Emp}(\text{id}[n'_k], \text{ename}[n'_a], \text{dep}[n_f])\} \\ \mathbf{S}' &= \{\text{Emp}(\text{id}[n'_k], \text{ename}[n'_a], \text{dep}[n_f], \text{dname}[n_a])\} \end{aligned}$$

The value correspondences between \mathbf{S} and \mathbf{S}' easily follow:

$$\begin{aligned} v_1 &= (d \in \mathbf{S}.\text{Dept}, e \in \mathbf{S}'.\text{Emp}, d.\text{dname} = e.\text{dname}) \\ v_2 &= (e \in \mathbf{S}.\text{Emp}, e' \in \mathbf{S}'.\text{Emp}, e.\text{id} = e'.\text{id}, e.\text{ename} = e'.\text{ename}, e.\text{dep} = e'.\text{dep}) \end{aligned}$$

We then obtain the data mapping scenario reported graphically in Figure 3.

In the spirit of [5] we are now able to automatically generate a schema exchange setting. Given the source schema \mathbf{S} , the target schema \mathbf{S}' with their constraints, and the value correspondences we obtain the following tgd:

$$t_1 = \mathbf{S}.\text{Emp}(ss, en, d), \mathbf{S}.\text{Dept}(d, dn) \rightarrow \mathbf{S}'.\text{Emp}(ss, en, d, dn)$$

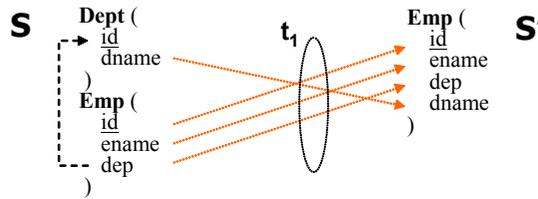


Fig. 3. Data exchange scenario for Example 5.

A number of results can be shown. First, the fact that the chase terminates and its output is a “correct” result, that is, is a solution of the schema exchange problem and is the most general one in the sense illustrated in Section 2.2. Some completeness result can also be shown. For instance the fact that any possible solution of the problem can be reduced to a result of the chase process.

4 Directions of research

We believe that several direction of research can be pursued. We just list some of them.

- *Metaquerying*. A template is actually a schema and it can therefore be queried. A query over a template is indeed a *meta* query since it operates over meta-data. There are a number of meta-queries that are meaningful. For instance, we can retrieve with a query over a template the pairs of relations that can be joined, being related by a foreign key. Also, we can verify whether there is a join path between two relations.
- *Special class of solutions*. Given a schema exchange problem, can we verify whether all the solutions of the problem satisfy some relevant property? For instance, we would like to obtain only relations that are acyclic or satisfy some normal form. We are also investigating under which conditions a schema exchange problem generates a data exchange setting with certain properties, e.g., the fact that the dependencies belong to some relevant class.
- *Combining data and metadata*. The framework we have presented can be extended to support mappings and constraints involving data and metadata at the same time. This scenario also allows the user to specify the transformation of metadata into data and vice versa. For instance, we could move the name of a relational attribute into a tuple of a relation.

References

1. R. Fagin, P. G. Kolaitis, and L. Popa. Data exchange: getting to the core. *ACM Trans. Database Syst.*, 30(1):174–210, 2005.
2. R. Fagin, P. G. Kolaitis, R. J. Miller, and L. Popa. Data exchange: Semantics and Query Answering. *Theor. Comput. Sci.*, 336(1):89–124, 2005.
3. C. Beeri, M. Y. Vardi. A Proof Procedure for Data Dependencies. *J. ACM*, 31(4):718–741, 1984.
4. L. Chiticariu, W. C. Tan Debugging Schema Mappings with Routes. *VLDB*, 79–90, 2006.
5. L. Popa, Y. Velegrakis, R. J. Miller, M. A. Hernández, R. Fagin. Translating Web Data. *VLDB*, 598–609, 2002.

Beyond Anonymity in Location Based Services

Linda Pareschi and Claudio Bettini

DICo - University of Milan
{pareschi,bettini}@dico.unimi.it

Abstract. This paper presents ongoing research on privacy issues in emerging data management applications. Current approaches to privacy preservation in location based services (LBS) have focused on anonymization techniques based on the spatio-temporal generalization of requests. These techniques prevent the identification of the issuer of a request among a group of potential issuers that happen to be in the same generalized region. In these approaches the only parameter characterizing the level of anonymity is the cardinality k of the group of potential issuers. Our research shows that when requests from multiple users for multiple services are considered, there are other important parameters affecting the ability of the attacker to identify an issuer. In this paper we investigate in particular the parameter characterizing the *diversity* of requested services and its relationship with the anonymity provided by request generalization.

1 Introduction

Location based services (LBS) pose new challenges for privacy preservation. With respect to requests for other services, LBS requests contain spatio-temporal information that may be used in various ways by malicious users to violate the privacy of the issuers. Using a pseudonym in the request instead of data that explicitly identify the user, like name, SSN, or phone number is not sufficient to preserve privacy. Indeed, both spatio-temporal information and specific service parameters included in the request can be used together with external knowledge to re-identify the issuer. As an example, a specific location and time may be used, by consulting data from web cams, to identify a person or a restricted group of people that were present in that location at that time.

The anonymity problem in LBS has at least two distinguishing aspects with respect to the analogous problem in the release of data from databases [8]. First, the fact that each request contains data about the location of the user at the time of request, introduces spatio-temporal data as a new kind of *quasi-identifier*, and it is well known that the effective management of this kind of data requires specific techniques. Second, anonymity in databases has been mostly studied considering a one-time publication of a given set of records, while the problem in LBS is inherently dynamic: the position of users is continuously changing and this has to be taken into account each time a request has to be anonymized. Moreover, inferencing based on previously anonymized requests can be used by the attacker.

The first attempts to protect privacy in LBS have mainly focused on anonymizing the user that issues a request by generalizing the location information that is forwarded to the service provider [4, 2, 7, 5, 3]. The degree of generalization characterizes the cardinality of the so called *Anonymity set*, i.e., the set of individuals that become indistinguishable for a possible attacker. These attempts have not considered that the presence of requests from other users may be relevant for an effective anonymization. They also seem to ignore that *i)* the generalization technique can be insufficient in certain cases (as observed in databases in [6]); *ii)* Different privacy threats can be identified depending on which information is considered *quasi-identifier* and which is considered *private information*, i.e., information that the user does not wish to be associated with her identity.

In this paper we report preliminary results of our investigation of these aspects focusing on a different parameter with respect to the cardinality of the anonymity set affecting the ability of the attacker to identify an issuer. We show that this parameter, named *diversity*, plays an important role if we assume that a malicious user may acquire generalized requests from multiple users. We also show that the technical characterization of these parameters changes depending on the privacy threats being considered, and on the assumptions about the external knowledge that may be available.

1.1 The LBS scenario

Our working scenario involves, as shown in Figure 1, three main entities:

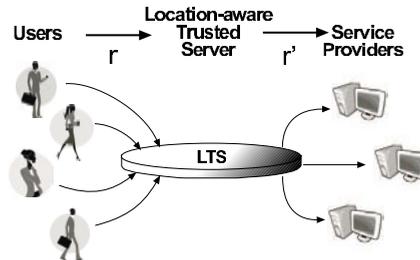


Fig. 1. The reference scenario.

- The **User** invokes or subscribes to location-based remote services. Each user can specify appropriate privacy policies for each kind of service. We assume that the current position and time of users is acquired locally and/or remotely from the infrastructure, and is stored by a Location-aware Trusted Server.
- The **Location-aware Trusted Server (LTS)** is supposed to have knowledge of the precise location data of a high number of users. The LTS either includes or it is tightly coupled with a component for privacy preservation.

Indeed, qualitative privacy preferences provided by each user are stored by the LTS and used to decide which user information is to be forwarded to a service provider and how.

- The **Service Provider (SP)** fulfills user requests and communicates with the user through the LTS.

Each user's request is composed by three logical elements dealing with information about the users' identity (*IDdata*), the spatio-temporal data of the request (*STdata*), and the required service(*SSdata*):

$$r = \langle IDdata, STdata, SSdata \rangle$$

We are planning to enrich the request with parameters derived by the declaration of user's privacy preferences: part of these data could be integrated in the already present fields according to their semantic, others could be grouped in a new logical component coupled with the request at the issuing time. In our model each request r is processed by the LTS resulting into a request r' with the same logical components but appropriately generalized for satisfying the privacy preferences given by users; e.g. the $r'.IDdata$ never contains the user identity but it is generalized to a pseudoid or to the NULL value. Therefore, for each request r received by the LTS it exists only one generalized request r' from the LTS to the SP according to the privacy policies required with r . In this paper we limit our investigation to a *static* scenario; i.e., we assume that the LTS works in a given time granularity (e.g., seconds or minutes) generalizing to the granule the temporal component of $r'.STdata$.

1.2 External knowledge assumptions

In order to model the attacks that could be performed and the corresponding defense techniques, it is necessary to state the precise assumptions on the external knowledge that may be available to an attacker. The network connection between users and LTS is considered trustworthy while the one between LTS and the SPs is not. Moreover, the SP itself may not be trustworthy. This paper does not consider the worst case scenario in which the attacker has access to the whole history of requests, but limits the investigation to the case in which the attacker may occasionally get access to one of the lots of requests sent by the LTS in a given granule. Each of these lots contains at most one request for each user. According to the approach proposed in [1], we formalize the external knowledge that may be available in our reference scenario. The first assumption follows from the above discussion.

Assumption 1 *The attacker may have access to Γ_G , the knowledge about all the generalized requests sent by the LTS to the SPs in a given time granule, but won't be able to reason about relationships among requests in different granules.*

The second assumption states that an attacker may get to know the precise position of the users monitored by the LTS. Despite it is very unlikely that complete user location information is available it is actually possible that through

data acquired by cameras, physical observations, or other ways the information about who is in a specific place in a certain interval of time is acquired. When it is not possible to identify when this can occur, the following assumption is used to model a worst case scenario.

Assumption 2 *The attacker may have access to Γ_{ST} , the knowledge about each user's location in any time granule.*

The third assumption models the case in which the attacker may exclude candidate issuers based on the fact that they are not entitled to request the service with the specific parameters of the request.

Assumption 3 *The attacker may have access to Γ_{EU} , the knowledge, for each user, of the services (and service parameters) she is entitled to require.*

Our investigation shows how the degree of anonymity that can be achieved by generalizing LBS requests changes depending on which assumptions actually hold in a given scenario.

2 Threats and relevant defense parameters

A privacy threat is intended as the disclosure of the association between the user identity and the data explicitly defined as private information. Hence, the characterization of privacy threats depends both on which request parameters are considered private information, or leading to the disclosure of private information (*PI*), and which instead act as *quasi-identifier* (*QI*), i.e. information that may be used to derive the user identity.

Depending on which fields of a request are intended as PIs, we have identified two different threats:

1. (PI: *SSdata*) *This threat is represented by the disclosure of the association between the identity of the user and the service she has requested.* For example, a user does not want to reveal she is asking for the closest shopping center during her working-hours.
2. (PI: *STdata*) *This threat is represented by the disclosure of the association between the identity of the user and her location at the moment of the request.* For example, a user does not want to reveal that she is calling from a shopping center during her working-hours.

We can distinguish which components of a request act as QI depending on the combination of external knowledge assumptions holding in a specific scenario. We have identified three main cases:

- ($\Gamma_G \cup \Gamma_{EU}$). In this case, the information contained in $r'.SSdata$ may act as QI leading to the user identity. For example, the request for a service s tailored to a very specific market target, may easily restrict the candidate issuers of the request r to a restricted group.

- $(\Gamma_G \cup \Gamma_{ST})$. In this case, the information contained in $r'.STdata$ may act as QI leading to the user identity. For example, the fact that the request comes from a specific location, together with the knowledge of which users were present in that area, would easily restrict the candidate issuers of the request to a restricted group or even to the real issuer.
- $(\Gamma_G \cup \Gamma_{ST} \cup \Gamma_{EU})$ In this case, the combination of $r'.SSdata$ and $r'.STdata$ acts as QI leading to the user identity. For example, if Γ_{ST} together with $r'.STdata$ restricts the candidate issuers to a set of users based on location, this set can be further restricted, using Γ_{EU} and $r'.SSdata$, by considering who is entitled to require the specific service.

Conceptually, each threat can be avoided by increasing the uncertainty on the values of each component of the association. The uncertainty on the identification of individuals (Id) is known as *anonymity*, proposed in [8, 9] as a model for privacy preservation, while the uncertainty on the private values has been named differently in the literature (e.g., *uncertainty* is used in [10] and *diversity* in [6]).

In order to formally model the anonymity of users and the uncertainty on private values, in the following subsections we introduce two fundamental structures related to each generalized request r' : the *Anonymity Set* (*Aset*), and the *Diversity Request Set* (*DRset*).

2.1 The Anonymity Set and the Anonymity Degree k

Given a request r sent by a user u , the *Anonymity Set* of its generalized version $r'(Aset(r'))$ is the set of all the *potential issuers* of r , such that each user in $Aset(r')$ is indistinguishable from u . This set can be determined, based on the information released by r' , and on the external knowledge that may be available. Intuitively, since r' never contains the explicit identification of the issuer, it is only by joining information in r' with external knowledge containing identification information that may lead to specific candidate issuers. In the following we describe how $Aset(r')$ can be determined based on the different assumptions on the available external knowledge.

$\Gamma_G \cup \Gamma_{EU}$. In this case the only information that may lead to the issuer identification is the requested service, i.e. a user can be indistinguishable only among other users that could potentially make an LBS request with the same values for the *SSdata* component. According to the external knowledge assumptions, both the LTS and the attacker are aware, for each user u , of a set denoted *AdmissibleService*(u) collecting all the services (possibly with specific parameters) a user is entitled to require. In this case, $Aset(r')$ contains each user u such that *AdmissibleService*(u) includes the service characterized by $r'.SSdata$.

$\Gamma_G \cup \Gamma_{ST}$. In this case the only information that may lead to the issuer identification is the spatio-temporal data characterizing time and position of the issuer; hence, due to the static scenario that limits the investigation to a single time granule, the anonymity set is defined depending only on the location of the request. In this case, $Aset(r')$ contains each user u that happened to be in the same area as specified in $r'.STdata$ and in the same time-granule.

$\Gamma_G \cup \Gamma_{ST} \cup \Gamma_{EU}$. In this case both service and location information can be used to restrict the set of candidate users. Therefore, both the conditions explained for the previous cases must hold in order to include a user in $Aset(r')$; Indeed, $Aset(r')$ will contain each user that is entitled to require the same service defined by $r'.SSdata$ and that happened to be in the same area and time granule as specified in $r'.STdata$.

Regardless of the specific combination of knowledge assumptions on which the definition of the anonymity set is based, the degree of anonymity (k) of a request r' is the cardinality of $Aset(r')$.

2.2 The Diversity Request Set and the Diversity Degree l

The k -anonymity model for privacy preservation exposes the user privacy to the so called *homogeneity attack*: if all the users in the $Aset$ have actually issued a request and there is no difference in the PI value of the requests, the fact that the users are indistinguishable is not sufficient to avoid revealing the association between the identity of a user and her private information. In database systems the exposure to homogeneity attacks has been characterized by the notion of *l-diversity*, [6]: a k -anonymized group of potential users is said to be *l-diverse* if the records belonging to these users contain at least l different values of private information.

In the context of LBS requests, when Γ_G is assumed, the attacker may compare the different generalized requests issued in a given time granule. This comparison can lead to the *homogeneity attack*. In order to adapt the *l-diversity* notion to LBS, we define the *Diversity Request Set* of a generalized request r' as the collection of all the requests issued by users in $Aset(r')$ in the same time granule as r . Note that, in DB systems, the proposed notion of anonymized group obtained by the generalization process does not correspond to the $Aset$ of a generalized request r' in LBS: the $Aset$ includes all the potential issuers of r even if some of them have not actually issued any request, while the anonymized group in DB contains only generalized data actually released. The structure in LBS analogous to the anonymized group in DB systems is the $DRset(r')$, that contains only requests actually sent by users in the $Aset(r')$.

Analogously to anonymity, a degree l of diversity can be defined for LBS requests. However, its definition depends on the specific privacy threat we are considering. As defined in Section 2, the two main threats are the ones characterized by PI being $SSdata$ or $STdata$.

PI: SSdata. In this case, the diversity degree l of a request r corresponds to the number of different services characterized by $SSdata.S$ in the requests belonging to $DRset(r')$.

PI: STdata. Considering this threat, the diversity degree computed on $DRset(r')$ depends only on the location information included in r' ; indeed, by the above definition of $DRset$ all the requests in $DRset(r')$ have as temporal information the same time granule as r' . In this case, the intuitive definition of the degree of diversity would be the number of different locations appearing in requests

of $DRset(r')$. This is a reasonable definition if Γ_{ST} is not available to the attacker, and hence, the location of the user is not generalized. Γ_{ST} intended as the knowledge of the precise location of users cannot be assumed to be available, since this threat considers $STdata$ as the information to be protected (there is no way to protect information that is already public). However, in some cases it may be available to a limited degree of precision. Then, the exact location will be probably generalized by the LTS. In these cases, there may be a problem with the significance of the diversity degree as defined above. Indeed, some of the generalized locations may overlap, and essentially cover the same region; then the attacker may conclude that all the users that issued requests in $DRset(r')$, included the issuer of r , were actually located in that region. This would lead to a privacy violation if the region is sufficiently small to be considered sensitive by the user. We are evaluating solutions to this problem based on clustering areas that significantly overlap and using the number of clusters as the diversity degree. A similar technique may be used on different domains of PI values.

Finally, the degree l of diversity does not represent by itself a level of user's privacy protection. In order to contrast the homogeneity attack, it is actually sufficient to guarantee *2-diversity*, but it is quite intuitive that an higher level of diversity could be more effective in the privacy protection process. In practice, it is very important how the diversity degree interacts with other defense parameters, including the degree of anonymity k .

3 Influence of k and l on privacy preservation

In the following we report some preliminary results on the study that we are currently doing on the combination of fundamental defense parameters.

We first note that, in case of lack of diversity ($l = 1$), the value of $\frac{|DRset(r')|}{|Aset(r')|}$ represents the probability of the attacker identifying the issuer of r among the potential issuers in $Aset(r')$. Generalizing this intuition, we can provide a general privacy measure as the probability of reconstructing the sensitive association even when the LTS guarantees a certain diversity degree for a request r' . The first step of our investigation is the definition of the probability of a successful *uniform attack*, i.e., the probability of reconstructing the sensitive association when each user in $Aset(r')$ has the same probability to be the issuer of r . Since we assume the attacker has access to the knowledge Γ_G , he may have access to all the requests in a $DRset$, hence he is also able to group those requests that have the same value of PI. The higher is the cardinality of the group containing r' , the higher is the probability of identifying the group of users in the $Aset(r')$ that issued requests in $DRset(r')$ with the same value PI. Then, the formula provided above in the case of lack of diversity is generalized to $\frac{|group_{PI}(r')|}{|Aset(r')|}$, characterizing the probability of reconstructing the sensitive association between the issuer of r and the value of PI in r . Note that the value of $|group_{PI}(r')|$ strongly depends on the diversity degree; for example when requests are uniformly distributed over the different values of PI we have $|group_{PI}(r')| = \frac{|DRset(r')|}{l}$. The following example shows the main idea of our privacy measure.

Example 1. Consider a generalized request r' , whose $Aset$ and $DRset$ have cardinalities 8 and 4, respectively. Since one of the 4 requests that have been issued contains the value PI_1 of private information, and the other 3 contain the value PI_2 , the degree of diversity is 2. If the issuer has actually sent the request with the value PI_1 , then the probability of an attacker being able to obtain the sensitive association is $\frac{1}{8}$, while it increases to $\frac{3}{8}$ if the sensitive value is PI_2 .

4 Conclusions

In this paper we have summarized the results of our investigation on the notions of anonymity and diversity in location based service requests. In particular, we have shown how their technical characterization changes depending on the privacy threats being considered, and on the assumptions about the external knowledge that may be available.

