Laurea Specialistica in Informatica
a.a. 2005-2006

Interazione Uomo-Macchina II:

Interfacce Intelligenti

Fiorella de Rosis

Programma del Corso

Introduzione

Prima parte: *Formalizzazione e Ragionamento*

1.1. Ragionamento logico:
- Formalizzazione
- ***Risoluzione***

1.2. Ragionamento incerto
- Reti Causali Probabilistiche
- Reti dinamiche
- Apprendimento di Reti

Seconda parte: *Modelli di Utente*

2.1. Modelli logici

2.2. Modelli con incertezza

Terza parte: *Interazione in linguaggio naturale*

3.1. Generazione di messaggi
- Introduzione
- Teorie
- Metodi

3.2. Comprensione di messaggi

Quarta parte: *Simulazione di dialoghi*

Ho bisogno di un metodo per derivare nuovi fatti da fatti noti, nellabase di conoscenza (sull'utente, sul dominio, sulle azioni da compiere).

Ad esempio:

- da Age(U,24) derivo Young(U)
- da Sport(s) $\rightarrow$ Implies(s, GoodHealth)
  e Sport(AIKIDO)
  devo derivare Implies(AIKIDO, GoodHealth)

… o cose più complesse.

Sceglieremo uno dei possibili sistemi logici: quello basato sul *principio di risoluzione*, che non elabora formule, ma *clausole.*

Espressioni logiche in forma di clausole

Letterale: una formula atomica, eventualmente negata:

Esempi: P, $\neg$Q, $\neg$ R(x), F(x).

P, F(y) sono letterali 'positivi'; $\neg$Q, $\neg$ R(x) sono letterali 'negativi'

Clausola: una disgiunzione di letterali, rappresentata con la notazione degli insiemi:

Esempi: $\{P, \neg Q\}$ è una clausola che sta per $P \vee \neg Q$;

$\{\neg R(x), F(x)\}$ è una clausola che sta per $\{\neg R(x), F(x)\}$,

e cioè per $R(x) \rightarrow F(x)$

Una *clausola di Horn* ha al massimo un letterale positivo

Liste di clausole

Una lista di clausole è intesa come *congiunzione* delle clausole in essa comprese.

Esempio: la lista

$\{\neg R(x), F(x)\}$

$\{R(A)\}$

equivale a: $\{\neg R(x), F(x)\} \wedge \{R(A)\}$

In sostanza, quindi, una lista di clausole corrisponde ad una formula in *forma normale congiuntiva.*

Nell'esempio precedente:

$(R(x) \rightarrow F(x)) \wedge R(A)$

Conversione da formule a clausole (1)

Procedure Convert (x):

```
1    Begin    x <- Implications_out (x)
2             x <- Negations_in (x)
3             x <- Standardize_variables (x)
4             x <- Existentials_out (x)
5             x <- Universals_out (x)
6             x <- Disjunctions_in (x)
7             x <- Operators_out (x)
8             x <- Rename_variables (x)
```

Genesereth & Nilsson, 1986

Conversione da formule a clausole (2)

Esempi:

Implications_out (x): $R(x) \rightarrow F(x)$ diventerebbe:

$\neg R(x) \vee F(x)$

Negations_in (x): $\neg(R(x) \vee F(x))$ diventerebbe

$\neg R(x) \wedge \neg F(x)$,

$\neg \forall x\, \varphi$ diventerebbe $\exists x \neg \varphi$,

$\neg \exists x\, \varphi$ diventerebbe $\forall x \neg \varphi$, …

Conversione da formule a clausole (3)

Esempi:

Existentials_out (x):

se l' $\exists$ non occorre nell'ambito di un $\forall$, può essere eliminato, sostituendo tutte le occorrenze della sua variabile con una costante (*costante di Skolem*):

$\exists x\, R(x)$ diventa $R(Sk)$

se l' $\exists$ occorre nell'ambito di un $\forall$, può essere eliminato, sostituendo tutte le occorrenze della sua variabile con un termine che corrisponde ad una funzione applicata alle altre variabili nel campo dell' $\forall$ (*funzione di Skolem*):

$\forall x\, \exists y\, P(x,y)$ diventa $\forall x\, P(x,Sk°(x))$

Conversione da formule a clausole (4)

Universals_out (x): tutti i $\forall$ vengono eliminati
$\forall x\ P(x,F(x))$ diventa $P(x,F(x))$

Disjunctions_in (x); la formula viene trasformata in forma normale congiuntiva
$(P \wedge Q) \vee S$ diventa $(P \vee S) \wedge (Q \vee S)$
$(P \wedge Q) \vee R \vee S$ diventa $(P \vee R \vee S) \wedge (Q \vee R \vee S)$, …

Operators_out (x); le formule ottenute vengono trascritte in forma di clausole
$(P \vee R \vee S) \wedge (Q \vee R \vee S)$ diventa $\{P, R, S\}; \{Q, R, S\}$