References

1. C. Bettini, S. Mascetti, X. S. Wang, and S. Jajodia. Anonymity in location-based services: towards a general framework. In *Proc. of the 8th International Conference on Mobile Data Management (MDM)*. IEEE Computer Society, 2007.
2. C. Bettini, X. S. Wang, and S. Jajodia. Protecting privacy against location-based personal identification. In *Proc. of the 2nd workshop on Secure Data Management (SDM)*, volume 3674 of *LNCS*, pages 185–199. Springer, 2005.
3. B. Gedik and L. Liu. Location privacy in mobile systems: A personalized anonymization model. In *Proc. of the 25th International Conference on Distributed Computing Systems (ICDCS)*, pages 620–629. IEEE Computer Society, 2005.
4. M. Gruteser and D. Grunwald. Anonymous usage of location-based services through spatial and temporal cloaking. In *Proc. of the 1st International Conference on Mobile Systems, Applications and Services (MobiSys)*. The USENIX Association, 2003.
5. P. Kalnis, G. Ghinita, K. Mouratidis, and D. Papadias. Preserving anonymity in location based services. Technical Report B6/06, National University of Singapore, 2006.
6. A. Machanavaajhala, J. Gehrke, D. Kifer, and M. Venkitasubramaniam. l-diversity: Privacy beyond k-anonymity. *icde*, 0:24, 2006.
7. M. F. Mokbel, C.Y. Chow, and W. G. Aref. The new casper: query processing for location services without compromising privacy. In *Proc. of the 32nd International Conference on Very Large Data Bases (VLDB)*, pages 763–774. VLDB Endowment, 2006.
8. P. Samarati. Protecting respondents' identities in microdata release. *IEEE Transactions on Knowledge and Data Engineering*, 13(6):1010–1027, 2001.
9. L. Sweeney. k-anonymity: a model for protecting privacy. *Int. J. Uncertain. Fuzziness Knowl.-Based Syst.*, 10(5):557–570, 2002.
10. C. Yao, X.S. Wang, and S. Jajodia. Checking for k-anonymity violation by views. In *VLDB '05: Proceedings of the 31st international conference on Very large data bases*, pages 910–921. VLDB Endowment, 2005.

Towards a Definition of an Image Ontology

Antonio Penta¹, Antonio Picariello¹, Letizia Tanca²

¹ University Federico II, Dipartimento di Informatica e Sistemistica
via Claudio 21, 80125, Naples, Italy
{a.penta,picus@unina.it}

² Politecnico di Milano, Dipartimento di Elettronica e Informazione
P.zza L. Da Vinci, 32 - 20133 Milano - Italy
tanca@elet.polimi.it

Abstract. Archiving, organizing, and searching multimedia data in an appropriate fashion is a task of increasing importance. The ontology theory may be appropriately extended in order to face this challenging issue. In this paper we propose an extension of an Image Data Base model based on NF^2 with an Ontology model. We first describe NF^2 and Image Ontologies concepts, then we propose a general architecture for supporting creation and management of multimedia objects.

1 Introduction

In the last decade there has been a tremendous growth of multimedia data, usually provided to a user by means of complex internet applications. Two different research areas are still requiring a great effort: first, the automatic analysis and indexing of multimedia data; second, several applications related to the *semantic web* have renewed interest in the creation of knowledge-based systems, and, more in detail, in the cooperation among different applications in areas like natural language translation, medicine, standardization of product knowledge, electronic commerce, and geographic information systems, among others. More in particular, tons of works have been done regarding ontologies.

Let us consider the actual applications related to one of the previously cited applications. We do not have doubt that many (or all) of these applications will use multimedia, thus requiring : i) ontology re-definition in order to capture the peculiarities of multimedia data w.r.t. the new functionalities introduced, ii) methodologies and systems for creating multimedia ontologies. Thus, it is the authors' opinion that *multimedia ontologies* will quickly become a crucial component of modern web based systems.

Ontologies are formal and explicit representations of domain knowledge, typically expressed via linguistic terms that include *concepts*, *concept properties*, and *relationships between concepts*. In the recent years several standard description languages for the expression of concepts and relationships in domain ontologies have been defined; among these the most important are: Resource Description Framework Schema (RDFS), Web Ontology Language (OWL) and, for multimedia, the XML Schema in MPEG-7. Using these languages metadata can be fitted

to specific domains and purposes, yet still remaining interoperable and capable of being processed by means of standard tools and search systems.

By the way, the semantics of multimedia data, and in particular of *images*, is very hard to capture both in a manual and in an automatic way, because it cannot be simply viewed as a clear set of terms and links between them. The information carried by an image is inherently both complex and uncertain, therefore its semantics has a fuzzy nature: in order to manage this aspect we must modify the ontological definition of image.

Let us consider the picture in Figure 1. Using an image processing and analysis system, for example the one described in [1], it is possible to obtain a description of the content of this image in terms of color and shape, together with a certain degree of uncertainty that each image processing algorithm produces. For this picture we obtain, for example, the following values of color features with their degrees of uncertainty: $\{\langle Green, 0.6 \rangle, \langle White, 0.7 \rangle, \langle Gray, 0.89 \rangle, \langle Blue, 0.65 \rangle\}$, and the following values of shape features with their uncertainty degrees: $\{\langle Rectangle, 0.8 \rangle, \langle Ellipse, 0.7 \rangle\}$. The content of the image itself may be also manually or automatically associated, for example: $\{\langle A\ lake\ with\ a\ snowy\ mountain \rangle, 0.7\}$.



Fig. 1. A lake with a snowy mountain

Thus requirements of image data and, more in general, of multimedia data, cannot be satisfied by the classical, flat relational model; if we want to include the entire semantics of multimedia data, ontology concepts and languages such as OWL and the DL-based reasoning services that complement such languages should be extended.

The remainder of this extended abstract is organized as follows: in Section 2 we start from the previous work on fuzzy image databases and explain the concepts of fuzzy tuples and NF^2 relations; in Section 3 we present the extraction of an ontology from a NF^2 schema; in Section 4 we comment some related works while Section 5 presents the conclusions.

2 Image Database Models

In the Marr based theory of computational vision [7], the vision process starts with a 2-D or a 3-D spatial representation of a certain scene. At this level, that sometimes is also defined “*low level layer*”, the raw images representing the scene of interest are computationally and abstractly thought of as discrete functions of two variables defined on some bounded and usually regular region of a plane. The interaction with these raw data is twofold: on the one hand, the maps can undergo some filtering process in order to obtain new maps; on the other hand, and more interestingly from our point of view, the maps are observed in order to structure their perceptual organization, according to any task-committed semantics. From a computational point of view, these structures are related to the use of spatial data – including points, lines, rectangles, regions, surfaces, volumes –, color features, textures and shape features. Note that, in the vision process, this level of representation is also called the “*intermediate level layer*”. For example, colors are usually described through *color histograms* and several features have been proposed for texture and shapes, all exploiting spatial interactions among a number of low level features (pixels) in a certain region. The analysis of such features is used to perform semantically consistent tasks, in order to extract the content of a certain image. We say that this last layer is the “*high level layer*”, related to the semantic concept conveyed by an image: an image containing circles, and a “lot of red” may “perhaps” represent a sunset, and an image containing a “face-like” shape may “perhaps” contain a portrait. The feature set can be easily characterized as a set of *logical predicates* that, due to the noise of a visual process, requires to be modeled in an appropriate fashion, and a fuzzy theory is considered as the natural framework to model them. In general, we say that any image may be characterized in terms of a number of fuzzy measurements on the image itself: consider, for example, the shape of an object in an image. Intuitively, one observer might say that the shape is “highly” oval, or that it is “a little bit” rectangular. Expressions such as *highly*, *a little bit*, and so on, recall this notion of fuzziness implicitly related to the *similarity of visual stimuli*. In this work, we extend the model presented in [3]: for the sake of clarity, we first provide a short introduction to the general *fuzzy image database model* adopted, briefly introducing the main components of the image database systems, in order to motivate our data model and the related uncertainty concepts. Fuzzy image data base models can be interpreted as extensions of traditional data models using fuzzy set theory and possibility theory [10]. The usual set theoretic operators can be extended to fuzzy sets in different ways, depending on the specific semantics associated with the fuzzy logical connectives.

The proposed NF^2 image database model is an extension of the standard NF^2 : the considered domains and set operators are fuzzy domains and fuzzy set operators respectively.

Table 1. An NF^2 relation with a motivating example

File	Color	Shape	Content
Im1.jpeg	{⟨ Green,0.6⟩, ⟨ White,0.7 ⟩, ⟨ Gray,0.89⟩ ⟨ Blue,0.65⟩}	{⟨ Rectangle ,0.8⟩, ⟨ Ellipse ,0.7 ⟩}	{⟨ A lake with a snowy mountain,0.7⟩}

Definition 1 (fuzzy tuple). Let D_1, \dots, D_n be n domains. A fuzzy n -tuple is any element of the cartesian product $2^{D_1} \times \dots \times 2^{D_n}$, being 2^{D_i} the fuzzy powerset of D_i , that is, the set of all fuzzy subsets of D_i .

According to the definition, any n -tuple is an array $\langle v_1, \dots, v_n \rangle$, where each v_i is a set of elements from the corresponding fuzzy domain D_i .

A fuzzy relation schema is used to associate attribute names to the domains of tuples.

Definition 2 (NF^2 relational schema). A NF^2 relational schema is defined as a symbol R , that is the name of the NF^2 relation, and a set ($X = \{A_1, \dots, A_n\}$) of (names of) fuzzy attributes. The schema is usually denoted as $R(X)$.

A NF^2 relation is an instance of a NF^2 relation schema; that is, a NF^2 relation is a set of fuzzy tuples, as stated in the following definition.

Definition 3 (NF^2 relation). Let $R(\{A_1, A_2, \dots, A_n\})$ be a relational schema. A NF^2 relation, defined over R , is a set of fuzzy tuples $t = \langle v_1, \dots, v_n \rangle$ such that each v_i is a fuzzy subset of $dom(A_i)$

The fuzzy relation of Table 1 has four attributes: **File**, **Color**, **Shape** and **Content**. As can be seen in this example, each attribute value is a set of $\langle domain_value, fuzzy_value \rangle$ pairs.

Note that the above data model is particularly suited for representing: low level features, intermediate level features and high level features, together with the associated uncertainty.

3 Extending an NF^2 schema to an Extracted Image Ontology

So far we have described how a NF^2 schema can represent much more information than a classical relational schema. In this paper, we propose a novel strategy having the aim of extending the NF^2 schema *with* an ontology derived from it. In this way we can use the expressivity of the ontology language in order to express both explicit and newly derived implicit knowledge from the image data. In addition, we explicitly note that the use of an ontology as an intermediate level between the application modules and the data can enhance the integration capability of different and distributed sources, the semantic retrieval of the data and more efficient content-based querying.

3.1 Image Ontology

We first define a *new concrete domain* in order to enrich the expressivity of the OWL language taking into account the specificity of image data.

Definition 4 (Image Concrete Domain). Let CS be a given Color Space, e.g. RGB , HSV , and so on. A CS -Image Concrete Domain is a pair $CS^D = (\Delta_{CS^D}, (\cdot)^{CS^D})$, $\Delta_{CS^D} \subset \mathbb{R}^3$ being a fixed set, whose elements are arrays of pixels, and $(\cdot)^{CS^D}$ being a function that assigns a set of binary relations r^{CS^D} over Δ_{CS^D} to each raw pixel image.

The definition of concrete domain provides a description of an image as a set of pixels in a given Color Space; each pixel is in relation r^{CS^D} with all its 8-connected pixels. We are now in a position for defining an Image Ontology.

Definition 5 (Image Ontology). An Image Ontology ImO is an OWL-DL ontology extended with an Image Concrete Domain CS^D .

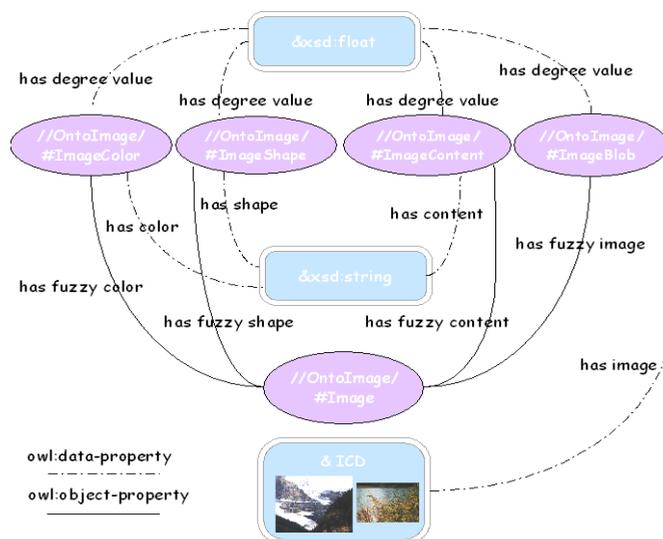


Fig. 2. The image ontology obtained from Table 1

The main problem to address is to define a methodology for building an ontology schema ImO starting from an NF^2 relational schema, re-defining in a suitable way standard *data base reverse-engineering* methodologies. In particular object-to-object relationship will be inferred from the foreign keys, while data properties will be derived from the other attributes. In the following we show an example that, without loss of generality, is particularly useful for describing the used approach. Let us consider the ontology schema in Fig. 2, derived from

the NF^2 schema in Table 1. In particular, we have the following classes, defined under the name space *OntoImage*:

- the *OntoImage:Image* class and the *OntoImage:ImageBlob* class, created considering the key attribute *File* of the NF^2 table ;
- a set of *owl:class*, one for each non-key attribute (color, shape, content): the *OntoImage:ImageColor* class for the color attribute, the *OntoImage:ImageShape* class for the shape attribute, the *OntoImage:ImageContent* for the content attribute;
- a set of *owl:objectproperty*, connecting the previous classes: *OntoImage:has fuzzy color*, *OntoImage:has fuzzy shape*, *OntoImage:has fuzzy content*, *OntoImage:has fuzzy image* ;
- a set of *owl:dataproperty*: the *OntoImage:has degree value*, *OntoImage:has color*, *OntoImage:has shape*, *OntoImage:has content*, *OntoImage:has image* that has Image Concrete Domain(&ICD) as range .

Several problems have to be addressed in order to transform NF^2 relations into ontologies. First, we note that NF^2 schema is made up of n-ary relations in non-first normal form; differently, in standard ontology languages we can express only unary or binary predicates. In order to solve this problem, we propose the owl-reification pattern, thus using - in a suitable way - some classes with some restriction over them, as described by a W3C working group note [8]. The second main problem is related to the identifiers of the objects belonging to such classes: we propose that these identifiers be generated by means of an appropriate invertible mapping, operating on NF^2 instances. In other words, we are addressing the well known *impedence mismatch* between a relational schema and an ontology schema.

Let us come back to our example. The mapping function can be implemented using a hash table – in NF^2 form – containing the pairs $\langle NF^2 \text{ instance, owl:class identifier} \rangle$, as follows:

NF^2 values	owl:classes identifier
Im1.jpeg	$\langle \text{OntoImage:Image}, 324 \rangle$
$\{ \langle \text{Rectangle}, 0.8 \rangle, \langle \text{Ellipse}, 0.7 \rangle \}$	$\langle \text{OntoImage:ImageShape}, 346 \rangle$
....

Eventually, we give the following definition:

Definition 6 (Extracted Image Ontology). *An Extracted Image Ontology \mathcal{EImO} is a triple:*

$$\mathcal{EImO} = (\mathcal{ImO}, \mathcal{R}, \mathcal{M}) \quad (1)$$

where \mathcal{ImO} is an image ontology, \mathcal{R} is an NF^2 schema, and \mathcal{M} is a mapping between \mathcal{ImO} and \mathcal{R} .

We explicitly notice that the extended image ontology approach has more advantages than the one based on the NF^2 schema. Indeed, the use of ontologies

allows to make inferences over image data, in order to retrieve semantically important facts from low level and intermediate level properties.

In particular, the semantic content of an image data, starting from the low level concrete domain –for example the image in Figure 1– might be found by making inferences on its color and shape features. Indeed, by associating (e.g. merging) the image ontology with a domain ontology or a thesaurus, the color *white* is found to be connected to *snow*, while *blue* is connected to *lake*, thus we may infer the semantic content of an image – i.e. that the image is *a lake with a snow mountain* – along with a certain degree of uncertainty. Note that the connection between terms and concepts could be weighted in different ways, as based on concept affinity describe in [6]. Accordingly the use of fuzzy domain ontologies is not required, once the degree of concept affinity determines the fuzzy truth value of the returned image.

4 Related Work

In the last few years, several papers have been presented about both image ontologies and fuzzy extension of ontology theory. In almost all the previous works, multimedia ontologies are effectively used to perform *semantic annotation* of the media content by manually associating the terms of the ontology to the individual elements of the image or of the video [4], [9], [11], thus demonstrating that the use of ontologies can enhance classification precision and image retrieval performance. Instead of creating a new ontology from scratch, other approaches [2] extend WordNet to image specific concepts, using the annotated image corpus as an intermediate step to compute similarity between example images and images in the image collection. For solving the uncertain reasoning problems, the theory of fuzzy ontologies is presented in several works, as an extension of the ontologies with crisp concepts. Recently, in [13] the authors derive a fuzzy ontology using a fuzzy hierarchy created by fuzzy conceptual clustering based on FCA theory; other papers [5] present a complete fuzzy framework for ontologies. An extension of description logic in order to manage fuzziness is presented in [12]. Differently from the previous works in both research fields, we propose a formal definition of multimedia ontology, particularly suitable for capturing the complex semantics of images during the several steps of the image analysis process. We do not propose any extension of the usual ontology theory and languages, but manage uncertainty implementing ternary properties by means of a reification process, thus taking advantage of the many of existing reasoning systems.

5 Conclusions and Discussion

The definition and implementation of ontologies for multimedia data is still a challenging task. Anyway, such a kind of ontologies can be of great use in many application areas, such as: *content visualization*, *content indexing*, *knowledge sharing*, *learning* and, of course, *automatic reasoning*. In this work we have described how to extend an efficient relational model based on NF^2 relations to

an equivalent ontology system, in order to obtain a more semantically treatable model. We have implemented a prototype system in JAVA on top of the Oracle 10g and MySQL back ends on a Windows XP platform. The system implements all the operations described in this paper and is integrated with the visual extraction module described in [1].

References

1. Massimiliano Albanese, Angelo Chianese, Vincenzo Moscato, and Antonio Picariello. Experience of animate similarity concepts in multimedia database. In *ICDE Workshops*, 2006.
2. Yih-Chen Chang and Lecture Notes in Computer Science Hsin-Hsi Chen Proceedings of Cross-Language Evaluation Forum, C. Peters et al. (Eds.). Approaches of using a word-image ontology and an annotated image corpus as intermedia for cross-language image retrieval. In *Proceedings of Cross-Language Evaluation Forum, C. Peters et al. (Eds.), Lecture Notes in Computer Science*, 2006.
3. Angelo Chianese, Antonio Picariello, Lucio Sansone, and Maria Luisa Sapino. Managing uncertainties in image databases: A fuzzy approach. *Multimedia Tools Appl.*, 23:237–252, 2004.
4. Aijuan Dong and Honglin Li. Multi-ontology based multimedia annotation for domain-specific information retrieval. In *IEEE International Conference on Sensor Networks, Ubiquitous, and Trustworthy Computing*, 2006.
5. Chang-Shing Lee; Zhi-Wei Jian; Lin-Kai Huang;. A fuzzy ontology and its application to news summarization. *Systems, Man and Cybernetics, Part B, IEEE Transactions on*, 35:859 – 880, 2005.
6. A. M. Rinaldi M. Albanese, A. Picariello. A semantic search engine for web information retrieval: an approach based on dynamic semantic networks. In ACM Press, editor, *ACM SIGIR Semantic Web and Information Retrieval Workshop*., 2004.
7. D. Marr and T Poggio. From understanding computation to understanding neural circuitry. *Neuroscience Research Progress Bulletin*, 15:470–488, 1977.
8. Natasha Noy and Alan Rector. Defining n-ary relation on the semantic web. Technical report, W3C, 2006.
9. S.; Saathoff C.; Simou-N.; Dasiopoulou S.; Tzouvaras V.; Handschuh S.; Avrithis Y.; Kompatsiaris Y.; Staab S.; Petridis, K.; Bloehdorn. Knowledge representation and semantic annotation of multimedia contentvision. In *Image and Signal Processing, IEEE Proceedings*, 2006.
10. Antonio Picariello and Maria Luisa Sapino. A fuzzy algebra for image data bases. In *Multimedia Information Systems*, 2002.
11. J. Pan J.Z. Schreiber G. Smith J.R. Stamou, G. van Ossenbruggen. Multimedia annotations on the semantic web. *Multimedia, IEEE*, 13:86 – 90, 2006.
12. U. Straccia. A fuzzy description logic for the semantic web. In *Capturing Intelligence: Fuzzy Logic and the Semantic Web, Elie Sanchez, ed., Elsevier*., 2006.
13. Quan Thanh Tho, Siu Cheung Hui, Alvis Cheuk M. Fong, and Tru Hoang Cao. Automatic fuzzy ontology generation for semantic web. *IEEE Trans. Knowl. Data Eng.*, 18:842–856, 2006.

Extending Datalog for Matchmaking in P2P E-Marketplaces

Azzurra Ragone¹, Umberto Straccia² Tommaso Di Noia¹,
Eugenio Di Sciascio¹, Francesco M. Donini³

¹ Politecnico di Bari, Via Re David, 200, I-70125, Bari, Italy
{a.ragone,t.dinoia,disciascio}@poliba.it

² ISTI - CNR, Via G. Moruzzi 1, 56124 Pisa, Italy
straccia@isti.cnr.it

³ Università della Tuscia, via San Carlo, 32, I-01100, Viterbo, Italy
donini@unitus.it

Abstract. We present an approach to matchmaking in P2P e-marketplaces, which mixes in a formal and principled way Datalog, fuzzy sets and utility theory, in order to determine most promising matches between perspective counterparts.

Use of Datalog ensures the scalability of our approach to large marketplaces, while Fuzzy Logic provides a neat connection with logical specifications and allows to model soft constraints and *how well* they could be satisfied by an agreement. Noteworthy is that our approach takes into account in the peer-to-peer matchmaking also preferences of each counterpart and their utilities. This allows to rule out of the match list those counteroffers that, although seemingly appealing for the buyer, would probably lead to failure due to contrasting preferences of the seller, and paves the way to the actual negotiation stage.

1 Introduction

In e-commerce settings, matchmaking can be defined as the process of finding “good” counterparts for a given entry in an marketplace. While in e-marketplaces dealing with undifferentiate products (as commodities) the “best” counterpart is the one offering the “best” price per unit, in e-marketplace dealing with complex products —cars, houses, computers— the price is not the single feature to take into account. Also, ads can involve bundle of issues, *e.g.*, *Sports car with optional package including both GPS system and alarm system* or implications *e.g.*, *If a car has leather seats then it is also provided with air conditioning*, and some kind of logical theory, able to let users express their needs/offers, could surely help. A matchmaking system has to be able both to find “good” counterparts and to evaluate how “good” a counterpart is. Currently, many commercial sites force the buyer to enter her request browsing a predefined classification that may be completely unsuitable for the characteristics the buyer might have in mind *e.g.*, they require to enter a brand first, then a model of that brand, etc. while a buyer may be not interested in a specific brand, but only on some limitations on price and color. In this respect, one may say that they provide no matchmaking assistance: the matchmaker is the buyer herself.

To assist buyers and sellers in marketplaces, several research proposals on matchmaking systems were issued. They either try to compute a score of possible counterparts, based on textual information [24], or to compare the logical representations of supply and demand [9], or combine both scores and logic in some way [6, 14]. Our proposal falls in this last category, mixing in a formal way Datalog, Fuzzy sets, and Utility Theory. While the above

logic-based proposals do not tackle the scalability problem, our resort on Datalog ensures the scalability of our approach to large marketplaces. On the other hand, the resort on Fuzzy Logic ensures a neat connection with the logical specification, while allowing the system to give an explanation of suggestions in terms of how well the preferences could be satisfied by an agreement.

Since we want both buyer and seller equally satisfied, the matchmaker computes a score as the maximum value of the *product* of the utilities of the buyer u_β and the seller u_σ over all possible agreements. In this way both buyer's and seller's preferences are taken into account ruling out of the match list those counteroffers that, although seemingly appealing for the buyer, would probably lead to failure due to contrasting preferences of the seller. The remaining of the paper is as follow: to set the stage we introduce basics of Datalog^{topk} and requirements for matchmaking process. Then fuzzy soft constraints are presented followed by the description of fuzzy matchmaking process in Datalog^{topk}. Discussion about relevant related work and conclusions close the paper.

2 Top-k Datalog

We use an extension of *Datalog* (cf. [1, 5, 23]) as our representation and query tool. We extend it by allowing soft constraint predicates to appear in rules and queries (we call the language Datalog^{topk}). The proposal here extends the work [20] in which soft constraint predicates, *i.e.*, fuzzy predicates, may appear in a query rules only and is conceptually equivalent to [21]. Basically, we allow vague/fuzzy predicates to occur in rule bodies, which have the effect that each tuple in the answer set of a query has now a score in $[0,1]$. Datalog^{topk} addresses the problem to compute the top-k answers in case the set of facts is huge, without evaluating all the tuples' score. As matching a buyer's request with a seller's offer is a matter of degree, we will use Datalog^{topk} to find the top-k matchings. *Datalog^{topk}* is as Datalog except that we additionally allow fuzzy predicates to occur in Datalog rule bodies. Specifically, let r be an $n + 1$ -ary predicate. A Datalog^{topk} rule is of the form

$$r(\mathbf{x}, s) \leftarrow \exists \mathbf{y} \text{body}(\mathbf{x}, \mathbf{y}), s = f(p_1(\mathbf{z}_1), \dots, p_n(\mathbf{z}_n))$$

where

1. \mathbf{x} are the n distinguished variables;
2. s is the score variable, taking values in $[0, 1]$, and r is functional on s ;
3. \mathbf{y} are so-called *non-distinguished variables* and are distinct from the variables in \mathbf{x} ;
4. $\text{body}(\mathbf{x}, \mathbf{y})$ is a conjunction of Datalog atoms;
5. \mathbf{z}_i are tuples of constants or variables in \mathbf{x} or \mathbf{y} ;
6. p_i is an n_i -ary fuzzy predicate assigning to each n_i -ary tuple \mathbf{c}_i as score $p_i(\mathbf{c}_i) \in [0, 1]$;
7. f is a scoring function $f: [0, 1]^n \rightarrow [0, 1]$, which combines the scores of the n fuzzy predicates p_i into and overall query score to be assigned to the score variable s . We assume that f is *monotone*, *i.e.*, for each $\mathbf{v}, \mathbf{v}' \in [0, 1]^n$ such that $\mathbf{v} \leq \mathbf{v}'$, $f(\mathbf{v}) \leq f(\mathbf{v}')$ holds, where $(v_1, \dots, v_n) \leq (v'_1, \dots, v'_n)$ iff $v_i \leq v'_i$ for all i ;
8. We assume that the computational cost of f and all fuzzy predicates p_i is bounded by a constant.

We call $s = f(p_1(\mathbf{z}_1), \dots, p_n(\mathbf{z}_n))$ a *scoring atom*. For instance,

$$\begin{aligned} \text{CheapCar}(x, p, s) &\leftarrow \text{NewCar}(x), \text{CarPrice}(x, p), s = \max(0, 1 - p/15000) \\ \text{CheapCar}(x, p, s) &\leftarrow \text{SecondHandCar}(x), \text{CarPrice}(x, p), s = \max(0, 1 - p/7500) \end{aligned}$$

are two Datalog topk -rules looking for cheap cars, assigning to each car a score depending on its price. If the price of a new car is above 15000 the car is not considered as a cheap one, while the scoring function is increasing as the price lowers. Hence, it is quite natural that if we are looking for cheap cars one wants that the retrieved cars are sorted in decreasing order with respect its score, *i.e.*, degree of cheapness. Furthermore, as the database may contain thousands of tuples, one usually wants to retrieve just the top- k ranked ones.

A Datalog topk query program is a pair $\mathcal{P} = \langle \mathcal{P}', q \rangle$, where \mathcal{P}' is Datalog topk program, q is a query rule and q does not occur in \mathcal{P}' . Essentially, the difference between Datalog and Datalog topk is that now scoring atoms may appear in the rule body of “query rules”.

The basic inference services that concerns us is the top- k retrieval problem, where this latter is defined as:

Top- k retrieval: Given a Datalog topk program \mathcal{P} , retrieve the top- k ranked tuples $\langle \mathbf{c}, v \rangle$ that instantiate the query q and rank them in decreasing order w.r.t. the score v , *i.e.*, find the top- k ranked tuples of the answer set of q , denoted

$$ans_k(\mathcal{P}, q) = \text{Top}_k\{\langle \mathbf{c}, v \rangle \mid \mathcal{P} \models q(\mathbf{c}, v)\}.$$

For instance,

$$q(x, p, s) \leftarrow \text{CheapCar}(x, p, s)$$

is a query asking for cheap cars. The top- k ranked cars, according to the score (that depends on their price), is obtained by $ans_k(\mathcal{P}, q)$.

3 Matchmaking Requirements

In the whole paper we adopt as reference in examples a P2P automobile marketplace, and motivate our work in this domain. With respect to this e-marketplace the users can express different preferences on several features, *i.e.*, a buyer can specify conditional preferences, such as *if the car is a sports one, than the feeding has to be gasoline, or a cheap car, yet if the car is provided with a GPS system she is ready to pay up to 17000*.

Looking at the previous example, we notice the buyer’s requests can be split into two different parts. In fact, the buyer expresses as a *hard* requirement that if the car is a sports one then the feeding has to be gasoline and as a preference, *soft* constraint, that she is willing to buy a cheap car but she could spend some more if the car is equipped with a GPS system. Strict requirements represent what the buyer and the seller want to be necessarily satisfied in order to accept the final agreement – in our framework we call strict requirements *hard constraints*. Preferences denote issues they are willing to negotiate on – this is what we call *soft constraints*. Let us now introduce an example request, we will use to explain some aspects of our approach:

Example 1. Suppose to have a buyer’s request like “*I want a station wagon black or gray. Preferably I would like to pay less than 14,000 € furthermore I’m willing to pay up to 17,000 € if warranty is greater or equal than 100000 km. (I don’t want to pay more than 19,000 € and I don’t want a car with a warranty less than 60,000 km)*”. In this example we identify:

hard constraints = **Body Type:** Station wagon. **Color:** Black or Grey. **Price:** $\leq 19,000$ €. **Warranty:** $\geq 60,000$ km.

soft constraints = **Price:** $\leq 14,000$. **Warranty-Price:** if Warranty $\geq 100,000$ then Price $\leq 17,000$ €

3.1 Preferences and Utilities

In a matchmaking process, retrieving supplies matching the request taking into account only *hard constraints* would be trivial. The matchmaker should just look for agreements where *hard constraints* of both the buyer and seller are satisfied. Given a request and a set of retrieved supplies, how should the matchmaker find –and rank– the most *suitable* or *promising* agreements to propose to both parties? The matchmaker should exploit *soft constraints* expressed both by the buyer and the seller.

Among the supplies completely satisfying all the requirements modeled in *hard constraints*, the top-ranked ones will be those best satisfying features expressed in the *soft constraints* proposed both by the buyer and the seller.

For what concerns *soft constraints*, we observe that the buyer's and seller's satisfaction degree, with respect to a final agreement, depends on which parts of her preference specifications have been satisfied. For instance, w.r.t. Example 1 suppose to have the following two supplies, where only soft constraints have been specified:

σ' : **Body Type:** Station wagon. **Color:** Grey. **Price:** 16,000 €. **Warranty:** 200,000 km.

σ'' : **Body Type:** Station wagon. **Color:** Black **Price:** 13,000 €. **Warranty:** 50,000 km

Comparing these supplies with buyer's *soft constraints* we note that σ' satisfies the preference $\beta_2 = \{\mathbf{Warranty-Price:}$ if Warranty $\geq 100,000$ then Price $\leq 17,000$ € $\}$, while σ'' the preference $\beta_1 = \{\mathbf{Price:}$ $\leq 14,000$ $\}$. How to evaluate the best one? We expect the buyer assigns a positive utility value representing the preference relevance to sub-parts of *soft constraints*. In this case we assume utility values — $u(\beta_1)$ and $u(\beta_2)$ — both for β_1 and β_2 . Actually, the same holds from the seller's perspective. In fact, in a P2P e-marketplace the seller may express his preferences — *soft constraints* e.g., on selling price, warranty, delivery time — with corresponding utilities $u(\sigma_j)$, as well as his *hard constraints* (e.g., color, model, engine fuel, etc.).

The only constraint on utility values is that both seller's and buyer's ones are normalized to 1 to eliminate outliers, and make them comparable [11].

$$\sum u(\beta_i) = 1 \qquad \sum u(\sigma_j) = 1 \qquad (1)$$

Since we assume utilities on preferences as additive, here we can write the global utility of the buyer u_β and of the seller u_σ as just a sum of the utilities of preferences satisfied in the agreement. If we are able, for each preference β_i of the buyer and σ_j of the seller, to evaluate a score s_i and s_j representing a degree of preference satisfaction, then it is reasonable to think that the global utility has to take into account also these information. Hence, in formulas, we write the two utility functions as:

$$u_\beta = \sum s_i * u(\beta_i) \qquad u_\sigma = \sum s_j * u(\sigma_j) \qquad (2)$$

From Example 1, we note that while considering numerical features, it is still possible to express hard and soft constraints on them. A hard constraint on a numerical feature can be considered as a *reservation value* [18] on the feature itself.

3.2 The Matchmaking Process

Based on the notions of preferences, utility, and reservation values introduced so far, we begin outlining the actual matchmaking process. Every time a seller (buyer) enters the marketplace, he proposes his supply (request) expressing both *hard constraints* and *soft constraints* (preferences). Then, he sets his reservation value $r_{\sigma,f}$ on numerical feature f . Eventually,

for each preference σ_j (β_i) he expresses the corresponding utility $u(\sigma_j)$ ($u(\beta_i)$). Based on buyer's and seller's specifications, the matchmaker returns a ranked list of supplies such that: [a] they satisfy both the *hard constraints* in the request and conversely their *hard constraints* are satisfied by the request; [b] the rank is evaluated taking into account preferences and utility functions u_β and u_σ as defined by equations (2). In a P2P e-marketplace, usually the aim is to find agreements maximizing not only the buyer's satisfaction or the seller's one, but trying to make them equally satisfied. So the matchmaker has to propose agreements mutually beneficial for both of them. Such agreements are computed considering the higher values of u_β and u_σ **utilities product** [16].

4 Requirements Fuzzy Representation

Marketplaces are typical scenarios where the notion of fuzziness is often involved. The concept of *Cheap* or *Expensive* are quite usual. Similarly, numerical variables involved in a commercial transaction expose a fuzzy behavior. For instance, suppose to have a buyer looking for a car provided with a warranty greater than 100,000 kilometers and a supplier selling his car with a 80,000 kilometers warranty. We can not say they do not match at all. Instead we can say they match with a certain degree. A logical language able to deal with fuzzy information would be then a good choice to model matchmaking. To represent buyer/seller requirements in a Datalog^{topk} setting, hereafter we will use the following notation:

$$\begin{aligned} \text{hard constraints} &= \begin{cases} \beta(x, \mathbf{y}) \text{ or } \beta(x) , \text{ buyer's strict requirements} \\ \sigma(x, \mathbf{y}) \text{ or } \sigma(x) , \text{ seller's strict requirements} \end{cases} \\ \text{soft constraints} &= \begin{cases} \beta_i(x, \mathbf{y}, s) \text{ or } \beta_i(x, s) , \text{ buyer's preferences} \\ \sigma_j(x, \mathbf{y}, s) \text{ or } \sigma_j(x, s) , \text{ seller's preferences} \end{cases} \end{aligned}$$

where x is a single variable, \mathbf{y} (if present) represents a vector of numerical variables and s represents the score variable as defined in Section 2. Since a score is associated to each fuzzy predicate, we can compute the global utility based on the two utility functions in Section 3.1. The two predicates σ and β model the minimal requirements the buyer and the seller want to be satisfied in order to accept the final agreement. Notice that, if seller and buyer set hard constraints in conflict with each other, the corresponding supply will not be retrieved and no agreement will be reached. *Soft constraints* are modeled via Datalog predicates β_i for the buyer and σ_j for the seller, where each of them represents a sub-part of the buyer/seller preferences.

The use of x , y and s should be clearer looking at how buyer's request in Example 1 is formalized:

$$\begin{aligned}
\beta_A(x) &\leftarrow \text{StationWagon}(x) \\
\beta_B(x, p) &\leftarrow \text{CarPrice}(x, p), 0 \leq p \leq 19000 \\
\beta_C(x, kmw) &\leftarrow \text{KmWarranty}(x, kmw), kmw \geq 60000 \\
\beta_D(x) &\leftarrow \text{Grey}(x) \\
\beta_D(x) &\leftarrow \text{Black}(x) \\
\beta(x, p, kmw) &\leftarrow \beta_A(x), \beta_B(x, p), \beta_C(x, kmw), \beta_D(x) \\
\beta_1(x, p, s) &\leftarrow \text{CarPrice}(x, p), \\
&\quad LS(0, 100000, 14000, 16000, p, s) \\
\beta_2(x, p, kmw, s) &\leftarrow \text{KmWarranty}(x, kmw), \text{CarPrice}(x, p) \\
&\quad RS(0, 400000, 80000, 100000, kmw, s_1), \\
&\quad LS(0, 100000, 17000, 19000, p, s_2), \\
&\quad s = \max(1 - s_1, s_2)
\end{aligned}$$

5 Fuzzy Matchmaking in Datalog ^{topk}

Now we have all what is needed to model the matchmaking framework in a Datalog ^{topk} setting. First of all we show how to write the corresponding Datalog ^{topk} program $\mathcal{P}_{match} = \langle \mathcal{P}_I, \mathcal{P}_E \rangle$.

1. for each supply write the corresponding Datalog fact and add it to the Extensional Database \mathcal{P}_E ;
2. encode buyer's *hard constraints* requirements as a Datalog rule where the head contains the predicate $\beta(x, \mathbf{y})$ as shown in Section 4; add the rule to the Intensional Database \mathcal{P}_I ;
3. encode seller's *hard constraints* requirements as a Datalog rule where the head contains the predicate $\sigma(x, \mathbf{y})$; add the rule to \mathcal{P}_I ;
4. for each buyer's preference β_i , write the corresponding rule in Datalog ^{topk} where the head contains the predicate $\beta_i(x, \mathbf{y}, s)$ as shown in Section 4; add the rule to \mathcal{P}_I ; set the utility value $u(\beta_i)$ as shown in Section 3.1;
5. for each seller's preference σ_j , write the corresponding rule in Datalog ^{topk} where the head contains the predicate $\sigma_j(x, \mathbf{y}, s)$ as shown in Section 4; add the rule to \mathcal{P}_I ; set the utility value $u(\sigma_j)$ as shown in Section 3.1;
6. add to \mathcal{P}_I the rules:

$$\begin{aligned}
\text{Buyer}(x, \mathbf{y}, u_\beta) &\leftarrow \beta(x, \mathbf{y}_0), \beta_1(x, \mathbf{y}_1, s_1), \beta_2(x, \mathbf{y}_2, s_2), \dots, \\
&\quad u_\beta = u(\beta_1) \cdot s_1 + u(\beta_2) \cdot s_2 + \dots
\end{aligned} \tag{3}$$

$$\begin{aligned}
\text{Seller}(x, \mathbf{y}, u_\sigma) &\leftarrow \sigma(x, \mathbf{y}_0), \sigma_1(x, \mathbf{y}_1, s_1), \sigma_2(x, \mathbf{y}_2, s_2), \dots, \\
&\quad u_\sigma = u(\sigma_1) \cdot s_1 + u(\sigma_2) \cdot s_2 + \dots
\end{aligned} \tag{4}$$

- where for each variable in \mathbf{y} in the head of one of the two previous rules, the same variable occurs in at least one of the arrays of the corresponding body: $\mathbf{y}_0, \mathbf{y}_1, \mathbf{y}_2, \dots$;
7. add to \mathcal{P}_I the "query rule":

$$\begin{aligned}
\text{Match}(x, \mathbf{y}, u) &\leftarrow \text{Buyer}(x, \mathbf{y}_\beta, u_\beta), \text{Seller}(x, \mathbf{y}_\sigma, u_\sigma), \\
&\quad u = u_\beta \cdot u_\sigma
\end{aligned} \tag{5}$$

where for each variable in \mathbf{y} in the head of the rule, the same variable occurs in \mathbf{y}_β or \mathbf{y}_σ

Once we have the Datalog ^{topk} program \mathcal{P}_{match} , then solve the following **Top- k retrieval** problem:

$$ans_k(\mathcal{P}_{match}, Match) = \text{Top}_k\{\langle x, \mathbf{y}, u \rangle \mid \langle \mathbf{y}, u \rangle \in \text{Top}_1\{\langle x, \mathbf{y}', u' \rangle \mid \mathcal{P} \models Match(x, \mathbf{y}', u')\}\}.$$

Basically, for each key value x of the database, we compute the best matching $\langle \mathbf{y}, u \rangle$ for it, *i.e.*, $\langle \mathbf{y}, u \rangle \in \text{Top}_1\{\langle x, \mathbf{y}', u' \rangle \mid \mathcal{P} \models Match(x, \mathbf{y}', u')\}$, and then rank the top- k key values.