Rename_variables (x): nessuna variabile deve apparire in più di una clausola

Esempi di trasformazione di formule in clausole

$\exists z\ \forall x\ \forall y\ (P(x,y) \rightarrow R(z))$ implication out
$\exists z\ \forall x\ \forall y\ (\neg P(x,y) \vee R(z))$ existential out
$\forall x\ \forall y\ (\neg P(x,y) \vee R(A))$ universal out
$(\neg P(x,y) \vee R(A))$ operator out
$\{\neg P(x,y), R(A)\}$

$\forall x\ \forall y\ \exists z\ (P(x,y,z) \wedge R(x,y,z))$ existential out
$\forall x\ \forall y\ (P(x,y,F°(x,y)) \wedge R(x,y,F°(x,y))$ universal out
$(P(x,y,F°(x,y)) \wedge R(x,y,F°(x,y))$ operator out
$\{P(x,y,F°(x,y)\}$
$\{R(x,y,F°(x,y)\}$

Esercizio sulla trasformazione di formule in clausole

$\forall x\ (C(x) \rightarrow B(x)) \rightarrow \forall y\ (A(y) \rightarrow D(y))$

$(\neg \forall x\ (C(x) \rightarrow B(x))) \vee (\forall y\ (A(y) \rightarrow D(y))$
$(\exists x \neg\ (C(x) \rightarrow B(x))) \vee (\forall y\ (A(y) \rightarrow D(y))$
$(\exists x \neg\ (\neg\ C(x) \vee B(x))) \vee (\forall y\ (\neg\ A(y) \vee D(y))$
$(C(Sk) \wedge \neg\ B(Sk)) \vee (\neg\ A(y) \vee D(y))$

$\{C(Sk), \neg\ A(y), D(y)\}$
$\{\ \neg B(Sk), \neg\ A(y), D(y)\ \}$

$\forall x\ (Bird(x) \wedge \neg Ostrich(x)) \rightarrow Flies(x))$
$\forall y\ (Mammal(y) \wedge \neg Whale(y)) \rightarrow OnGround(y)$
$Ostrich(P)$
$Whale(B)$
$\forall x\ Ostrich(x) \rightarrow Bird(x)$
$\forall y\ Whale(y) \rightarrow Mammal(y)$
$Flies(z)?$

Riprendiamo un esempio già visto

In clausole:
$\{\neg Bird(x), Ostrich(x), Flies(x)\}$
$\{\neg Mammal(y), Whale(y), OnGround(y)\}$
$\{\neg Ostrich(x), Bird(x)\}$
$\{\neg Whale(y), Mammal(y)\}$
$\{Ostrich(P)\}$
$\{Whale(B)\}$
$Flies(z)?$

Unificazione

E' il processo mediante il quale si determina se due espressioni (formule o letterali) possono essere rese identiche mediante un'opportuna sostituzione di valori per le variabili in esse contenute.

Un esempio di *sostituzione*: {x/A, y/B, F°(A,B)/C}
Questa sostituzione rende identiche ('unifica')le due formule:
(P(x,y,F°(x,y)) e P(A,B,C)

Principio di Risoluzione

1. Formulazione semplice (nel calcolo degli enunciati)

Date: una clausola Φ che contiene il letterale φ e
una clausola Ψ che contiene il letterale $\neg\varphi$
Si può dedurre una nuova clausola: $(\Phi - \{\varphi\}) \cup (\Psi - \{\neg\varphi\})$

Esempio:
1 $\{\neg P, Q\}$
2 $\{\neg Q, R\}$

3 $\{\neg P, R\}$ (1,2) dove $\varphi = Q$

Esercizi sul Principio di Risoluzione

1 $\{\neg P, Q\}$
2 $\{\neg Q\}$

3 ?

1 $\{\neg P, Q\}$
2 $\{Q, R\}$

3 ?

1 $\{\neg P, Q\}$
2 $\{P, \neg Q\}$

3 ?

1 $\{\neg P, Q\}$
2 $\{\neg Q, R\}$
3 $\{\neg R, S\}$

4 ?

Modus Ponens

Notiamo che il modus ponens è un caso particolare del principio di risoluzione:

$P \rightarrow Q$	$\{\neg P, Q\}$
P	$\{P\}$
Q	Q

Principio di Risoluzione

2. Formulazione più complessa (nel calcolo dei predicati)

Date: una clausola Φ che contiene il letterale φ e
una clausola Ψ che contiene il letterale $\neg\psi$
se φ e ψ sono *unificabili* con una sostituzione λ
e Φ', Ψ' sono le clausole ottenute applicando λ
(rispettivamente) a Φ e Ψ,
allora
si può dedurre, da Φ e Ψ, una nuova clausola:
$((\Phi' - \{\varphi\}) \cup (\Psi' - \{\neg\psi\}))\ \lambda$

Esempio:

1 $\{\neg P(x), Q(x)\}$
2 $\{\neg Q(A), R(B)\}$

3 $\{\neg P(A), R(B)\}$ (1,2) con $\lambda = \{x/A\}$

Esercizi sul Principio di Risoluzione

1 $\{\neg P(x), Q(y)\}$
2 $\{\neg Q(A)\}$

3 ?