Notice that the rank is computed considering the product of buyer's and seller's utilities as stated at the end of Section 3.2.

6 Related Work and Discussion

In this paper we presented a fuzzy matchmaking approach exploiting Datalog to find the most promising agreements in a P2P e-marketplace. More precisely, exploiting fuzzy rules we have been able to model *soft constraints*, while taking into account both buyer's and seller's preferences and utilities to find agreements mutually beneficial for them both. The P2P matchmaking process is modeled as a Datalog program able also to consider background domain knowledge while keeping the approach effective and scalable. A prototype is currently being implemented to further validate the approach through large scale experiments. Recently, the problem of matchmaking has been investigated under different perspectives and many approaches have been proposed. An initial approach to matchmaking can be dated back to vague query answering [15] where the need to go beyond pure relational databases was addressed using weights attributed to several search variables. More recently similar approaches have been proposed extending SQL with "preference" clauses, in order to allow relaxed queries in structured databases [10] where only buyer's preferences are taken into account while retrieving promising supplies. Classified-ads matchmaking, at a syntactic level, was proposed in [19] and [24] to perform a matchmaking between semi-structured descriptions. Approaches to matchmaking using LOOM as description language can be found, among others, in [3] and [8]. Due to the growing interest in the Semantic Web initiative many approaches to matchmaking have been proposed based on Description Logics (DL). Matchmaking as satisfiability of concept conjunction in DLs was first proposed in [9]. In the framework of Retsina Multiagent infrastructure [22], a specific language was defined for agent advertisement, and matchmaking engine was developed [17], which carries out the process on five possible match levels. This approach was later extended in [13], with two new levels for matching classification. A similar classification was proposed — in the same venue — in [7], along with properties that a matchmaker should have in a DL based framework, and algorithms to classify and semantically rank matches within classes. An initial DL-based approach, adopting penalty functions ranking, has been proposed in [4], in the framework of dating systems. An extended matchmaking approach, with negotiable and strict constraints in a DL framework has been proposed in [6], using both concept contraction and concept abduction. Approximation and ranking in DL-based approaches to matchmaking has also recently led to adopting fuzzy-DLs, as in sSMART [2] or hybrid approaches, as in the OWLS-MX matchmaker [12]. In [14] a language able to express conditional preferences is proposed to perform a matchmaking in Description Logics. Also in this case nothing is said on how to compute a agreement — as needed in P2P scenarios. Furthermore, the notion of fuzzy/vague requirements is not addressed.

References

1. S. Abiteboul, R. Hull, and V. Vianu. *Foundations of Databases*. Addison Wesley Publ. Co., Reading, Massachusetts, 1995.
2. S. Agarwal and S. Lamarter. smart - a semantic matchmaking portal for electronic markets. In *Proceedings of the 7th International IEEE Conference on E-Commerce Technology*, 2005.
3. Y. Arens, C. A. Knoblock, and W. Shen. Query Reformulation for Dynamic Information Integration. 6:99–130, 1996.
4. A. Cali, D. Calvanese, S. Colucci, T. Di Noia, and F. M. Donini. A description logic based approach for matching user profiles. In *Proceedings of the 17th International Workshop on Description Logics (DL'04)*, volume 104 of *CEUR Workshop Proceedings*, 2004.
5. S. Ceri, G. Gottlob, and L. Tanca. *Logic Programming and Databases*. Springer Verlag, 1990.
6. S. Colucci, T. Di Noia, E. Di Sciascio, F. Donini, and M. Mongiello. Concept Abduction and Contraction for Semantic-based Discovery of Matches and Negotiation Spaces in an E-Marketplace. *Electronic Commerce Research and Applications*, 4(4):345–361, 2005.
7. T. Di Noia, E. Di Sciascio, F. M. Donini, and M. Mongiello. A system for principled matchmaking in an electronic marketplace. *International Journal of Electronic Commerce*, 8(4):9–37, 2004.
8. Y. Gil and S. Ramachandran. PHOSPHORUS: a Task based Agent Matchmaker. In *Proc. International Conference on Autonomous Agents '01*, pages 110–111. ACM, 2001.
9. J. Gonzales-Castillo, D. Trastour, and C. Bartolini. Description Logics for Matchmaking of Services. In *Proceedings of the KI-2001 Workshop on Applications of Description Logics (ADL-2001)*, volume 44. CEUR Workshop Proceedings, 2001.
10. B. Hafenrichter and W. Kießling. Optimization of relational preference queries. In *The Sixteenth Australasian Database Conference (ADC 2005)*, pages 175–184, Newcastle, Australia, Jan. 2005.
11. R. L. Keeney and H. Raiffa. Decisions with multiple objectives - preferences and value trade-offs. *Cambridge University Press*, 1993.
12. M. Klusch, B. Fries, M. Khalid, and K. Sycara. Owls-mx: Hybrid owl-s service matchmaking. In *Proceedings of 1st Intl. AAI Fall Symposium on Agents and the Semantic Web*, 2005.
13. L. Li and I. Horrocks. A Software Framework for Matchmaking Based on Semantic Web Technology. In *Proc. of WWW '03*, 2003.
14. T. Lukasiewicz and J. Schellhase. Variable-strength conditional preferences for matchmaking in description logics. In *Proc. of KR 2006*, pages 164–174, 2006.
15. A. Motro. VAGUE: A User Interface to Relational Databases that Permits Vague Queries. *ACM Trans. Office Inf. Syst.*, 6(3):187–214, 1988.
16. J. F. Nash. The bargaining problem. *Econometrica*, 18 (2):155–162, 1950.
17. M. Paolucci, T. Kawamura, T. Payne, and K. Sycara. Semantic Matching of Web Services Capabilities. In *The Semantic Web - ISWC 2002*, number 2342, pages 333–347. 2002.
18. H. Raiffa, J. Richardson, and D. Metcalfe. *Negotiation Analysis - The Science and Art of Collaborative Decision Making*. The Belknap Press of Harvard University Press, 2002.
19. R. Raman, M. Livny, and M. Solomon. Matchmaking: distributed resource management for high throughput computing. In *Proceedings of IEEE High Performance Distributed Computing Conf.*, pages 140–146, 1998.
20. U. Straccia. Towards top-k query answering in deductive databases. In *Proceedings of the 2006 IEEE International Conference on Systems, Man and Cybernetics (SMC-06)*, pages 4873–4879. IEEE, 2006.
21. U. Straccia. Towards vague query answering in logic programming for logic-based information retrieval. In *World Congress of the International Fuzzy Systems Association (IFSA-07)*, Cancun, Mexico, 2007.
22. K. Sycara, M. Paolucci, M. Van Velsen, and J. Giampapa. The RETSINA MAS infrastructure. *Autonomous agents and multi-agent systems*, 7:29–48, 2003.
23. J. D. Ullman. *Principles of Database and Knowledge Base Systems*, volume 1,2. Computer Science Press, Potomac, Maryland, 1989.
24. D. Veit, J. Muller, M. Schneider, and B. Fiehn. Matchmaking for Autonomous Agents in Electronic Marketplaces. In *Proc. International Conference on Autonomous Agents '01*, pages 65–66. ACM, 2001.

On Homogeneity Evaluation and Seed Selection in Clustering Relational Data

Antonio Varlaro, Annalisa Appice, Antonietta Lanza, and Antonio Fittipaldi

Dipartimento di Informatica, Università degli Studi di Bari
via Orabona, 4 - 70126 Bari - Italy
{varlaro, appice, lanza}@di.uniba.it

Abstract. Clustering is a data mining task to group objects such that data inside each cluster model the continuity of some environment, while separate clusters model variation over it. CORSO is a multi-relational data mining method to discover clusters of structured objects possibly related each other according to some relation defining a discrete data structure. Clusters are built by merging partially overlapping sets of neighbors which are homogeneous with respect to the cluster description. The quality of clusters depends on the evaluation of cluster homogeneity as well as the selection of the objects which are seeds in the neighborhood construction. To face these issues, we illustrate some solutions whose validity is confirmed by experimental results on artificial and real data.

1 Introduction

Clustering is a form of unsupervised learning to group objects into classes of similar objects, called clusters. The task is to partition a data collection into clusters so that the resemblance between objects within individual clusters is high (cluster homogeneity), while the resemblance between objects from different clusters is low.

Clusters to be discovered are affected by the representational language used to describe the data, the possible background knowledge and the discovered model. The attribute-value description is the most commonly used in data mining methods due to the fact that this representation makes it possible to devise efficient algorithms that estimate resemblance of objects in terms of the pairwise resemblance of the values of each of their attributes. The resemblances along single attributes are combined into a single overall measure of objects resemblance.

Anyway, real-world data is seldom collected in the attribute-value format. In fact, data may have different properties which are modeled by as many data relations as the number of object types (multi-relational data). This leads to distinguish between *reference objects* of analysis and other *task-relevant objects*, and to represent their interactions. In this case, data is a collection of structured objects (or examples). Each example is descriptive of one reference object, zero, one or more task relevant objects, possibly of different type, as well as the interactions among them. The interactions involving two or more reference objects define relational constraints on separate examples (*discrete spatial structure*).

Since classical data mining methods do not make this distinction, nor do they allow the representation of any kind of interaction, the alternative is the use of a powerful representation language, e.g., subset of first order logic, that encompasses most of the real world clustering problems by dealing with both relational data and relational patterns. In [2] the authors presented CORSO, a multi-relational data mining method that is able to discover clusters of structured objects related each other according to some relation. Data is described in a first-order formalism and the resemblance between examples is computed as degree of matching with respect to a common generalization. In this work, we investigate two issues of the original proposal, which may affect the cluster quality, that is, the evaluation of cluster homogeneity and the selection of the seed objects.

The paper is organized as follows. In the next Section, we briefly present CORSO. In Section 3, we discuss some solutions to improve the evaluation of cluster homogeneity in CORSO, while in Section 4 we describe seed selection criteria. The validity of these solutions is then confirmed by some experimental results reported in Section 5. Finally, some conclusions are drawn.

2 Clustering Related Structure Objects

Theoretically, the problem solved by CORSO is formulated as follows:

Given a graph structure $G = \langle O, R \rangle$ representing a set of structured objects O related according to a binary relation R , and a background knowledge BK

Find a partition of O into a set of clusters C_i that are homogeneous and coherent with R .

An object $o_j \in O$ is described by a conjunction of ground atoms (see Example 1), while the BK is expressed by a set of definite clauses (see Example 2). In both cases, each atom is in the form $f(t_1, \dots, t_n) = v$, where f is a function symbol or *descriptor*, t_i are constants or variables (but not functions) and v is a value taken from the categorical or numerical range of f . In general, R is an asymmetric relation, such as the direct linkedness between objects, although symmetric relations, such as spatial adjacency, are also supported by CORSO.

Example 1. Let us consider data consisting of observations for a site (e.g., areal units) descriptive of one or more (spatial) primary units, possibly of different type, collected within the same site boundary. The areal units are the (structured) objects to be clustered, while the discrete data structure is naturally imposed by the spatial adjacency relation among areal units. Areal units are described in terms of both spatial and aspatial properties, such as:

```
arealunit(apulia) :- contain(apulia, bari), is_a(bari)=town, inhabitants(bari)=342129,
    contain(apulia, taranto), is_a(taranto)=town, distance(bari, taranto)=98, ...
arealunit(basilicata) :- contain(apulia, potenza), is_a(potenza)=town,
    inhabitants(potenza)=68141, contain(apulia, matera), is_a(matera)=town, ...
```

In this case $adjacent(apulia, basilicata)$ and $adjacent(basilicata, apulia)$ are two instances of the relation R (adjacency relation).

Example 2. Background knowledge is a source of domain independent knowledge. For example the definite clause:

$accessibility(X, Y) : -town(X) = XName, town(Y) = YName, cross(X, Z) = true, cross(Y, Z) = true, road(Z) = ZName.$

expresses accessibility of a town from another town by means of one road.

Cluster coherence with R means that two objects o_1 and o_2 can belong to the same cluster C_i only if a linking path exists from o_1 to o_2 (or vice-versa) according to R . Cluster homogeneity means that objects belonging to a cluster must be structurally similar each other, also respect to the given BK.

The cluster construction starts from an arbitrary object o (seed object) such that the R -based neighborhood N_o^R of o in G is an homogeneous cluster C . C is intentionally described by the cluster theory T_C that is a generalization of all objects in C . CORSO iteratively expands the cluster C by merging a partially “overlapping” neighborhood $N_{o_i}^R$ ($o_i \in C$), where $N_{o_i}^R = \{o_j \in O \mid o_i R o_j \text{ and } o_j \text{ is not yet classified into a different cluster}\}$.

The neighborhood $N_{o_i}^R$ is merged to the cluster C only if $N_{o_i}^R$ results an homogeneous set w.r.t the cluster theory T_{C_i} ($C_i = C \cup N_{o_i}^R$), that is:

$$h(N_{o_i}^R, T_{C_i}) = \frac{1}{\#N_{o_i}^R} \sum_{o_j \in N_{o_i}^R} fm(o_j, T_{C_i}) \quad (1)$$

where $\#N_{o_i}^R$ denotes the cardinality of $N_{o_i}^R$, while T_{C_i} includes both $\{T_1, \dots, T_w\}$, i.e., the model of C , and T_{w+1} , i.e., the model of $N_{o_i}^R$.

The model of a neighborhood $N_{o_i}^R$ is built as a set of first-order clauses such that $T_i : \{cluster(X) = c \leftarrow H_{i1}, \dots, cluster(X) = c \leftarrow H_{iz}\}$. Each H_{ij} is a conjunctive formula describing a sub-structure shared by one or more objects in $N_{o_i}^R$ ($\forall o_i \in N_i, BK \cup T_i \models o_i$). This set of first-order clauses is learned by means of the ILP system ATRE [1]. The function fm (flexible matching) returns a number in $[0, 1]$, that is, the probability of precisely matching o_j against T_{C_i} , provided that some change described by a substitution θ is possibly made in the (first-order) description of o_j .

In the case a new cluster is built then $C = \emptyset$ and the seed o is an object not yet assigned to any cluster. Further details on the algorithm are provided in [2].

3 Homogeneity Evaluation

CORSO builds clusters by merging partially overlapping neighborhoods, which are *homogeneous* according to the homogeneity function h . This neighborhood-based evaluation suffers from several problems.

Firstly, the homogeneity is estimated at the neighborhood level. This means that a cluster as well as its description is “incrementally” built without guaranteeing that the entire cluster results homogeneous with respect to its final description. To avoid this incoherence, we propose to modify CORSO in order to build clusters by merging contiguous neighborhoods, but evaluating homogeneity at *cluster level*. The neighborhood $N_{o_i}^R$ is merged to the cluster C if and only if the homogeneity of the candidate cluster $C_i = C \cup N_{o_i}^R$ is greater than a user defined threshold. Homogeneity of C_i is evaluated as:

$$h(C_i, T_{C_i}) = \frac{1}{\#C_i} \sum_{o_i \in C_i} fm(o_i, T_{C_i}), \quad (2)$$

where $T_{C_i} = \{T_C, T_{N_{o_i}^R}\}$ (motivations of this formula are extensively reported in [2]). Since homogeneity evaluation at cluster level is more complex than evaluation at neighborhood level, some caching techniques are applied to improve scalability of the algorithm, resulting in the following equation:

$$h(C_i, T_{C_i}) = \frac{\frac{\#T_C \cdot \#C \cdot h(C, T_C)}{\#T_{C_i}} + \frac{\#C \cdot h(C, T_{N_{o_i}^R})}{\#T_{C_i}} + \#(C_i - C) \cdot h(C_i - C, T_{C_i})}{\#C_i}. \quad (3)$$

Consequently, $h(C, T_C)$ values can be stored as basis for future evaluations.

Secondly, the homogeneity is estimated with respect to a cluster description (T_{C_i}) that is a set of first-order theories, one for each neighborhood merged to form the candidate cluster C_i . Originally, the theory for each neighborhood was learned independently from the theory currently associated with the cluster to be expanded. Hence, the same theory can be learned for separate neighborhoods by introducing duplicates in the evaluation of homogeneity. This can be avoided by changing the cluster model induction in order to take into account the current cluster description T_C when generalizing objects of a potential neighborhood. Practically, learning is done on only the neighbors of $N_{o_i}^R$ not covered by T_C itself. In this way, we improve efficiency of the cluster evaluation avoiding to recompute the homogeneity of an object with respect to the same theory (labeling several neighborhoods) more than one time. In addition, clusters are naturally labeled with more compact descriptions which are simpler to be interpreted.

Thirdly, the clustering expansion operates at neighborhood level. This prevents the expansion of a cluster by adding only a “portion” of a neighborhood so forcing the shape of discovered clusters. To avoid this problem, a single-neighbor based expansion is investigated when failing the neighborhood-based one.

4 Seed Selection Criteria

The cluster shape depends on the object that CORSO chooses at each step as seed of the neighborhood to be considered. CORSO adopts a sequential strategy. In the case a new cluster has to be discovered, the seed is the first object accessed in O not yet assigned to any cluster. In the case an existing cluster has to be expanded, the seed is the object stored in the first position of a list collecting the cluster objects not yet considered for the expansion step. The sequential seed selection (SEQ) is efficient, but quality of clustering clearly depends on the “order” of storing and accessing objects, which is not necessarily the best one. In alternative, we have empirically investigated some heuristics that take advantage from the graph structure in seed selection step. In particular, we base the choice of the candidate seed on the concept of density (cardinality) of a neighborhood.

For each candidate seed, the candidate “best” seed is the object whose neighborhood in the graph has the highest (DESC) or lowest (ASC) density. In the first case (DESC), we follow the intuition coming from the density-based framework where dense areas are labeled as clusters. Our suggestion is to estimate density in terms of the number of connections in the graph from the candidate seed to the objects not yet assigned to any cluster. Alternatively, a dense area in the graph may correspond to an area covered by contiguous different clusters, hence, discovery should preferentially start from peripheral objects (ASC).

5 Experimental Results

CORSO has been applied to artificial and real data. In this section, we present the application of CORSO to the computer network analysis and topographic map interpretation. The former application involves artificial data and aims at highlighting the improvement in clustering obtained with cluster-based homogeneity function and new seed selection criteria. The latter concerns real dataset in order to valuate quality of clustering in real-world domains.

5.1 Computer Network Analysis

In this experiment the goal is to apply clustering to artificial data (LAN) deliberately built to validate the method. Here, the arrangement of personal computers composing a LAN are described (see Figure 1). The task is to find groups of similar computers connected each others.

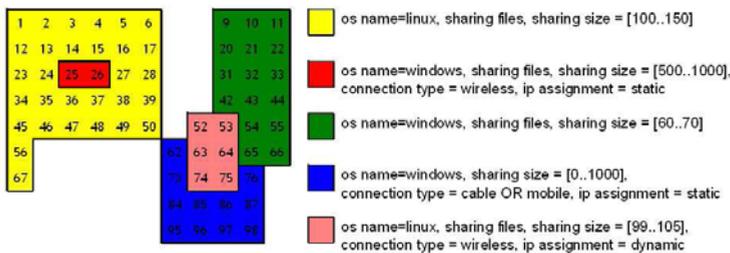


Fig. 1. The LAN dataset

Each structured object is identified by a natural number and graphically represented in the figure with a square. It corresponds to a PC described in terms of operative system used, shared option, share space size, type of net connection and type of IP assignment, while the discrete spatial structure is defined by the adjacency relation. PC are distributed in five different clusters. Each cluster is associated with a description and it is identified by a different color. For instance, the yellow cluster describes the set of contiguous computers

having Linux as o.s. and sharing an amount of HD space ranging from 100 MB to 150 MB. The arrangement of data has been artificially generated by including irregularly shaped (green and blue) and/or concentric (red and yellow) clusters.

Results obtained with different parameter configurations are shown in Figure 2 (homogeneity threshold is set to 0.95). In this study, we have investigated the cluster-based homogeneity function in combination with the sequential (SEQ), ascending (ASC) and descending (DESC) order of connectivity for the seed selection as well as the cluster expansion at level of single-neighbor (SNE).