1 $\{\neg P(A), Q(A)\}$
2 $\{P(x), \neg Q(y)\}$

3 ?

1 $\{\neg P(x), Q(y)\}$
2 $\{Q(A), R(y)\}$

3 ?

1 $\{\neg P(A), Q(A)\}$
2 $\{\neg Q(B), R(B)\}$
3 $\{\neg R(C), S(C)\}$

4 ?

Strategie di Risoluzione: risoluzione 'grezza'

Input: un set Δ di n clausole C(k), con k = 1,…, n

1 i = 1; j = 2;
2 Prova a risolvere C(i) con C(j): se possibile, ottieni una clausola che aggiungi a Δ; poni n=n+1;
altrimenti: i = i+1 e ripeti la 2 finché i=j-1;
3 i = 1; j = j+1; ripeti da 2 *finché la condizione di 'stop' è verificata oppure j=n*.

Nota: la condizione di stop varia a seconda degli obiettivi per cui la procedura di risoluzione è utilizzata.

Condizioni di stop

Il significato della 'clausola vuota' $\{\}$

Se, da un insieme di clausole Δ, si ottiene, per risoluzione, la clausola vuota,
questo indica che l'insieme *Δ contiene una contraddizione*, cioè che è inconsistente.
Infatti: $A \wedge \neg A$ si rappresenta, in forma di clausole, come

1 $\{A\}$
2 $\{\neg A\}$ e permette di dedurre:

3 $\{\}$

Esercizio sulla Risoluzione 'Grezza'

1	{P}	Δ
2	{¬P,Q}	Δ
3	{¬Q,R}	Δ
4	{¬R}	Δ
5	{Q}	(1,2)
6	{R}	(3,4)
7	{}	(5,6)

… dunque, l'insieme Δ è inconsistente!

Infatti: se P è vero, P →Q, Q → R, anche R deve essere vero!

Possibili Usi Della Risoluzione

1.Dimostrazione di inconsistenza

Se da un insieme Δ si ottiene, per risoluzione,
la clausola vuota,
allora Δ è inconsistente.

Possibili Usi Della Risoluzione

2.Risposta a domande T/F

Dato un insieme Δ e una formula ω,
si può rispondere al quesito:
"Δ implica ω?"
verificando se esiste una contraddizione fra Δ e ¬ω,
cioè se l'insieme di clausole {Δ ∪ ω} è consistente.

Per far ciò:
si aggiunge ¬ω all'insieme Δ;
si risolve;
se si trova la clausola vuota, si può concludere con una risposta positiva (Δ implica ω),
altrimenti il risultato resta ignoto (per il Teorema di Completezza)

Esempio di Quesiti T/F

I padri sono genitori; anche le madri sono genitori;
Pietro è il padre di Fiorella.
Quesito: Pietro è genitore di Fiorella?

1	{¬Father(x,y), Parent(x,y)}	Δ
2	{¬Mather(x,y), Parent(x,y)}	Δ
3	{Father(P,F)}	Δ
4	{¬Parent(P,F)}	ω
5	{Parent(P,F)}	(1,3)
6	{}	(4,5)

Possibili Usi Della Risoluzione

3. Quesiti del tipo 'Fill-in the blank'

Dato un insieme Δ e una formula ω che contengono la variabile x,
si può rispondere al quesito:
"Per quale valore della variabile x Δ implica ω?"
verificando qual è (se esiste) il valore di x
che produce una contraddizione fra Δ e $\neg\omega$,
cioè che rende soddisfacibile l'insieme di clausole $\{\Delta \cup \omega\}$.

Per far ciò:
si aggiunge all'insieme Δ la clausola $\{\neg\omega, Ans(x)\}$;
si risolve, ponendo come condizione di stop la deduzione di una formula con il solo letterale 'Ans';
se si deduce la clausola $\{Ans(A)\}$, si può concludere che
"Δ implica ω per x/A"
altrimenti il risultato resta ignoto (per il Teorema di Completezza)

Esempio di Quesiti 'fill-in the blank'

I padri sono genitori; anche le madri sono genitori;
Pietro è il padre di Fiorella.
Quesito: Chi è genitore di Fiorella?

1	$\{\neg Father(x,y), Parent(x,y)\}$	Δ
2	$\{\neg Mather(x,y), Parent(x,y)\}$	Δ
3	$\{Father(P,F)\}$	Δ
4	$\{\neg Parent(x,F), Ans(x)\}$	ω
5	$\{Parent(P,F)\}$	(1,3)
6	$\{Ans(P)\}$	(4,5)

Le Risposte possono essere più di una!!

I padri sono genitori; anche le madri sono genitori;
Pietro è il padre di Fiorella; Anna è sua madre.
Chi è genitore di Fiorella?

1	$\{\neg Father(x,y), Parent(x,y)\}$	Δ
2	$\{\neg Mather(x,y), Parent(x,y)\}$	Δ
3	$\{Father(P,F)\}$	Δ
4	$\{Mather(A,F)\}$	Δ
5	$\{\neg Parent(x,F), Ans(x)\}$	ω
6	$\{Parent(P,F)\}$	(1,3)
7	$\{Parent(A,F)\}$	(2,4)
8	$\{Ans(P)\}$	(5,6)
9	$\{Ans(A)\}$	(5,7)

Quando utilizzare le tre modalità? Qualche esempio

1. *Verifiche di consistenza sulla conoscenza 'generale'* (relativa al dominio o al task), per sua natura 'frammentaria'
2. *Aggiornamento del modello dell'utente*: derivazione di caratteristiche dell'utente non note, dato un insieme di caratteristiche note e un insieme di regole di associazione fra caratteristiche.
3. *Aggiornamento della conoscenza sul dominio*.
4. *Individuazione dell'azione da suggerire* fra diverse azioni possibili; o dell'informazione da fornire, fra diverse alternative.

Uso della modalità 'risposta a domande T/F'

Sistema per il suggerimento di programmi televisivi.
So che l'utente è di genere femminile, ha 23 anni e sportiva.
Posso dedurre che le interessa vedere il derby Roma-Lazio?

Uso della modalità 'fill-in the blank'

Sistema per il suggerimento di programmi televisivi.
So che l'utente è di genere femminile, ha 23 anni e sportiva.
Quali spettacoli posso dedurre che le interessa vedere?
Quali spettacoli (molto probabilmente) detesta?

Strategie di Risoluzione

Per rendere più veloce la ricerca di soluzioni, si possono applicare due classi di strategie:

a. *Cancellazione di clausole* dall'insieme esaminato

b. *Definizione di un ordine* nella considerazione delle clausole

Strategie di Cancellazione

1. Eliminazione di 'letterali puri': un letterale è 'puro' se non compare in modo complementare (negato) in nessun'altra clausola
2. Eliminazione di tautologie: una clausola è una tautologia se contiene una coppia di letterali complementari
3. Eliminazione di sussunzioni: una clausola Φ subsume una clausola Ψ iff esiste una sostituzione λ tale che
   $\Phi\lambda \subseteq \Psi$ (cioè che rende Φ inclusa in Ψ)

Esempi di strategie di cancellazione

1 *Tautologie*: $\{\neg F(x,y), P(x,y), F(x,y)\}$

2 *Letterali puri*:

1. $\{\neg Mather(x,y), Parent(x,y)\}$
2. $\{Father(P,F)\}$
3. $\{Mather(A,F)\}$
4. $\{\neg Parent(x,F), Ans(x)\}$

Father(P,F) è un letterale puro.

3. *Sussunzioni*:

$\{\neg Mather(x,y), Parent(x,y)\}$ subsume
$\{\neg Mather(x,F), \neg Female(x), Parent(x,F)\}$
per la sostituzione y/F

Strategie basate sull'ordine: Risoluzione Unitaria

Almeno una delle clausole considerate nell'applicare la regola di risoluzione è una clausola unitaria, che contiene cioè un solo letterale.

Esempio:

1	{P,Q}	
2	{¬P,R}	
3	{¬Q,R}	
4	{¬R}	
5	{¬P}	2,4 *escludiamo cioè le coppie (1,2),(1,3)*
6	{¬Q}	3,4
7	{Q}	1,5
8	{P}	1,6
9	{R}	3,7
10	{}	6,7

Strategie basate sull'ordine: Risoluzione Ordinata

La risoluzione è consentita soltanto sul *primo letterale* di ciascuna delle due clausole.

Esempio:

1	{P,Q}	
2	{¬P,R}	
3	{¬Q,R}	
4	{¬R}	
5	{Q,R}	1,2
6	{R}	3,5
7	{}	4,6

Vediamo come usare questa strategia

Simulazione di Ragionamento Backward vs Forward

La risoluzione ordinata consente di simulare ricerche di soluzioni 'backward' o 'forward' a seconda di come sono ordinati i letterali nelle clausole considerate.

Facciamo l'ipotesi che tutte le clausole in Δ siano *clausole di Horn*. Possiamo ordinare gli elementi di ogni clausola in uno dei due modi seguenti:

a. *Ordinamento forward:* il letterale positivo si trova *alla fine* della clausola

{¬Mather(x,y), Parent(x,y)}

b. *Ordinamento backward:* il letterale positivo si trova *all'inizio* della clausola

{Parent(x,y), ¬Mather(x,y)}

Un esempio

Le zebre sono mammiferi di grandezza media con il manto a strisce. I mammiferi sono animali a sangue caldo.