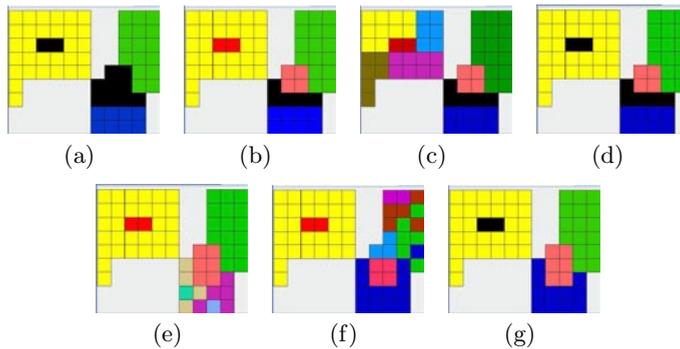


Fig. 2. Clusters discovered with the original version of CORSO (a) are compared with the clusters discovered by using the cluster-based homogeneity evaluation in different configurations: sequential (b), ascending (c) and descending (d) order for seed selection. In the second row, results obtained with sequential (e), ascending (f) and descending (g) order for seed selection and single-neighbor level in cluster expansion are shown.

Results confirm that, except for the ASC configuration, the quality of clusters is improved since the shape of clusters better fits the original one reported in Figure 1. At the same time, the number of objects labeled as noise (black objects) decreases from 11 in the original version of CORSO (neighborhood-based evaluation function and sequential seed selection) to 5 in SEQ and ASC configurations, 7 in DESC configuration when resorting to the cluster-based homogeneity evaluation. The ASC configuration detects a higher number of clusters as an evidence that preferring less connected objects in the seed selection generally leads to more fragmented results. This is a consequence of the fact that peripheral objects may belong to different clusters smoothing the discontinuity of some phenomenon (first Law of Geography, [3]). In particular, the original yellow cluster shown in Figure 1 is fragmented into four separate clusters (see Figure 2.c), but a deeper analysis reveals that the descriptions associated with this clusters are quite similar:

```

cluster(X)=yellow :- osName(X)=linux, sharedFiles(X), sharedSize(X) ∈ [100..150]
cluster(X)=brown :- osName(X)=linux, sharedFiles(X), sharedSize(X) ∈ [100..135]
cluster(X)=fuchsia :- osName(X)=linux, sharedFiles(X), sharedSize(X) ∈ [100..150]

```

$\text{cluster}(X)=\text{azure} \text{ :- osName}(X)=\text{linux}, \text{sharedFiles}(X), \text{sharedSize}(X) \in [100..125]$.

The clusters descriptions discovered with cluster-based homogeneity are simpler than those discovered with the neighborhood-based homogeneity evaluation. For example, the description of the yellow cluster in Figure 2.a is composed by 16 similar (or at worst identical) clauses, while the same description includes only one clause in the SEQ configuration (Figure 2.b). This depends on the fact that the cluster-based homogeneity evaluation re-uses the current cluster description when building the generalization of a candidate neighborhood thus avoiding to introduce redundant clauses in the final cluster description.

Finally, results reported in Figures 2.e,f,g confirm that the cluster expansion performed at single-neighbor level allows to discover irregularly shaped clusters. In fact, CORSO with SEQ seed selection is able to perfectly discover the pink cluster even if some errors are performed in detecting the blue one (see Figure 2.e). Still, the worst results are obtained with the ASC seed selection, although no object is labeled as noise. On the contrary, the DESC seed selection produces the best result, that is, it perfectly detects the actual clusters except for objects 25 and 26 that are labelled as noise during the expansion of contiguous clusters.

5.2 Topographic Map Interpretation

In this section we present a real-world application of clustering to characterize spatial continuity of some morphological elements over the topographic map of the Apulia region in Italy. In this case, each structured object represents a square region of the map, described in terms of geometrical and topological features concerning geographical objects in the territory such as roads, cultivations, fonts, rural areas, etc., while the discrete data structure is defined by adjacency relations among cells. The task is to group adjacent cells and identify morphologically homogeneous areas. Further details can be found in [2].

This study aims at comparing the original and the improved version of CORSO on a real dataset in order to investigate both the cluster quality and the running time. Results are depicted in Figure 3.

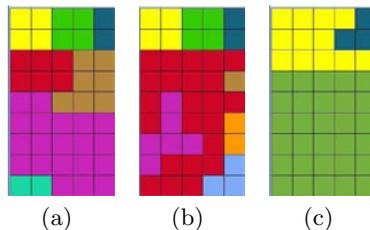


Fig. 3. Comparison of results obtained by running CORSO with the neighborhood-based evaluation function (a) and the cluster-based homogeneity function combined with the single-neighbor check and the sequential (b) or descending (c) order of connectivity in seed selection. All runs are performed with homogeneity threshold=0.99

Figure 3 shows that results change significantly when varying the homogeneity evaluation function. The cluster-based homogeneity function with the sequential seed selection (Figure 3.b) detects highly irregularly shaped clusters. This depends on the fact that the cluster-based evaluation also employees the check at level of single-neighbor. Furthermore, the models associated with clusters in Figures 3.b,c are more compact and simpler to read. For instance, the descriptions of the red and brown clusters in Fig. 3.a include the following clauses:

red(X):- grapevines(X) \in [2..19], hasStreet(X,Y), extension(Y) \in [11..1179],
street2parcel(Y,Z)=adjacent, area(Z) \in [262..249975]

red(X):- grapevines(X) \in [2..19], hasStreet(X,Y), extension(Y) \in [11..1179],
street2parcel(Y,Z)=adjacent, area(Z) \in [262..249975]

brown(X):- grapevines(X) \in [6..27], hasStreet(X,Y), extension(Y) \in [11..1115]

brown(X):- grapevines(X) \in [6..24], hasStreet(X,Y), extension(Y) \in [22..1115]

while approximatively a super set of the same area is modeled in Fig. 3.b by the red cluster that is described by the single clause:

red(X):- grapevines(X) \in [2..15], hasStreet(X,Y), extension(Y) \in [11..1023],
street2parcel(Y,Z)=adjacent, area(Z) \in [262..249975]

Looking at running times, CORSO with the cluster-based homogeneity function takes about 1.5 the time consumed by CORSO with the neighborhood-based one (2450 vs 1696 secs, on a Pentium IV 3.2 GHz, 1 GB of RAM). This is because the homogeneity evaluation is performed at cluster level instead of at neighborhood one and single-neighbor checks are performed. The loss of efficiency is partially mitigated by the optimized version, even if the low number of objects in the dataset reduces the effect of optimization (only 105 secs gained over 2450).

Finally, Figure 3.c shows that the descending order of connectivity in the seed selection decreases the number of clusters: starting from highly connected objects generates cluster models representative of an higher number of objects.

6 Conclusions

Issues concerning the homogeneity evaluation of clusters and the seed selection criteria in the MRDM clustering method CORSO are investigated and solutions are proposed. The effectiveness of presented solutions is confirmed by experimental results concerning both artificial and real data. Results confirm that applied changes have improved both the readability and quality of clusters. As future work, we intend to extend CORSO with classification capability, find discriminating features in clusters and apply CORSO to additional relational datasets.

References

1. D. Malerba. Learning recursive theories in the normal ilp setting. *Fundamenta Informaticae*, 57(1):39–77, 2003.
2. D. Malerba, A. Appice, A. Varlaro, and A. Lanza. Spatial Clustering of Structured Objects. In *ILP 2005*, pages 227–245. Springer, 2005.
3. W. Tobler. Cellular geography. In S. Gale and G.Olsson, editors, *Philosophy in Geography*, 1979.

Distributed Information Retrieval and Automatic Identification of Music Works in SAPIR

M. Agosti, E. Di Buccio, G.M. Di Nunzio, N. Ferro, M. Melucci, R. Miotto and N. Orio

Department of Information Engineering, University of Padua, Italy
{agosti, dibuccio, dinunzio, ferro, melo, miottori, orio}@dei.unipd.it

1 The Context

The *Search in Audio-visual content using Peer-to-peer Information Retrieval* (SAPIR) project¹ is an EU IST FP6 research project, started in January 2007. The SAPIR consortium includes experts from industry and academia, and it will provide major innovations for powerful peer-to-peer (P2P) search on audio-visual content. It will be based on a scalable, completely decentralized, largely self-organizing P2P system where peers act both as client and servers and the users produce audio-visual content using multiple devices – ultra-peers may act as service providers which maintain indexes and provide search capabilities. The Information Management System research group of the Department of Information Engineering of the University of Padua is partner of the SAPIR consortium and it is mainly concerned with (1) complex search, retrieval and ranking in distributed environments, such as P2P networks and (2) media analysis and enrichment for search. This work reports on the results that have been reached so far and that have been mapped on an initial prototype.

2 Modeling Distributed Retrieval

The representation of the uncertain nature of the retrieval process and of the routing mechanisms which characterize a distributed system is a key issue when modeling a distributed and heterogeneous information retrieval (IR) system, such as the one in the P2P network designed within SAPIR [3]. The uncertainty is basically caused by the very limited knowledge about the documents and collections which store the data relevant to the information needs represented as free multimedia queries by the end users.

A probabilistic model would naturally allow for dealing with uncertainty. In addition, the multimedia character of the data requires a general probabilistic model — an important example of a non-standard medium is music, which plays a relevant role within SAPIR and is addressed in Section 4 where a module of the prototype developed within the project is illustrated.

¹ <https://sysrun.haifa.il.ibm.com/sapir/index.html>

Uncertainty and media heterogeneity can be met if the layered approach proposed in the model reported here can be adopted. When a probabilistic model is adopted for modeling IR, three layers can be distinguished: event space, representation and description [2]. The layered approach allows for separating conceptual modeling (event space) from representation and description. This separation allows for defining different event spaces, different representations for a single event space, and different descriptions for a single representation.

The *event space* contemplates the set of the original resources of interest – in the context of SAPIR, these resources are documents, peers, ultrapeers and queries. The set of resources are then combined for defining different event spaces accordingly to the architecture of the P2P network – for example, the hybrid network architecture would comprise three levels of resources: media objects, peers and ultrapeers. Two event spaces are currently studied within SAPIR. The first event space is a set of tuples of resources at different levels; this approach allows for modeling routing mechanisms as well as the distribution of information across many peers. The second approach consists of defining independent event spaces for different resource levels; in this way an event can be represented with a higher number of degrees of freedom.

An IR system *represents* these entities because of its limited computational capabilities for understanding the content. This is the reason for introducing the *representation* and *description* layers. The representation layer is implemented as random variables on the event space of the conceptual layer, while a description level is implemented as estimators of the random variables. Representation and description may depend on the medium.

3 Architectural Approach to Distributed Retrieval

The approach to modeling multimedia distributed retrieval introduced in the previous section has been the starting point for designing a software architecture, called SPINA (Superimposed Peer Infrastructure for iNformation Access). SPINA aims at being independent of both the underlying network infrastructure and the media of the documents stored in the network; it is focused on exchanging statistics about the features extracted from the indexed documents and aggregating the resources according to the level hierarchy; it selects peers and routes query by the degree of belief that a peer or a document store relevant information. Finally, it aims at integrating the software mod-

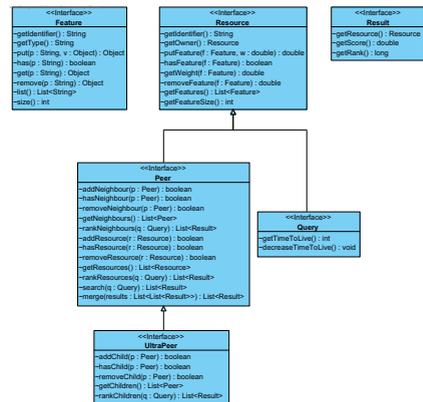


Fig. 1. SPINA conceptual architecture.

ules developed by us in SAPIR. The current status of the design of SPINA is depicted in Figure 1. In particular:

- **Resource**: represents the basic unit carrying information, i.e. generic multimedia content, and is a container for **Features** together with their weights;
- **Peer**: is a container of **Resources**. A **Peer**: is capable of ranking its own **Resources** with respect to a given **Query** and providing information about their **Features**; may have neighbour **Peers**; is capable of ranking its own neighbours with respect to a given **Query**; may be owned by another **Peer** which actually is an **UltraPeer**;
- **UltraPeer**: manages and organizes a group of **Peers** and is capable of rank them with respect to a given **Query**.

4 A Prototype System for Automatic Music Identification

The application of automatic identification of music works ranges from digital right management to automatic metadata extraction, and to music access and retrieval. A common approach is to compute an *audio fingerprint* directly from a recording in digital format [1], with the goal of providing the user with metadata about the recording – title, composer, year, genre, and so on. The prototype presented in this demo extends the concept of audio fingerprint, because the system is able to generalize the information of an audio recording and to recognize alternative versions of the same recording.

The approach is based on previous work on music identification on audio to score matching [5] and audio to audio matching [4]. As for fingerprinting, the methodology is based on a collection of labeled works, stored in a database, and tagged with relevant metadata. For each work in the database, an Hidden Markov Model (HMM) is automatically built, where each state in the HMM is labeled by a musical event of the musical work (notes, chords, rests). The transition probabilities model the fact that musical events are ordered in time, which correspond to a topology which is called *left to right* because only self-transitions – modeling the duration of an event – and transitions towards the next event are allowed. This is an important characteristics, which from quadratic becomes linear in the number of states.

Labeling states with musical events allows us to define their observation probabilities according to the characteristics of these events. In particular, the spectrum of the audio signal to be recognized has been statistically modeled considering that each musical event is related to the presence of peaks at particular frequency bands, which correspond to the first harmonics of the musical events. The approach can be exemplified considering that, for each event, a bank of bandpass filters is computed, each filter centered on the expected harmonics of the events. The observations are then computed measuring the amount of energy output by the filterbank divided by the overall energy.

Identification is carried out using standard techniques, based on the use of forward probabilities. Alternative approaches have been tested, as described

in [5], taking into account the particular paths across the HMM states, yet the use of forward probabilities is the best compromise between computational cost and identification results. Experimental results with a collection of 50 unknown recordings and a database of 200 recordings showed that 90% of the analyzed recordings were ranked among top 3 positions, and the 66% of them were correctly identified. Moreover, only 4% returned the correct match after the first 10 positions. The mean average precision for all the 50 recordings was 78.9%.

A prototype system has been designed and developed to test the architecture. A user-friendly graphical interface have been developed on the top of the identification routines. The current demonstrator, which will be installed on a laptop computer, allows us to carry out an identification task in about 0.3 seconds. After having uploaded the system with an unknown music file, the user is presented with a list of candidate recordings which are sorted in order of relevance and may be played for verifying if they have been correctly identified. The main interface is reported in Figure 2. Current research is oriented towards the integration of the standalone prototype with the distributed P2P architecture, by spreading the database among the peers, which compute a partial match with the unknown recording. Such a solution will give high scalability and parallelism to the model thus increasing considerably the performance in term of speed and robustness.

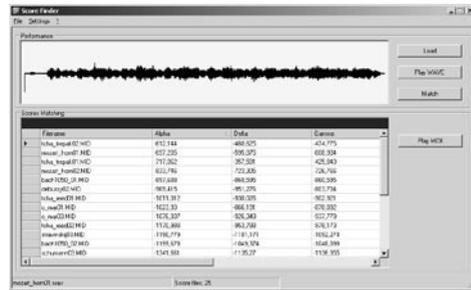


Fig. 2. The prototype system.

Acknowledgments

The work reported in this work has been partially supported by the SAPIR project, as part of the Information Society Technologies (IST) Program of the European Commission (Contract IST-045128).

References

1. P. Cano, E. Batlle, T. Kalker, and J. Haitsma. A review of audio fingerprinting. *Journal of VLSI Signal Processing*, 41:271–284, 2005.
2. N. Fuhr. Probabilistic models in information retrieval. *The Computer Journal*, 35(3):243–255, 1992.
3. M. Melucci and A. Poggiani. A study of a weighting scheme for information retrieval in hierarchical peer-to-peer networks. In *Proc. of the European Conf. on IR Research (ECIR)*, LNCS 4425, Springer, pages 136–147, 2007.
4. R. Miotto and N. Orio. Recognition of music performances through audio matching. *Italian Research Conf. on Digital Library Systems*, 2007.
5. N. Orio. Musifind: a System for the Automatic Identification of Music Works. In *Atti del XV Convegno Nazionale su Sistemi Evoluti per Basi di Dati*, 2007.

Simultaneous Previews for Time Series Forecasting

Paolo Buono

Dipartimento di Informatica, Università di Bari
via Orabona, 4 - 70125 Bari, Italy
buono@di.uniba.it

Abstract. Time series data consist of sequences of real numbers, representing the measurements or observations of a real variable at equal time intervals. Huge amounts of time series data are available today in many domains, like auctions, new stock offerings, industrial processes. People often desire estimates of the future behavior of partial time series. In this paper we present a data driven forecasting method, that we call Similarity-Based Forecasting (SBF). The forecast is displayed graphically in a forecasting preview interface, allowing users to analyze the effects of alternative pattern matching parameters.

1 Introduction

Whenever the future is uncontrollable and uncertain, people need to predict the likelihood of success when signing contracts, making investments or buying products. Among the different types of forecasting, time-series forecasting is the most common and has the largest number of applications [1]. Researchers have paid much attention to improve forecasts and their accuracy; however, little attention has been given to the visualization and user interaction. Interactive exploration of forecasts has the potential to enable understanding of interesting phenomena that are hard to attain with the existing practices and approaches. The visualization and exploration of time series has been studied extensively but challenges remain [8]. A recent innovation uses the focus+context technique [6] particularly useful for spotting (ir)regularities.

Classic statistical approaches for time series forecasting are either model-based (e.g., ARIMA models) or data-driven (e.g. exponential smoothing). The model driven approach is based on fitting a model to the time series, based on the specific domain. The data driven approach identifies patterns in the current time series and uses those to extrapolate into the future. We take a data driven approach and we compare the partial time series with historical time series in order to find similar behavior in the database. This assumes that: the partial time series behaves similarly to some historic time series; it is required the adoption of similarity algorithms; exceptional events are not taken into account; all possible events are represented in the historic database. Since a data-driven approach requires larger datasets than model driven methods, the historic dataset must include a sufficient number of records. The data driven approach is domain independent, and enables automated forecasting.

In our proposal, users conduct a pattern matching search in a dataset of historical time series, and generate a subset of curves similar to the partial time series to be

forecasted. The forecast is displayed graphically as a river plot showing statistical information about the similarity-based subset. A preview interface allows users to interactively explore the effect of the pattern matching parameters and see multiple forecasts simultaneously. This new interactive forecasting interface was built on top of TimeSearcher [3, 4], a time series visualization tool.

2 Forecasting with TimeSearcher 3

TimeSearcher is a time series visualization tool that allows interactive exploration of time series data [4, 5]. Examples of data used in TimeSearcher include weather or air quality measures, oil well production, online auctions, or stock prices. For each item (e.g. an auction) TimeSearcher displays multiple time series representing multiple variables (e.g. price, price velocity, and price acceleration). TimeSearcher can also associate each item with a set of attribute data (metadata) that remain constant over time (e.g. seller rating or auction start day).

The work presented in this paper builds on previous work done at the University of Maryland that explored the use of Timeboxes to query time series data. Timeboxes are rectangular regions that are selected and directly manipulated by the user. Boundary values of the Timeboxes are used to specify the relevant parameters of the query. The concept of Timebox was enhanced by adding the SearchBox, that is used to perform a specific pattern search anywhere in the data [4, 7]. Our target users may be expert or casual users but our interfaces do not require any specialized analysis skills such as statistical knowledge.

We exploit a database of historical time-series (auction price curves, medical records, meteorological data, etc.) and a partial time series, to perform the data-driven forecast. Users first select a partial time series to be forecasted (the source). Then they decide if the entire database should be used for forecasting, or a subset of it.

Running the similarity search produces a subset of similar items, the Similarity-Based Forecast (SBF) subset, which is displayed to the user either in the variable view or the river plot view. The median of the river plot of this SBF subset is the forecast, while the minimum, maximum, the 25th and 75th percentile indicate the statistical distribution of the SBF subset, which visually indicates the forecast uncertainty (Figure 1).