Nina è una zebra. Il suo manto è non uniforme?

Zebra(x) → Mammifero(x) *…omettiamo i quantif. universali…*
Zebra(x) → GMedia(x)
Zebra(x) → AStrisce(x)
Mammifero(x) → Animale(x)
Mammifero(x) → SCaldo(x)
GMedia(x) → ¬GPiccola(x) *… conoscenza implicita…*
GMedia(x) → ¬GGrande(x)
AStrisce(x) → NonUniforme(x)
Zebra(NINA) NonUniforme(NINA)?

Rappresentiamo il problema in modo forward

{¬Zebra(x), Mammifero(x)}
{¬Zebra(x), GMedia(x)}
{¬Zebra(x), AStrisce(x)}
{¬Mammifero(x), Animale(x)}
{¬Mammifero(x), SCaldo(x)}
{¬GMedia(x), ¬GPiccola(x)}
{¬GMedia(x), ¬GGrande(x)}
{¬AStrisce(x), NonUniforme(x)}
{Zebra(NINA)}
{¬ NonUniforme(NINA)}

In quanti passi arriviamo a trovare la formula vuota?

Come funziona il ragionamento forward

Da dati iniziali positivi
Si deducono dati intermedi positivi
Che, alla fine, si risolvono con il goal negato

(propagazione 'in avanti' delle evidenze disponibili)

Rappresentiamo il problema in modo backward

{ Mammifero(x), ¬Zebra(x)}
{ GMedia(x), ¬Zebra(x)}
{ AStrisce(x), ¬Zebra(x)}
{ Animale(x), ¬Mammifero(x)}
{ SCaldo(x), ¬Mammifero(x)}
{ ¬GPiccola(x), ¬GMedia(x)}
{ ¬GGrande(x), ¬GMedia(x)}
{ NonUniforme(x), ¬AStrisce(x)}
{Zebra(NINA)}
{¬ NonUniforme(NINA)}

…… in quanti passi arriviamo a trovare la clausola vuota?

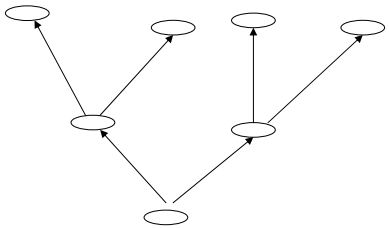
Come funziona il ragionamento backward

Da dati iniziali negativi (il goal negato)
Si deducono dati intermedi negativi (sobgoal negati)
Fino a risolvere con le evidenze disponibili.

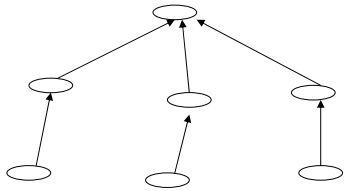
(deduzione per *decomposizione del goal in subgoal*)

Simulazione di Ragionamento Backward vs Forward

L'ordinamento più conveniente dipende dalla struttura del grafo di ricerca:



backward



forward

Coniglio(R)
∃y Levriero(y) ∧ (∀z Coniglio(z) → PiuVeloce(x,z))
EI: Levriero(L) ∧ (∀z Coniglio(z) → PiuVeloce(L,z))
AE: Levriero(L)
(∀z Coniglio(z) → PiuVeloce(L,z))
UI: Coniglio(R) → PiuVeloce(L,R)
MP: PiuVeloce(L,R)

∀y Levriero(y) → Cane(y)
UI: Levriero(L) → Cane(L)
MP: Cane(L)

Cavallo(F)
∀x∀y Cavallo(x) ∧ Cane(y) → PiuVeloce(x,y)
UI: Cavallo(F) ∧ Cane(L) → PiuVeloce(F,L)
AI: Cavallo(F) ∧ Cane(L)
MP: PiuVeloce(F,L)

∀x∀y ∀z PiuVeloce(x,y) ∧ PiuVeloce(y,z) → PiuVeloce(x,z)
UI: PiuVeloce(F,L) ∧ PiuVeloce(L,R) → PiuVeloce(F,R)
AI: PiuVeloce(F,L) ∧ PiuVeloce(L,R)
MP: PiuVeloce(F,R)

Riprendiamo L'esempio:

MP: modus ponens;
MT: modus tollens;
AE: and elimination;
AI: and introduction:
UI: universal instantiation;
EI: existential instantiation

Risolviamolo con il Principio di Risoluzione

∃y Levriero(y) ∧ (∀z Coniglio(z) → PiuVeloce(x,z))
∀y Levriero(y) → Cane(y)
∀x∀y Cavallo(x) ∧ Cane(y) → PiuVeloce(x,y)
∀x∀y ∀z PiuVeloce(x,y) ∧ PiuVeloce(y,z) → PiuVeloce(x,z)
Cavallo(F)
Coniglio(R)
PiuVeloce(F,R)?

{¬ Coniglio(z),PiuVeloce(x,z)} *conoscenza generale*
{¬ Levriero(L),Cane(L)}
{¬ Cavallo(F), ¬Cane(L), PiuVeloce(F,L)}
{¬ PiuVeloce(F,L), ¬ PiuVeloce(L,R), PiuVeloce(F,R)}
{Levriero(L)} *conoscenza specifica*
{Cavallo(F)}
{Coniglio(R)}
{ ¬ PiuVeloce(F,R) } *goal negato!*

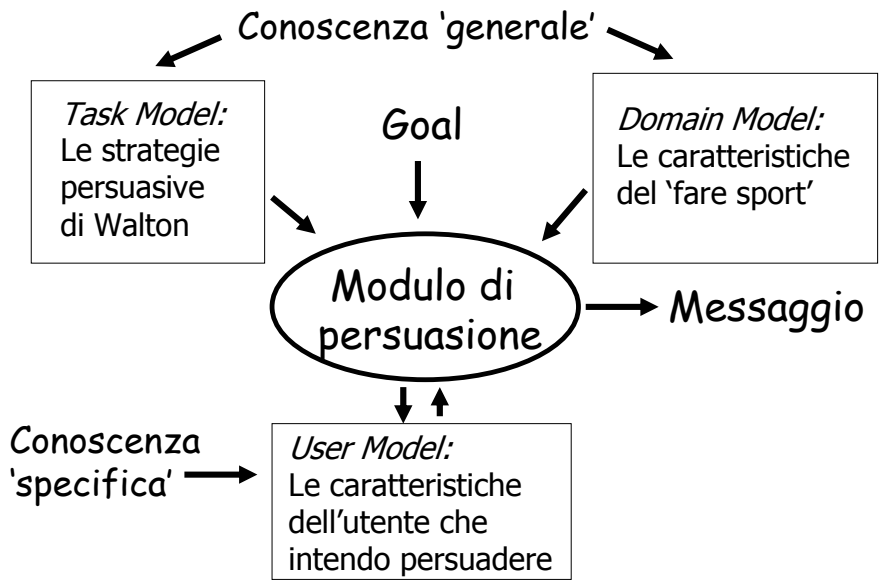
… sviluppate l'esercizio!!…

Riprendiamo l' esempio della persuasione

Ci proponiamo di realizzare un sistema che suggerisce all'utente lo sport più adatto a lui
(o a lei)
all'interno di un insieme di sport di cui conosce le caratteristiche.

L'esempio appartiene alla categoria della 'generazione di monologhi adattati all'utente'

Architettura del sistema



Le fonti di conoscenza

- Task model: le strategie persuasive di Walton

$\forall x \forall a \forall g$ ((Implies(a,g)$\wedge$Likes(x,g)$\wedge$CanDo(x,a))$\rightarrow$(ShouldDo(x,a))
$\forall x \forall a \forall g$ ((Implies(a,g)$\wedge\neg$Likes(x,g)$\wedge$CanAvoid(x,a))$\rightarrow$
$\neg$(ShouldDo(x,a))

- Domain model

$\forall s$ Sport(s) $\rightarrow$ Implies(s,GoodShape)
$\forall s$ Sport(s) $\rightarrow$ Implies(s,GoodHealth)
$\forall s \forall x$ (Sport(s) $\wedge$ Young(x) $\wedge$ Healthy(x)) $\rightarrow$ CanDo(x,s)
Sport(R)

- User model
  Young(G)
  Healthy(G)
  Likes(G,GoodShape)
  $\neg$Likes(G,GoodHealth) ……

Conoscenza in forma di clausole

Task model:

$\forall x \forall a \forall g$ ((Implies(a,g)$\wedge$Likes(x,g)$\wedge$CanDo(x,a))$\rightarrow$(ShouldDo(x,a))
$\forall x \forall a \forall g$ ((Implies(a,g)$\wedge\neg$Likes(x,g)$\wedge$CanAvoid(x,a))$\rightarrow$
$\neg$(ShouldDo(x,a))

Trasformiamo in clausole:

{$\neg$Implies(a,g), $\neg$Likes(x,g), $\neg$ CanDo(x,a), ShouldDo(x,a)}
{$\neg$Implies(a,g), Likes(x,g) $\neg$CanAvoid(x,a), $\neg$ShouldDo(x,a)}

Conoscenza in forma di clausole

Domain model:

$\forall s$ Sport(s) $\rightarrow$ Implies(s,GoodShape)
$\forall s$ Sport(s) $\rightarrow$ Implies(s,GoodHealth)
$\forall s \forall x$ (Sport(s) $\wedge$ Young(x) $\wedge$ Healthy(x)) $\rightarrow$ CanDo(x,s)
Sport(R)
Sport(A)

In clausole:

{$\neg$ Sport(s), Implies(s, GoodShape)}
{$\neg$Sport(s), Implies(s, GoodHealth)}
{$\neg$Sport(s), $\neg$Young(x), $\neg$Healthy(x)), CanDo(x,s))}
{Sport(R)}{Sport(A)}

Conoscenza in forma di clausole

User model :

Young(G)
Healthy(G)
Likes(G,GoodShape)
¬Likes(G,GoodHealth)

In clausole:

{Young(G)}
{Healthy(G)}
{Likes(G, GoodShape)}
{¬Likes(G,GoodHealth)}

Il Goal negato (in modalità 'fill-in the blank')
{¬ShouldDo(G,a), Ans(a)}: ***quale sport*** dovrebbe praticare Giuseppe?

… Complessivamente:

1. {¬Implies(a,g), ¬Likes(x,g), ¬ CanDo(x,a), ShouldDo(x,a)}
2. {¬Implies(a,g), Likes(x,g) ¬CanAvoid(x,a), ¬ShouldDo(x,a)}
3. {¬ Sport(s), Implies(s, GoodShape)}
4. {¬Sport(s), Implies(s, GoodHealth)}
5. {¬Sport(s), ¬Young(x), ¬Healthy(x)), CanDo(x,s))}
6. {Sport(R)}
7. {Sport(A)}
8. {Young(G)}
9. {Healthy(G)}
10. {Likes(G, GoodShape)}
11. {¬Likes(G,GoodHealth)}
12. {¬ShouldDo(G,a), Ans(a)}: ***quale sport*** dovrebbe praticare Giuseppe?

… provate a risolvere, con la strategia unitaria o con quella ordinata.

Cosa fare dei risultati

Assumiamo che il quesito precedente abbia prodotto le risposte:

{Ans(R)}
{Ans(A)}

Questi risultati dovranno essere tradotti in un messaggio in linguaggio naturale, del tipo:

"Dovresti andare a correre, Giuseppe! Oppure, potresti fare Aikido!

… Vedremo in seguito come questo si può fare, con un 'generatore di linguaggio naturale'

Estendiamo il quesito

Ma

può essere necessario, per aumentare la *forza* del messaggio persuasivo, spiegare anche il *perché del suggerimento,* generando quindi un messaggio più complesso:

"Dovresti andare a correre, Giuseppe! **Il suggerimento**
Sei giovane, sano, ci tieni ad essere in forma, **I dati su U**
e correre aiuta a mantenersi in forma!" **La regola generale**

Come farlo? Occorre, in qualche modo, tenere traccia del processo di derivazione che ha portato ad ottenere la conclusione (o le conclusioni).

1	{¬Implies(a,g), ¬Likes(x,g), ¬ CanDo(x,a), ShouldDo(x,a)}	
2	{¬Implies(a,g), Likes(x,g) ¬CanAvoid(x,a), ¬ShouldDo(x,a)}	
3	{¬ Sport(s), Implies(s, InShape)}	
4	{¬Sport(s), Implies(s, GoodHealth)}	
5	{¬Sport(s), ¬Young(x), ¬Healthy(x)), CanDo(x,s))}	
6	{Sport(RUNNING)}	
7	{Sport(AIKIDO)}	
8	{Young(G)}	
9	{Healthy(G)}	
10	{Likes(G, GoodShape)}	
11	{¬Likes(G,GoodHealth)}	
12	{¬ShouldDo(G,a), Ans(a)} ... risolviamo...	
13	**Implies(RUNNING,GoodShape)**	**(3,6)**
14	**Implies(RUNNING,GoodHealth)**	**(4,6)**
15	**Implies(AIKIDO,GoodShape)**	**(3,7)**
16	**Implies(AIKIDO,GoodHealth)**	**(4,7)**
17	**¬Young(x), ¬Healthy(x)), CanDo(x,RUNNING)**	**(5,6)**
18	**¬Young(x), ¬Healthy(x)), CanDo(x,AIKIDO)**	**(5,7)**
19	**¬Healthy(G)), CanDo(G,RUNNING)**	**(8,17)**
20	**¬Healthy(G)), CanDo(G,AIKIDO)**	**(8,18)**
21	**CanDo(G,RUNNING)**	**(9,19)**
22	**CanDo(G,AIKIDO)**	**(9,20)**
23	**¬Likes(x,GoodShape), ¬ CanDo(x,RUNNING), ShouldDo(x,RUNNING)**	**(1,13)**
24	**¬Likes(x,GoodHealth), ¬ CanDo(x,RUNNING), ShouldDo(x,RUNNING)**	**(1,14)**
25	**¬Likes(x,GoodShape), ¬ CanDo(x,AIKIDO), ShouldDo(x,AIKIDO)**	**(1,15)**
26	**¬Likes(x,GoodHealth), ¬ CanDo(x,AIKIDO), ShouldDo(x,AIKIDO)**	**(1,16)**
27	**¬ CanDo(x,RUNNING), ShouldDo(x,RUNNING)**	**(10,23)**
28	**¬ CanDo(x,AIKIDO), ShouldDo(x,AIKIDO)**	**(10,25)**
29	**ShouldDo(x,RUNNING)**	**(21,27)**
30	**ShouldDo(x,AIKIDO)**	**(21,28)**
31	**Ans(RUNNING)**	**(12,29)**
32	**Ans(AIKIDO)**	**(12,30)**