Similarity algorithms have many parameters that affect which items will be found similar and remain in the SBF subset and those parameters need to be selected. Users can select one or more variables of the dataset for the search algorithm to consider. When more than one variable is selected, they are used conjunctively. In addition, users can dynamically change the setting of the other pattern matching parameters, which are the similarity algorithm, the time interval to be considered for matching, the tolerance, and the data transformations applied [4].

The effect of those changes is reflected immediately on the display, which facilitates experimenting with choices of parameters. Observing users interacting with TimeSearcher 3 revealed that exploring all possible choices of parameters is a challenge and led to the design of a forecasting preview interface, which allows users to systematically see the results of multiple parameter variations at once. We were

mainly inspired from the multiple previews of Adobe Photoshop “Variations” interface [2] and from Side Views [9].

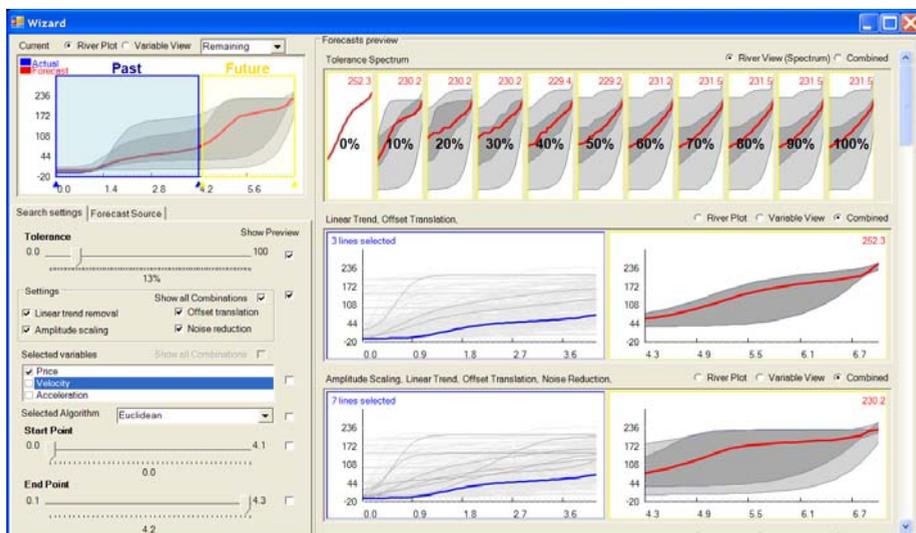


Figure 1 – The simultaneous preview interface. The top left area shows the current selection. The bottom left area contains the options the user can select. The right area is the preview area, that contains the Tolerance Spectrum Preview (on the top) and the preview of binary parameters (in this figure is shown the combined view, composed by both the variable view and the river plot)

The interface shown in Figure 1 is divided into three main areas: the option panel (bottom left), the current selection (upper left) and the preview area (right). In the option panel, users can select values and choose which parameter to vary in the previews. For each parameter there is a “Show Preview” checkbox. When clicked, the corresponding preview panel is added in the preview area. The preview can handle continuous parameters (such as the tolerance - that is the similarity factor, or the start and end point of the pattern to be matched) and sets of binary parameters (such as all possible transformations). Continuous parameters are shown with several previews taken at various steps in the full range that the parameter can assume. Binary parameters are shown with two series of panels for all possible combinations.

The combined view, displaying both the river plot and the variable view, shows users exactly on which items the forecast is based, thereby allowing them to base decisions on the visible results.

Conclusions

Similarity-Based Forecasting (SBF) is a data driven forecasting method that uses the similarity of the series to be forecasted with a set of similar historical time series. The enhancement of TimeSearcher 3 with the forecasting tool shows the applicability and

feasibility of this approach. The forecasting preview allows users to look at many combinations simultaneously. In order to validate this, we collected feedback from a total of eight users in two rounds of testing. They interacted with the preview interface without training for about 15-20 minutes using a think aloud protocol and then summarized the problems they encountered. Most of the users agreed that seeing many previews at once helped them to forecast time series more easily and accurately. Novice users had difficulties in noticing the often subtle differences between the various panels, so future work could focus on adding tools and features that assist users in judging the outcomes of the forecasts. More experienced users wanted future versions to display even more previews simultaneously and to filter the results so that the more relevant ones are more easily presented.

Further work might also support faster pattern search algorithms and could consider multiple sources for the forecast, possibly weighted.

We believe that this work provides a basis for further research on building effective user interfaces for exploratory time series analysis and forecasting.

References

- [1] Armstrong, J.S., Combining Forecasts, *Principles of Forecasting: A Handbook for Researchers and Practitioners*, J. Scott Armstrong (ed.): Norwell, MA: Kluwer Academic Publishers (2001),417-439.
- [2] Adobe Systems Inc.: <http://www.adobe.com>, last accessed (January 2007).
- [3] Aris, A., Shneiderman, B., Plaisant, C., Shmueli, G., and. Jank, W, Representing Unevenly-Spaced Time Series Data for Visualization and Interactive Exploration, Proc. of INTERACT 2005, Rome (Sept. 2005), 835-846.
- [4] Buono, P., Aris, A., Plaisant, C., Khella, A., and Shneiderman, B., Interactive Pattern Search in Time Series, Proc. of VDA'05, SPIE, Washington, DC (2005), 175-186.
- [5] Hochheiser H., and Shneiderman, B., Dynamic Query Tools for Time Series Data Sets, Timebox Widgets for Interactive Exploration, *Information Visualization 3(1)* (Mar. 2004), 1-18.
- [6] Kincaid, R. and Lam, H. (2006), Line graph explorer: scalable display of line graphs using Focus+Context. In Proc. of the Working Conference on Advanced Visual interfaces (Venezia, Italy) AVI '06. ACM Press, New York, NY (2006), 404-411.
- [7] Shmueli, G., Jank, W., Aris, A., Plaisant, C., and Shneiderman, B., Exploring auction databases through interactive visualization, *Decision Support Systems* 42, 3 (2006), 1521-1538
- [8] Silva, S. F. and Catarci, T. (2000), Visualization of Linear Time-Oriented Data: A Survey. Proc. of the 1st international Conference on Web information Systems Engineering (Wise'00) IEEE Computer Society, Washington, DC (2000), 310.
- [9] Terry, M., Mynatt E. D., Side Views: Persistent, On-Demand Previews for Open-Ended Tasks, Proc. 15th Annual ACM Symposium on User Interface Software and Technology, ACM Press (2002), 71-80.

Analysing Multidimensional Data with a Visualization Tool

Paolo Buono and Maria Francesca Costabile

Dipartimento di Informatica, Università degli Studi di Bari, Italy
{buono,costabile}@di.uniba.it

Abstract. Data Analysis is one of the main activities in a decision making process. In order to improve the users' ability to grasp information from data, information visualization techniques can be exploited. They support users in exploring large datasets by allowing them to directly interact with visual representations of data and dynamically modify parameters to see how they affect the visualized data. In this paper, we present a visualization tool that supports the analysis of multidimensional data.

1 Introduction

Information Visualization combines interaction techniques with appropriate visualizations. By allowing dynamic user control of the visual information through direct manipulation principles, it is possible to traverse large information spaces and facilitate comprehension with reduced anxiety. In a few tenths of a second, humans can recognize features in mega-pixel displays, identify patterns and exceptions, recall related images.

In this paper we illustrate DaeQP, a visualization tool that allows users to analyze multidimensional data. It is based on the Query Preview technique proposed in [4]. Its name comes from the fact that DaeQP is actually a component of a framework for data analysis, called Data Analysis Engine (DAE), developed by our research group [2]. Compared with the original proposal in [4], DaeQP provides the user the possibility of visualizing more data dimensions; it also better exploits screen real estate in a way similar to Treemap [5, 1].

2 DaeQP at work

DaeQP, has been originally developed in a context of analysis of trade fairs data [3]. The tool has also been used with data about the Computer Science students at the University of Bari. They are data about 1190 students, extracted from the University database. DaeQP is useful to help monitoring students during their university career. For example, the coordinator and her staff are interested in examining student profit at the end of each semester. Actually they are interested in analyzing many other relations among data, such as type of high school that

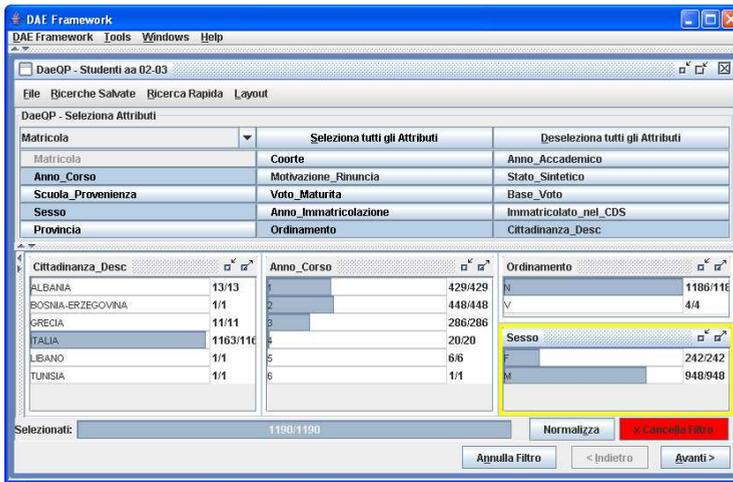


Fig. 1. Overview of data about students along *Matricola* as main attribute and *Anno_Corso*, *Sesso*, *Ordinamento* and *Cittadinanza_Desc* attributes to visualize.

the students with higher profit attended, province and/or country they come from, male student profit versus female student profit, etc.

DaeQP allows the user to retrieve information about students by initially generating an overview in which data are visualized along some major attributes. In the upper part of the screen in Fig. 1, the data attributes are shown. In this case, the attributes are fifteen, but this is a dynamic parameter depending on the dataset being visualized. Among these fifteen attributes, the user must select one main attribute and the other attributes to visualize in order to carry out the analysis. As visible in Fig. 1, the coordinator has chosen *Matricola* as the main attribute, which represents the unique identifier of the students, and four other attributes, namely *Anno_Corso*, *Sesso*, *Ordinamento* and *Cittadinanza_Desc*.

In the lower part of the screen, the overview of all students of the dataset is shown. The long bar at the bottom indicates that they are 1190 and all are displayed (1190/1190). The user immediately gets a lot of information from this overview. For example, only one student come from Tunisia and one from Lebanon (“Libano” in Fig. 1), while most students are italians and exactly 286 students attend the third course year. Close to each bar there are two numbers, the right number indicates the total number of students having that attribute value, the left number indicates how many of them are selected (the left number is always less or equal to the right one). Therefore, in the overview, in which the whole dataset is shown, the two numbers are always equal. When the user makes a selection, the left number is updated giving a “preview” of the resulting data that indicates how many items with that attribute value are in the selected dataset.

Starting from the overview in Fig. 1, the coordinator may easily select students with specific attribute values. For example, if the coordinator is interested

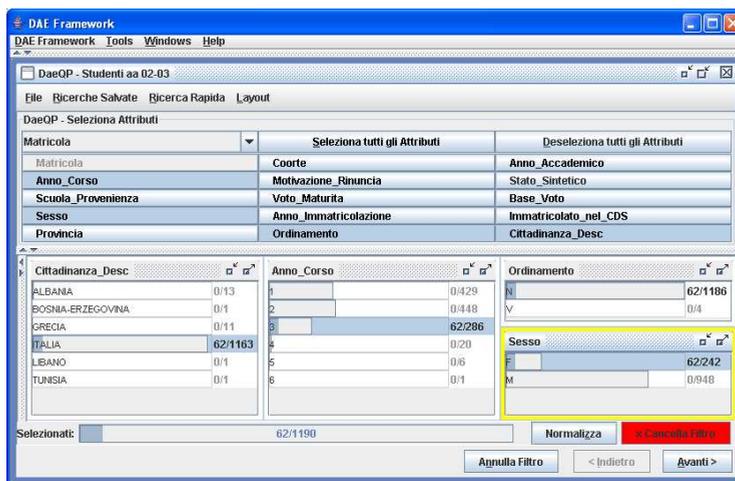


Fig. 2. Query preview after a selection on specific values of *Cittadinanza_Desc*, *Anno_Corso* and *Sesso*.

in Italian female students that are attending the third course year then (s)he clicks on either the corresponding value or the bar of each attribute. Fig. 2 shows the resulting dataset after the user has clicked on the value “3” of *Anno_Corso*, on the value “Italia” of *Cittadinanza_Desc* and on the value “F” of *Sesso*. As indicated by the bar at the bottom of Fig. 2, the selected dataset includes 62 students. All values in the showed attributes but the selected ones are grayed-out to clearly indicate the characteristics of the selected students. A click on the *Avanti* button will show the list of the student in the selected subset. The coordinator may use the set of these 62 students to send them an e-mail or for other purposes. If the coordinator is not happy with this selection, she can easily choose the values of the attributes or go back to the overview in Fig. 1 and make different selections. At any moment of the interaction session, the user may decide to display further attributes by clicking on the buttons on top of the window in Fig. 2. Similarly, to remove a displayed attribute, the user clicks on the attribute button.

A further novelty of DaeQP with respect to [4] and to its previous version described in [3] is the algorithm for displaying attributes and values, which better exploits the screen real estate. The new algorithm is based on the so-called “space-filling” technique implemented in Treemap [5]. There are several layout algorithms to dispose on the screen the various boxes that compose the treemap, called: Slice-and-dice, Pivot-by-middle, Pivot-by-size, Pivot-by-split, Strip, Cluster, Squarified [1]. The new version of DaeQP adopts a variation of the Strip algorithm, which displays the sub-windows along two lines. The sub-windows are first ordered by decreasing height. The algorithm starts displaying the sub-windows from left to right. In Fig. 2, the sub-window having the *Cittadinanza_Desc* attribute is displayed on the left, then the sub-window having

the *Anno_Corso* attribute, and so on. When the right margin of the screen is reached, the remaining sub-windows are displayed in the available space on the second line, this time going from left to right. In Fig. 2, one sub-window is on the second line, namely *Sesso*. This layout can easily show 8-10 attributes at a glance without problems. This is not a limitation since we observed that users generally do not consider more attributes at a time for the analysis.

3 Conclusions

The previous version of DaeQP, which did not implement the strip layout algorithm described in Section 2, has been used by the coordinator (and her staff) of the Computer Science curriculum of the University of Bari for the analysis of students data. We observed them working with the tool and we interviewed them. Especially staff people, without any computer science background, are very pleased of the ease of use of this tool and of the meaningful insights they provide. However, they complained when displaying more than five attributes, since the readability of the attribute values was compromised. This was the main motivation to design the new layout algorithm. The user comments suggested other improvements of the user interface.

The new version of DaeQP has been evaluated so far with four users performing the thinking aloud technique. Two users are staff people working on the described student dataset. The other two users are environmental-chemistry experts, which use DaeQP for exploring relationships between pollution sources and pollutants.

DaeQP is implemented in Java, its main use is as a desktop application. It gets the data from a query to a database, from a csv or an XML file. It can also be used as an applet or through Java Web Start.

This work is funded by the University of Bari and by EU and Puglia Region through DIPIS grant.

References

1. B. B. Bederson, B. Shneiderman, and M. Wattenberg. Ordered and quantum treemaps: Making effective use of 2d space to display hierarchies. *ACM Transactions on Graphics*, 21(4):833–854, October 2002.
2. P. Buono and M. F. Costabile. Visualizing association rules in a framework for visual data mining. In M. Hemmje, C. Niederee, and T. Risse, editors, *From Integrated Publication and Informations Systems to Virtual Information and Knowledge Environments*, volume 3379 of *LNCS*. Springer-Verlag, Berlin, 2005.
3. P. Buono, M. F. Costabile, E. Covino, and G. Pani. A visual tool for multidimensional data analysis. In *Proc. of DMS '05*, pages 333–338, Banff, Sept. 5-7 2005.
4. C. Plaisant, B. Shneiderman, K. Doan, and T. Bruns. Interface and data architecture for query preview in networked information systems. *ACM Transaction in Information Systems*, 17(3):320–341, 1999.
5. B. Shneiderman. Tree visualization with treemaps: a 2-d space-filling approach. *ACM Transactions on Graphics*, 11(1):92–99, January 1992.

DB2OWL: A Tool for Automatic Database-to-Ontology Mapping

Nadine Cullot, Raji Ghawi, and Kokou Yétongnon
Laboratoire LE2I, Université de Bourgogne, Dijon, FRANCE
{Nadine.Cullot, Raji.Ghawi, Kokou.Yetongnon}@u-bourgogne.fr

Abstract. We present a tool, called DB2OWL, to automatically generate ontologies from database schemas. The mapping process starts by detecting particular cases for conceptual elements in the database and accordingly converts database components to the corresponding ontology components. We have implemented a prototype of DB2OWL tool to create OWL ontology from relational database.

1 Introduction

In order to achieve an efficient interoperability between heterogeneous information systems, many solutions have been proposed. Particularly, ontologies play an important role in resolving semantic heterogeneity by providing a shared comprehension of a given domain of interest. An ontology formally defines different concepts of a domain and relationships between these concepts. In interoperability approaches a local ontology is used for each information. The advantage of wrapping each information source to a local ontology is to allow the development of source ontology independently of other sources or ontologies. Hence, the integration task can be simplified and the addition and removal of sources can be easily supported.

Information sources may contain different types of data structures: data may be structured as databases, semi-structured as XML documents, and/or non-structured as web pages or other type of documents. However, all of these sources must be mapped to a local ontology which will express the semantic of information sources. In this paper, we focus only on the mappings between databases and the local ontology.

We have developed a tool called DB2OWL to create ontology from a relational database. It looks for some particular cases of database tables to determine which ontology component has to be created from which database component. The created ontology is expressed in OWL-DL language¹ which is based on Description Logics. The mapping process starts by detecting some particular cases for tables in the database schema. According to these cases, each database component (table, column, constraint) is then converted to a corresponding ontology component (class, property, relation). The set of correspondences between database components and ontology components is conserved as the mapping result to be used later.

¹ <http://www.w3.org/TR/owl-features/>.

2 Database to Ontology Mappings: DB2OWL Tool

2.1 Different table cases

The mapping process used in our approach depends on particular database table cases that are taken in account during the ontology creation. These cases are illustrated using examples from database schema shown in figure 1.

Case 1. When a table T is used only to relate two other tables T₁, T₂ in a many-to-many relationship, it can be divided into two disjoint subsets of columns A₁, A₂, each participating in a referential constraint with T₁ and T₂ respectively. Therefore all T columns are foreign keys and they are primaries as well because their combination uniquely defines the rows of T. For example, the table PRESENCE is in case 1 because it relates STUDENT and SESSION tables in a many-to-many relationship.

Case 2. This case occurs when a table T is related to another table T₁ by a referential integrity constraint whose local attributes are also primary keys. In this case all the primary keys of T are foreign keys because they participate in a referential integrity constraint. For example, the table STUDENT is in case 2, because it is related to PERSON table by a foreign key which is primary key at the same time.

Case 3. This case is the default case, it occurs when none of previous cases occur.

When these different cases are detected in the database, the mapping process can use them to appropriately map database components to suitable ontology components as follows.

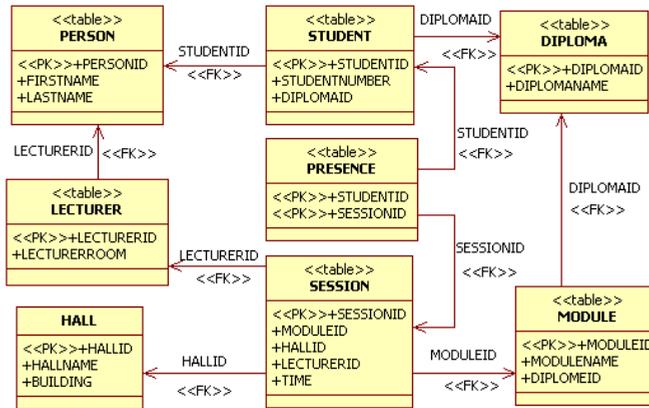


Fig. 1. The schema of schooling database

2.2 Mapping process

The mapping process is done progressively as follows. It starts by mapping the tables to concepts and then mapping the columns to properties. Thus, the table cases mentioned above are used twice: one time for table-to-class mapping and the other time for column-to-property mapping. The mapping process consists therefore of the following steps:

1. The database tables that are in case 3 are mapped to OWL classes.
2. The tables in case 2 are mapped to subclasses of those classes corresponding to their related tables, i.e. if T is in case 2 and related to T_1 by a foreign key which is primary key at the same time, then T is mapped to a subclass of the class corresponding to T_1 .
3. Each table in case 1 is not mapped to class, but the many-to-many relationship that it represents is expressed by object properties. Two object properties are added, one for each class whose corresponding table was related to the current table. In other words, when a table T is in case 1 and relates between T_1 and T_2 , and if c_1 , c_2 are the two classes corresponding to T_1 , T_2 respectively, so we assign to c_1 an object property op_1 whose range is c_2 , and assign to c_2 an object property op_2 whose range is c_1 . Each of these two properties op_1 , op_2 are inverse to the other.
4. For tables that are in case 3, we map their referential constraints to object properties whose ranges are classes corresponding to their related tables; i.e. if a table T is in case 3 and has a referential constraint with T_1 , and if c , c_1 are the classes corresponding to T , T_1 respectively, then we assign to c an object property op whose range is c_1 , and we assign to c_1 an object property op' whose range is c . To preserve the original direction of the referential constraint from T to T_1 , we set the object property op as functional. So it will have at most one value for the same instance. This characteristic is obvious because it comes from the uniqueness of key.
5. For tables that are in case 2 and have other referential constraints than the one used to create the subclass, we map them to object properties as previously.
6. Finally, for all tables we map their columns that are not foreign keys to datatype properties. The range of a datatype property is the XML schema data type² equivalent to the data type of its original column.

2.3 Mapping Generation

During the mapping process, a R2O [2] document is automatically generated to record the relationships between generated ontology components and the original database components. It includes (1) a full description of the database schema, (2) a set of concept map definitions consisting of the name of concepts with their identifying column(s), and (3) a set of relation and attribute map definitions. This document can be used to translate ontological queries into SQL queries and retrieve corresponding instances.

² <http://www.w3.org/TR/xmlschema-2/>.

3 Conclusion and Future Work

We have presented DB2OWL our tool to automatically map relational databases to OWL ontologies. This tool consider particular table cases and take them into account while the mapping process. We suppose that a user intervention may be needed later to refine the created ontology, but this still beyond the mapping process. Currently there are many approaches and tools to deal with database to ontology mapping [1][2][3][4][5][6][7][8]. Similarly to [4], [7], [8] approaches, DB2OWL is intended to create a new ontology from the databases sources, it does not allow to map a database to an already existing ontology like the rest of approaches. As in [4], DB2OWL builds an ontology in OWL language, but, unlikely, does not need any additional ontological primitives. We have implemented a prototype of this tool in Java and using Jena³. This prototype deals currently with Oracle and MySQL databases because they provide specific views about the database metadata. Extension of the presented tool are underway to deal with other DBMS that provide such views. In addition, DB2OWL will be developed further to map several databases to one ontology, and to map databases from other models such as object-relational model.

References

1. An Y., Borgida A., and Mylopoulos J. (2005): Inferring Complex Semantic Mappings Between Relational Tables and Ontologies from Simple Correspondences. In CoopIS, DOA, and ODBASE, OTM Confederated International Conferences, Cyprus, Part II, volume 3761 of LNCS, pages 1152 - 1169. Springer, 2005.
2. Barrasa J., Corcho O., Gómez-Pérez A. (2004): R2O, an Extensible and Semantically Based Database-to-Ontology Mapping Language. A. Second Workshop on Semantic Web and Databases (SWDB2004), Toronto, Canada. 2004.
3. Bizer C. (2003): D2R MAP – A Database to RDF Mapping Language, The twelfth international World Wide Web Conference, WWW2003, Budapest, Hungary.
4. de Laborda C. P. and Conrad S. (2005): Relational.OWL A Data and Schema Representation Format Based on OWL. In Second Asia-Pacific Conference on Conceptual Modelling (APCCM2005), volume 43 of CRPIT, pages 89 -96, Newcastle, Australia, 2005. ACS.
5. Konstantinou N., Spanos D., Chalas M., Solidakis E., and Mitrou N. (2006): VisAVis: An Approach to an Intermediate Layer between Ontologies and Relational Database Contents. International Workshop on Web Information Systems Modeling (WISM2006) Luxembourg.
6. Petrini J. and Risch T. (2004): Processing Queries over RDF views of Wrapped Relational Databases. In 1st International Workshop on Wrapper Techniques for Legacy Systems, WRAP 2004, Delft, Holland, 2004.
7. Volz R., Stojanovic L., Stojanovic N. (2002): Migrating data-intensive Web Sites into the Semantic Web. ACM Symposium on Applied Computing (SAC 2002). Madrid, Spain, March 2002.
8. Volz R., Handschuch S., Staab S., Studer R. (2004): OntoLiFT Demonstrator.

³ <http://jena.sourceforge.net/>.

A Tool for the Visual Synthesis and the Logical Translation of Spatio-Temporal Conceptual Schemas

Donatella Gubiani¹ and Angelo Montanari²

¹ Dipartimento di Scienze, Università degli Studi G. D'Annunzio di Chieti-Pescara

² Dipartimento di Matematica e Informatica, Università degli Studi di Udine

Abstract. In this paper we describe a software tool for the visual synthesis of ChronoGeoGraph spatio-temporal conceptual schemas and their translation into either relational schemas or XML Schema documents. We focus our attention on the management of spatial and temporal information. In particular, we describe the system supports for the specification and verification of spatio-temporal integrity constraints both at the conceptual level and at the logical one.

1 Introduction

Conceptual design is a crucial phase in the development of spatio-temporal databases and geographic information systems. In this paper we describe the distinctive characteristics of a tool for the visual synthesis of spatio-temporal conceptual schemas and their translation into either relational schemas or XML Schema documents. Conceptual schemas are based on the spatio-temporal model ChronoGeoGraph (CGG for short) that extends the Enhanced Entity-Relationship model (EER) with additional constructs for spatio-temporal information [3, 4]. In particular, it supports both the object-based and the field-based view of spatial information, it encompasses multiple temporal dimensions, and it makes it possible to describe the temporal evolution of geometrical properties of the modeled entities/phenomena.

The module for the visual synthesis of CGG schemas provides a set of simple dialog windows for the management of spatio-temporal features (similar tools have been developed for ST USM [5], MADS [6], and STER [7]). In addition, it allows the designer to automatically check a number of integrity constraints on spatio-temporal information imposed by the CGG model, e.g., to check the consistency of the geometric types of the spatial entities participating in a topological relation. The module has been implemented in Java to guarantee its portability on different platforms and it takes advantage of the JGraph library [1] for drawing, storing, updating, and visualizing the various components of the graph that diagrammatically represents a CGG schema. Two additional Java modules transform CGG conceptual schemas into logical schemas. The first one maps CGG schemas into relational schemas, while the second one maps them into XML Schema documents. The details of the system and a downloadable version of it can be found at the CGG project website [2].

2 The module for the visual synthesis of CGG schemas

The system interface is reported in Figure 1. It features a menu bar with the standard primitives for file manipulation and visualization and the management of the interface; in addition, it provides the links to the logical translation modules. Two additional tool bars allow the designer to easily access the main primitives of the menu and to select the appropriate CGG constructs to insert into the schema under development, respectively. The central panel consists of three distinct parts: an area where the conceptual schema can be drawn (up right), a tree organization of the elements of the current schema (left), and an area where the system records the set of constraints violated by the current schema (down right). An on-line assistant helps the designer in selecting the new elements to insert. CGG constructs can be added to the current schema by selecting them from the appropriate tool bar and following the instructions given by the on-line assistant. Once inserted, a construct can possibly be revised by using contextual menus or specific editors (one for every construct). Contextual menus allow one to modify some features of the selected construct, e.g., the geometry of an entity or the type of a topological relation. As an example, in Figure 2 we show how to update the temporal support of an entity (left side) and how to modify the topological relation between two given entities (right side).

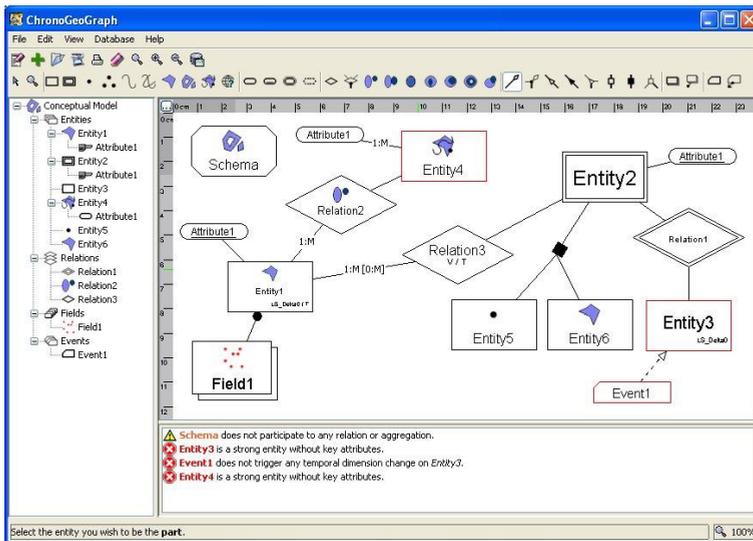


Fig. 1. The system interface.

A distinctive feature of the module is its ability to deal with integrity constraints. On the one hand, some integrity constraints are imposed by construction, e.g., the module does not allow the designer to annotate an attribute by the event time if the valid time has not been already associated with it. On the other hand, in some circumstances the module allows the user to violate some integrity constraints, e.g., it allows the user to insert an entity devoid of

a primary key or to modify the geometry of a spatial entity even though such a change makes a topological relation in which it participates (temporarily) inconsistent. However, in such cases the module puts in evidence the violations by highlighting the involved components of the schema in red color. Moreover, it provides a textual account of the violations in the area at the bottom right of the central panel which specifies the names of the involved constructs, gives a short description of the violations, and provides an indication of the seriousness of the violations (Warning or Error). By selecting the description of a specific violation, the designer can get some hints on how to possibly solve the problem.

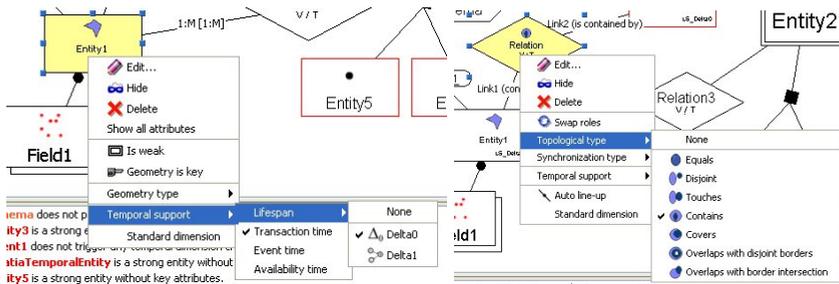


Fig. 2. Contextual menus associated with the entity and relation constructs.

3 The translation modules

Both translations are performed by suitable Java modules. As a preliminary step, the modules verify that there are not pending constraint violations in the CGG schema; then, they execute the appropriate translation steps.

From CGG schemas to relational schemas. The translation of basic EER schemas into relational ones is nowadays a routine activity. This is not the case with spatio-temporal ones. To encode spatio-temporal information, the logical model must be extended with suitable spatial and temporal dimensions. Moreover, ad hoc procedures must be added to cope with spatial and/or temporal constraints that cannot be encoded into the relational schema. As for the first requirement, most existing DBMSs, e.g., MySQL, PostgreSQL, and Oracle, are provided with a spatial extension; on the contrary, there are various proposals for the addition of one or more temporal dimensions to the relational model, but there exists no a consensus one. As for the second requirement, there are no general methodologies that drive the development of triggers for the management of spatio-temporal integrity constraints. We chose the Oracle Spatial data model because it includes both spatial objects and spatial relations as primitive concepts. Moreover, the procedural language PL/SQL can be used to encode advanced spatio-temporal constraints of CGG schemas. To capture temporal aspects, we extend the model with tuple timestamping over point and interval domains (some temporal dimensions take value over the point domain, others over the interval one). Moreover, for every CGG element provided with a transaction time dimension, we introduce a distinction between the current schema/instances

and the historical schema/instances of its relational counterpart. Unlike the case of basic EER schemas, CGG entities and relations originate more than one table. The attributes that constitute the atemporal key of an entity (resp., the union of the atemporal keys of the participating entities) defines the kernel table for the entity (resp., relation). All other entity features, e.g., other atemporal attributes, temporal (collections of) attributes, the lifespan, and the geometry, are mapped into distinct tables related to the kernel one via foreign keys. The same holds for relation features, if any. Advanced spatio-temporal constraints imposed by CGG schemas are encoded via PL/SQL triggers.

From CGG schemas to XML Schema documents. The encoding of CGG schemas into XML Schema documents takes advantage of XML Schema expressiveness, e.g., its ability to specify and check domain constraints on attributes. Spatio-temporal constraints of CGG schemas that cannot be directly dealt with in XML Schema are managed by properly annotating the generated XML Schema document (**annotation** feature) and delegating their verification to a spatio-temporal validation library integrated with the XML Schema Validator. As for spatial information, the encoding of spatial entities extends that of basic entities by adding a geometry reference to the sequence of attributes. Since XML Schema cannot impose spatial constraints on the relations between entities, spatial relations (topological, metric, direction, and aggregation relations) are dealt with as standard relations and annotated with the relation type. As for temporal information, temporal entities are provided with a special subelement, called lifespan, that can be viewed as a temporal attribute. All temporal relations, including aggregation, are dealt with by using **key** references (the inclusion method cannot be applied to temporal relations, because relation changes would force one to repeatedly and redundantly insert all tree subelements). Every temporal dimension of a relation is modeled as an XML attribute of the element that encodes the relation (the same happens to temporal attributes).

References

1. Alder, G., JGraph and JGraph Layout Pro User Manual, 2006.
2. ChronoGeoGraph. Available from <http://dbms.dimi.uniud.it/cgg/>, 2007.
3. Gubiani, D., Montanari, A., ChronoGeoGraph: an Expressive Spatio-Temporal Conceptual Model. In: *Proc. of the 15th Italian Symposium on Advanced Database Systems*, Torre Canne (Fasano, BR), Italy, 2007.
4. Gubiani, D., Montanari, A., ChronoGeoGraph: a Development Framework for Spatio-Temporal Databases. Research Report UDMI/01/07/RR, University of Udine, 2007.
5. Khatri, V., Ram, S., Snodgrass, R.T., ST USM: Bridging the Semantic Gap with a Spatio-Temporal Conceptual Model, TIMECENTER Technical Report TR-64, 2001.
6. Parent, C., Spaccapietra, S., Zimányi, E., *Conceptual Modeling for Traditional and Spatio-Temporal Applications: The MADS Approach*, Springer, 2006.
7. Tryfona, N., Jensen, C.S., *Conceptual Data Modeling for Spatiotemporal Applications*, *Geoinformatica* **3**:245–268, 1999.

Disambiguation of Structure-Based Information in the STRIDER System*

Federica Mandreoli, Riccardo Martoglia, and Enrico Ronchetti

DII, University of Modena e Reggio Emilia, via Vignolese, 905/b - I 41100 Modena
(fmandreoli, rmartoglia, eronchetti@unimo.it)

Abstract. We present the current version of STRIDER¹, a versatile system for the disambiguation of structure-based information like XML schemas, structures of XML documents and web directories. It can be of support to the semantic-awareness of a wide range of applications, thanks to its novel and fully-automated disambiguation algorithms.

1 Introduction

Knowledge based approaches are rapidly acquiring more and more importance in a wide range of application contexts, like schema matching and query rewriting [2, 5], peer data management systems (PDMS), XML data clustering and classification [8] and ontology-based annotation of web pages and query expansion [1, 3]. In these contexts, most of the proposed approaches share a common basis: They focus on the structural properties of the accessed information, which are represented adopting XML or ontology based data models, and their effectiveness is heavily dependent on knowing the right meaning of the employed terminology. Fig. 1-a shows the hierarchical representation of a portion of the web directories offered by GoogleTM. It is an example of a typical tree-like structure-based information managed in the above mentioned contexts and which our approach is successfully able to disambiguate. It contains many polysemous words, from **track** to which WordNet [6], the most used commonly available vocabulary, associates 11 meanings, to **home** (9 meanings), **intelligence** (5 meanings), and so on. The information given by the surrounding nodes allows us to state, for instance, that **track** is a “racing course” and not a “selection of music”, and **intelligence** is “a unit responsible for gathering information about an enemy” and not “the ability to comprehend”.

In this paper we demonstrate the current version of STRIDER, a system which can be of support to these kinds of approaches in overcoming the ambiguity of natural language, as it makes explicit the meanings of the words employed in tree-like structures. STRIDER builds on the novel versatile structural disambiguation approach we proposed in [4].

* A previous version of this demo has been presented at the EDBT’06 Conference. This work is partially supported by the FIRB NeP4B national project.

¹ STRucture-based Information Disambiguation ExpeRt

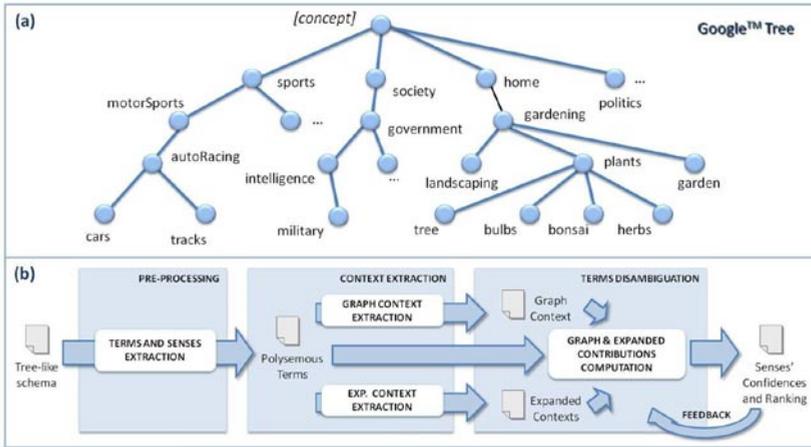


Fig. 1. (a) A part of Google web directories;(b) The complete STRIDER architecture.

2 An overview of the STRIDER System

STRIDER is designed to perform effective disambiguation of tree-like structures. As shown in Fig. 1-b, which depicts the complete architecture of our system, STRIDER takes in input structure-based information like XML schemas, structures of XML documents and web directories and disambiguates the terms contained in each node's label using WordNet as external knowledge source. The outcome of the disambiguation process is a ranking of the plausible senses for each term. In this way, the system is able to support both the completely automatic semantic annotation whenever the top sense of the ranking is selected and the assisted one through a GUI that assists the user providing useful suggestions. The STRIDER system has the following features:

- automated extraction of terms from the schema nodes (**Terms and Senses Extraction** component in Fig.1-b);
- high-quality and *fully-automated disambiguation* that: (i) is independent from training or additional data, which are not always available [7]; (ii) exploits a context which goes beyond the simple “bag of words” approach and preserves the information given by the hierarchy (*graph context*); (iii) allows flexible extraction and full exploitation of the graph context according to the application needs (**Graph Context Extraction** component in Fig.1-b); (iv) enriches the graph context by considering the *expanded context*, with additional information extracted from WordNet definitions and usage examples (**Expanded Context Extraction** component in Fig.1-b);
- *interactive and automated feedback* to increase the quality of the disambiguation results;
- user-friendly GUI speeding up the *assisted disambiguation* of schemas, providing an easy-to-use layout of the informative components.

Technical details about the implemented techniques for structural disambiguation are available in [4].

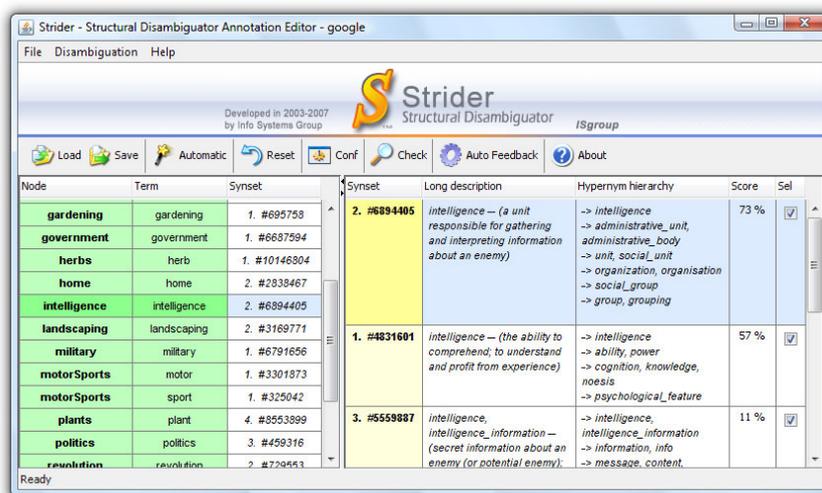


Fig. 2. The Graphical User Interface of the STRIDER System.

3 Demonstration

In this section we demonstrate the main features of STRIDER. The effectiveness of the system has been experimentally measured on several tree-like schemas differing in the level of specificity and polysemy [4] (schemas are available online at www.isgroup.unimo.it/paper/strider).

Fig. 2 shows STRIDER's GUI with the results of the disambiguation process for the Google example (Fig. 1-a). In the left part of the GUI we see columns **Node**, **Term** that show the outcome of the automated extraction of terms from the tree's nodes and column **Synset** that contains the chosen sense for the corresponding term. For flexibility purposes, the GUI allows users to fill it in either by manually choosing one of the senses in the right part or by pressing the *Magic Wand* button. This simple act triggers the fully *automatic disambiguation* process of STRIDER which is applied to the entire loaded tree and automatically chooses the top sense in the ranking of each term. When the user highlights a term in the left part of the GUI, the right part shows all the available senses and for each of them the synset's hypernym hierarchy. One of the major strengths of our system is the versatility of being able to choose the crossing setting that is best suited to the tree characteristics. For instance, when the crossing setting is made up of the whole tree, the term **bulb** of Fig.1-a is not disambiguated as “an underground stem serving as a reproductive structure”, but as “an electric lamp” due to the presence of terms like **plant** that could have the meaning of “industrial complex” rather than “vegetables living organism”. This behavior is typical of trees that gather very heterogeneous concepts like web directories. On the other hand, only by using the whole tree as the crossing setting in trees that have a very particular scope, for instance the SIGMOD Record scientific digital library, terms like **conference** and **issue** are correctly disambiguated whereas

a restricted crossing setting made of only ancestors and descendants provides wrong results. In general, the performed tests demonstrate that most of the term's senses are correctly assigned straightforwardly with the disambiguation (the mean precision level on the tested trees is generally over 80% [4]). Such good performance is obtained even when the graph context provides too little information, as in generic bibliographic schemas, thanks to the *context expansion* feature which is able to deliver a higher disambiguation precision, by expanding the context with additional related nouns contained in the description and in the examples of each sense in WordNet. To get even better results the user could choose to refine them by performing successive disambiguation runs; for this purpose he/she is able to deactivate/activate the influence of the different senses of the available context words on the disambiguation process. Further, the flexibility of our approach allows the user to benefit from a completely *automated feedback*, where the results of the first run are refined by automatically disabling the contributions of all but the top ranked X senses in the following runs.

4 Conclusions

The disambiguation performances achieved by STRIDER are encouraging and demonstrate the very good effectiveness of the adopted approach. The intuitive GUI provides easy interaction with the user. Further, the system is currently undergoing a major feature enhancement and, in order to meet the needs of the most cutting edge semantic-aware applications even better, it will soon be able to: (a) support the disambiguation of several additional input formats, such as complete relational schemas and non tree-like ontologies; (b) exploit additional disambiguation techniques offering integration with a larger number of external knowledge sources, including on-line search engines and thesauri.

References

1. P. Cimiano, S. Handschuh, and S. Staab. Towards the self-annotating web. In *Proc. of the 13th WWW Conference*, 2004.
2. H. Do, S. Melnik, and E. Rahm. Comparison of schema matching evaluations. In *Proc. of the 2nd WebDB Workshop*, 2002.
3. Marc Ehrig and Alexander Maedche. Ontology-focused crawling of web documents. In *Proc. of the ACM SAC*, 2003.
4. F. Mandreoli, R. Martoglia, and E. Ronchetti. Versatile Structural Disambiguation for Semantic-aware Applications. In *Proc. of the 14th CIKM Conference*, 2005.
5. F. Mandreoli, R. Martoglia, and P. Tiberio. Approximate Query Answering for a Heterogeneous XML Document Base. In *Proc. of the 5th WISE Conference*, 2004.
6. G. A. Miller. WordNet: A Lexical Database for English. *CACM*, 38(11), 1995.
7. I. Tatarinov and A. Halevy. Efficient Query Reformulation in Peer Data Management Systems. In *Proc. of ACM SIGMOD*, 2004.
8. M. Theobald, R. Schenkel, and G. Weikum. Exploiting Structure, Annotation, and Ontological Knowledge for Automatic Classification of XML Data. In *Proc. of the WebDB Workshop*, 2003.

Ontology-based Data Access with MASTRO

Antonella Poggi and Marco Ruzzi

Dipartimento di Informatica e Sistemistica,
Università di Roma “La Sapienza”
{poggi, ruzzi}@dis.uniroma1.it

1 Introduction

In several areas, such as Enterprise Application Integration, Data Integration [5], and the Semantic Web [4], clients need to access a shared conceptualization of the intensional level of the application domain in order to specify the access to services exported by the system. Ontologies are nowadays considered as ideal tools providing such a conceptualization. Indeed, one of their most interesting usages is *ontology-based data access*, where, on top of the usual data layer of an information system, a conceptual layer is superimposed, allowing the client to abstract away from how the information is actually maintained in the data layer. On the other hand, relational DBMSs are natural candidates for the management of the data layer. The combination of these two mechanisms for representing information constitutes a significant advance in the use of databases. Indeed, it requires on the one hand to deal with the characteristics of both formalisms, and on the other hand to address properly their interaction in a combined system.

In this paper, we present MASTRO, an ontology-based data access system, that addresses the above mentioned issues. More precisely, the main MASTRO distinguishing features can be summarized as follows:

1. It exhibits to the client a conceptual view that is expressed by means of a new Description Logic of the *DL-Lite* family [3], called *DL-Lite_A*. This turns out to be an “optimal” trade-off between language expressivity and efficiency of access tasks through the ontology. Indeed, it is worth noting that an ontology is not a database, instead it is an abstraction of a possibly infinite set of databases, that correspond to the ontology models. Thus, an ontology can be rather seen as a database with incomplete information. Clearly, this has an impact on all data access tasks. In particular, query answering over an ontology differs from standard query evaluation. Rather, it is equivalent to compute the so-called *certain answers*, which, from the point of view of logic, is the logical implication problem checking whether it logically follows from the ontology specification that a certain tuple satisfies the query.
2. It is the first system we are aware of, which provides advanced forms of reasoning, including conjunctive query answering, that are LOGSPACE in the size of the underlying data.
3. It allows to access through the ontology data managed by an autonomous relational DBMS. Such a feature has the notable advantage of allowing to access potentially huge amount of data, that exists *a priori*, and is managed by the most commonly used and efficient database technology available nowadays.

4. It addresses the mismatch between the way in which data is (and can be) represented in a relational database, and the way in which the corresponding information is rendered through an ontology over the database. Specifically, while the database of a data source stores data, instances of concepts in an ontology are objects, each one denoted by an object identifier, and not to be confused with a data value. Such a problem is known as *impedance mismatch*.

This demonstration proposal is organised as follows. Section 2 briefly introduces the MASTRO data access framework and its services. Then, Section 3 illustrates MASTRO through the university data scenario.

2 Overview of the MASTRO system

In this section, we introduce the main services provided by MASTRO, namely (i) ontology specification, (ii) query answering, (iii) ontology satisfiability, and (iv) meta-level query answering.

Ontology specification MASTRO allows to define an ontology in terms of both an intensional and an extensional level. The former is specified by means of *DL-Lite_A*¹. Thus, the domain of interest is represented in terms of (i) concepts, denoting set of objects, (ii) roles, denoting binary relations between objects, and, notably, (iii) domain values, denoting set of values. Interestingly, values allow both concepts and roles to be qualified by attributes. Concerning the ontology extensional level, MASTRO allows for *mappings* specification, that establish how data retrieved from an existing database are related to extensions of terms used in the ontology intensional level. This is achieved by (i) allowing *object terms* to denote constants, that are built by applying Skolem functors to data values, (ii) defining mappings of the form $\Psi \rightsquigarrow \varphi$, where Ψ is an arbitrary SQL query over the database, and φ is a conjunctive query over the ontology intensional level without existential variables, whose atoms may contain *variable object terms*, i.e. terms obtained by applying Skolem functors to variables denoting values.

Query answering The MASTRO query answering service is similar to query answering provided by QuOnto [1]. In particular, MASTRO is able to answer unions of conjunctive queries (UCQs) expressed over the ontology alphabet: the class of UCQs is one of the most important classes of query arising in practical cases. The query answering process is performed through *query rewriting* and strongly separates the intensional level from the extensional one: user queries are first reformulated on the basis of the ontology intensional knowledge, and then evaluated directly over the database by means of the mappings. MASTRO also provides two different mapping handling techniques. The first one exploits the SQL engine of the DBMS managing the data layer of the ontology: views are defined for concepts and roles, using the SQL query specified into the mapping assertions. The second one unfolds the query by producing an SQL statement that can be directly issued over the source tables.

Ontology satisfiability Since the ontology extensional level is specified by linking the ontology to data in general autonomous, it is very likely that the problem arises of mutually inconsistent data. It is therefore necessary to check whether there exists a model of the ontology. MASTRO provides ontology satisfiability by reducing it to a particular

¹ For details about *DL-Lite_A* constructs and assertions, we refer the reader to [2, 6].

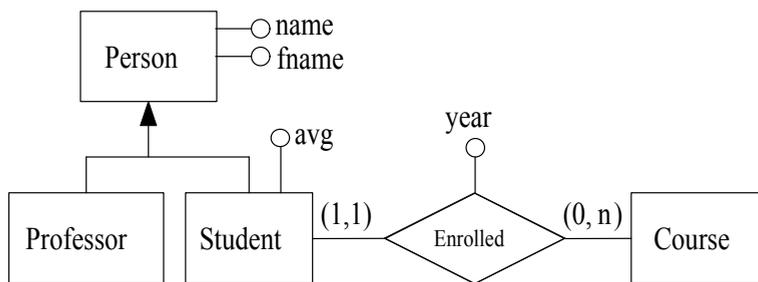


Fig. 1. An E-R representation of the sample ontology

form of query answering.

Meta-level query answering MASTRO maintains the ontology meta-level information into a local data structure called *Meta-data Repository* (MDR). Such a meta-level is itself represented by means of a $DL\text{-}Lite_A$ ontology. This enables the MASTRO system to provide some meta-level reasoning services. In particular, a query answering service is provided for answering queries expressed over the ontology meta-level.

3 Accessing university data with MASTRO

To illustrate the main features of MASTRO and test services provided by the system, we defined an example ontology, modeling information about a generic university domain (professors, students and courses attended by them) and then mapped such ontology over a real large database instance of the University of Rome “La Sapienza”. Due to the lack of space, we omit the formal specification of the intensional level of such an ontology, which can be expressed by means of the E-R diagram depicted in Figure 1. Concerning the extensional level, the database stores information about students, master degrees, professors, courses, exams and students’ administrative and didactic information, for a whole number of 27 different tables, with an overall size of 200.000 tuples. Several mappings have been defined in order to specify how objects populating the ontology can be constructed starting from the values stored within the underlying database. Mappings definitions for the *student* concept and for the *enrolled* role follow.

$$M_1 = \text{SELECT code FROM student} \rightsquigarrow \text{student}(\text{st}(\text{code}))$$

$$M_2 = \text{SELECT s, c FROM career} \rightsquigarrow \text{enrolled}(\text{st}(\text{s}), \text{cr}(\text{c}))$$

As an example, we now show how to map entity attributes to database values by means of the mappings:

$$M_3 = \text{SELECT s, AVG(gr) FROM exam_record GROUP BY s} \rightsquigarrow \text{avg}(\text{st}(\text{s}), \text{AVG}(\text{gr}))$$

In the mappings above, the *st* function symbol allows for the creation of object identifying students, starting from database values. We evaluated several queries of practical interest: the results clearly show that MASTRO performs well also with very large underlying databases. As a simple example, let us consider the query

$$q(X) \leftarrow \text{student}(X).$$

issued over the ontology defined above, asking for all the students. First the query is *reformulated* [2, 3] according to the ontology intensional-level knowledge, by producing the UCQ below:

$$q(X) \leftarrow student(X). \vee q(X) \leftarrow enrolled(X, Y).$$

We now consider the case in which the mappings are handled by exploiting the SQL engine of the DBMS managing the data layer of the ontology. In this case, a view is defined for each atomic concept, using the SQL query specified into the mapping assertion, and then user queries are processed in terms of such views. The resulting UCQ is then translated in SQL:

(SELECT s FROM student.v) UNION (SELECT e.stud FROM enrollment.v)

In the SQL query above, *student.v* and *enrollment.v* are views defined according to the mappings M_1 and M_2 . The evaluation of such SQL query over the underlying database retrieves the set of objects representing students of our ontology. Notice that there are no limitations in the SQL syntax that can be used for mappings definition. For brevity, we omit here the illustration of the alternative mapping handling technique resorting on unfolding.

We now show an working example of the satisfiability service. Consider again the ontology defined above. In order to check ontology satisfiability, MASTRO checks the effective disjointness between *student* and *professor*, by evaluating over the underlying database the SQL query below:

(SELECT s FROM student.v) UNION (SELECT p FROM professor.v)

where *professor.v* is the view obtained according to the mapping from the *professor* concept. If such a query returns a non empty result set, then the ontology is unsatisfiable. An analogous technique is used to check whether some functionality assertions are not satisfied.

Finally, we consider here concept subsumption as an example of meta-level query answering. Referring to the example ontology described above, an end-user may want to ask MASTRO whether the domain of the attribute **name** subsumes the concept *student*. In this specific case MASTRO will return a positive answer. This follows from the fact that every *student* is a *person* and every *person* has a **name** attribute. Hence, every *student* has a **name**.

References

1. A. Acciarri, D. Calvanese, G. De Giacomo, D. Lembo, M. Lenzerini, M. Palmieri, and R. Rosati. QUONTO: Querying ONTOlogies. In *Proc. of AAAI 2005*, pages 1670–1671, 2005.
2. D. Calvanese, G. De Giacomo, D. Lembo, M. Lenzerini, A. Poggi, and R. Rosati. Linking data to ontologies: The description logic DL-Lite_A. In *Proc. of OWLED 2006*, 2006.
3. D. Calvanese, G. De Giacomo, D. Lembo, M. Lenzerini, and R. Rosati. DL-Lite: Tractable description logics for ontologies. In *Proc. of AAAI 2005*, pages 602–607, 2005.
4. J. Heflin and J. Hendler. A portrait of the Semantic Web in action. *IEEE Intelligent Systems*, 16(2):54–59, 2001.
5. M. Lenzerini. Data integration: A theoretical perspective. In *Proc. of PODS 2002*, pages 233–246, 2002.
6. A. Poggi. *Structured and Semi-Structured Data Integration*. PhD thesis, Dip. di Inf. e Sist., Univ. di Roma “La Sapienza”, 2006.

Author Index

- Aberer, Karl, 2
Abul, Osman, 233
Agosti, Maristella, 479
Aloia, Nicola, 4
Amato, Giuseppe, 242
Appice, Annalisa, 16, 471
Artale, Alessandro, 250
Atzori, Maurizio, 233
- Baronti, Paolo, 242
Bartolini, Ilaria, 258
Basile, Teresa M.A., 40, 52
Berardi, Margherita, 184
Bergamaschi, Sonia, 266
Berlingiero, Michele, 274
Bettini, Claudio, 447
Bianchini, Devis, 28
Biba, Marenglen, 40, 52
Bonchi, Francesco, 233, 274
Brambilla, Marco, 282
Bruno, Giulia, 290
Buccafurri, Francesco, 64, 298
Buono, Paolo, 483, 487
- Cabot, Jordi, 282
Cali, Andrea, 308, 316
Calvanese, Diego, 250, 324
Carbotta, Domenico, 316
Caruso, Costantina, 332
Castano, Silvana, 341
Cellini, Jessica, 76
Chessa, Stefano, 242
Ciaccia, Paolo, 258, 349
Colazzo, Dario, 357
Comai, Sara, 282
Concordia, Cesare, 4
Console, Luca, 298
Costa, Gianni, 88
Costabile, Maria Francesca, 487
Cullot, Nadine, 491
Cuzzocrea, Alfredo, 365
- D'Alessandro, Saverio, 184
d'Amato, Claudia, 124, 373
Džeroski, Saso, 16
- De Antonellis, Valeria, 28
De Giacomo, Giuseppe, 324
De Meo, Pasquale, 298
De Virgilio, Roberto, 100
Di Buccio, Emanuele, 479
Di Giulio, Domenico, 112
Di Mauro, Nicola, 40, 52
Di Noia, Tommaso, 463
Di Nunzio, Giorgio Maria, 479
Di Sciascio, Eugenio, 463
Diamantini, Claudia, 76, 148
Domenico, Lembo, 324
Donini, Francesco M., 463
Dragone, Luigi, 381
- Esposito, Floriana, 40, 52, 124
- Fanizzi, Nicola, 124
Fassetti, Fabio, 389
Ferilli, Stefano, 40, 52
Fernandes Silva, Sonia, 136
Ferrara, Alfio, 341
Ferro, Nicola, 479
Fittipaldi, Antonio, 471
Folino, Francesco, 88
Fugini, Maria Grazia, 298
- Garza, Paolo, 290
Gemelli, Alberto, 148
Ghawi, Raji, 491
Giannotti, Fosca, 233, 274
Goy, Anna, 298
Greco, Gianluigi, 389, 397
Greco, Sergio, 172, 405
Gubiani, Donatella, 160, 495
Guerra, Francesco, 266
Gullo, Francesco, 172
Gunopulos, Dimitrios, 430
Guzzo, Antonella, 397
- Kalogeraki, Vana, 430
Kifer, Michael, 3, 308
Kontchakov, Roman, 250
- Lanza, Antonietta, 471
Lax, Gianluca, 64, 298

Lenzerini, Maurizio, 324
 Leo, Pietro, 184
 Loglisci, Corrado, 184
 Lops, Pasquale, 298

Manco, Giuseppe, 88
 Mandreoli, Federica, 414, 499
 Martinenghi, Davide, 316
 Martoglia, Riccardo, 499
 May, Michael, 1
 Mecca, Giansalvatore, 422
 Meghini, Carlo, 4
 Melchiori, Michele, 28
 Melucci, Massimo, 479
 Menicori, Paolo, 136
 Miotto, Riccardo, 479
 Modafferi, Stefano, 298
 Molinaro, Cristian, 405
 Montanari, Angelo, 160, 495
 Montanelli, Stefano, 341

Orio, Nicola, 196, 479
 Orlando, Salvatore, 208
 Orsini, Mirko, 266
 Orsini, Renzo, 208
 Ortale, Riccardo, 88

Pagliariacci, Francesco, 220
 Palpanas, Themis, 430
 Papotti, Paolo, 439
 Pappalardo, Alessandro, 422
 Pareschi, Linda, 447
 Parisi, Francesco, 405
 Penta, Antonio, 455
 Penzo, Wilma, 414
 Perdichizzi, Antonio M., 414
 Pernici, Barbara, 298
 Picardi, Claudia, 298
 Picariello, Antonio, 455
 Pistore, Marco, 220

Poggi, Antonella, 324, 503
 Ponti, Giovanni, 172
 Pontieri, Luigi, 397
 Potena, Domenico, 76, 148

Quintarelli, Elisa, 290

Raffaetà, Alessandra, 208
 Ragone, Azzurra, 463
 Raunich, Salvatore, 422
 Redavid, Domenico, 298
 Roncato, Alessandro, 208
 Ronchetti, Enrico, 499
 Rosati, Riccardo, 324
 Rossato, Rosalba, 290
 Russo, Vincenzo, 365
 Ruzzi, Marco, 503

Saccà, Domenico, 365
 Salvi, Denise, 28
 Sartiani, Carlo, 357
 Sartori, Claudio, 266
 Semeraro, Giovanni, 298
 Serafino, Paolo, 365
 Silvestri, Claudio, 208
 Spalazzi, Luca, 220
 Straccia, Umberto, 463

Tagarelli, Andrea, 172
 Tanca, Letizia, 455
 Terracina, Giorgio, 389
 Torlone, Riccardo, 100, 439
 Traverso, Paolo, 220

Ursino, Domenico, 298

Varlaro, Antonio, 471

Yétongnon, Kokou, 491

Zakharyashev, Michael, 250

Organizing institution



Dipartimento di Informatica - Università di Bari

Sponsoring institutions



Regione Puglia



ISBN 978-88-902981-0-3



9 788890 298103 >