1	{¬Implies(a,g), ¬Likes(x,g), ¬ CanDo(x,a), ShouldDo(x,a)}	
3	{¬ Sport(s), Implies(s, InShape)}	
5	{¬Sport(s), ¬Young(x), ¬Healthy(x)), CanDo(x,s))}	
6	*{Sport(RUNNING)}*	
7	{Sport(AIKIDO)}	
8	*{Young(G)}*	
9	*{Healthy(G)}*	
10	*{Likes(G, GoodShape)}*	
11	{¬Likes(G,GoodHealth)}	
12	{¬ShouldDo(G,a), Ans(a)}	
13	***Implies(RUNNING,GoodShape)***	**(3,6)**
14	**Implies(RUNNING,GoodHealth)**	**(4,6)**
15	**Implies(AIKIDO,GoodShape)**	**(3,7)**
16	**Implies(AIKIDO,GoodHealth)**	**(4,7)**
17	**¬Young(x), ¬Healthy(x)), CanDo(x,RUNNING)**	**(5,6)**
18	**¬Young(x), ¬Healthy(x)), CanDo(x,AIKIDO)**	**(5,7)**
19	**¬Healthy(G)), CanDo(G,RUNNING)**	**(8,17)**
20	**¬Healthy(G)), CanDo(G,AIKIDO)**	**(8,18)**
21	***CanDo(G,RUNNING)***	**(9,19)**
22	**CanDo(G,AIKIDO)**	**(9,20)**
23	**¬Likes(x,GoodShape), ¬ CanDo(x,RUNNING), ShouldDo(x,RUNNING)**	**(1,13)**
24	**¬Likes(x,GoodHealth), ¬ CanDo(x,RUNNING), ShouldDo(x,RUNNING)**	**(1,14)**
25	**¬Likes(x,GoodShape), ¬ CanDo(x,AIKIDO), ShouldDo(x,AIKIDO)**	**(1,15)**
26	**¬Likes(x,GoodHealth), ¬ CanDo(x,AIKIDO), ShouldDo(x,AIKIDO)**	**(1,16)**
27	**¬ CanDo(x,RUNNING), ShouldDo(x,RUNNING)**	**(10,23)**
28	**¬ CanDo(x,AIKIDO), ShouldDo(x,AIKIDO)**	**(10,25)**
29	***ShouldDo(x,RUNNING)***	**(21,27)**
30	**ShouldDo(x,AIKIDO)**	**(21,28)**
31	**Ans(RUNNING)**	**(12,29)**

Indichiamo le clausole utilizzate nel processo di inferenza

Consideriamo solo la 31.

31: (12,29); 29: (21,27); (21: (9,19); 19: (8,17); 17: (5,6); 27: (10,23); 23: (1,13); 13: (3,6);

3 {¬ IsASport(s), Implies(s, InShape)}
5 {¬Sport(s), ¬Young(x), ¬Healthy(x)), CanDo(x,s))}
6 {Sport(RUNNING)}
8 {Young(G)}
9 {Healthy(G)}
10 {Likes(G, GoodShape)}
12 {¬ShouldDo(G,a), Ans(a)}
13 Implies(RUNNING,GoodShape)
17 ¬Young(x), ¬Healthy(x)), CanDo(x,RUNNING)
19 ¬Healthy(G)), CanDo(G,RUNNING)
21 CanDo(G,RUNNING)
23 ¬Likes(x,GoodShape), ¬ CanDo(x,RUNNING), ShouldDo(x,RUNNING)
29 ShouldDo(x,RUNNING)
31 Ans(RUNNING)

31: (12,29); 29: (21,27); ((21: (9,19); 19: (8,17); 17: (5,6);) 27: (10,23); 23: (1,13); 13: (3,6);

Indico in rosso le clausole che corrispondono a dati iniziali sull'Utente, utilizzati nel processo di derivazione, e le clausole unitarie derivate

6 {Sport(RUNNING)}
8 {Young(G)}
9 {Healthy(G)}
10 {Likes(G, GoodShape)}
13 Implies(RUNNING,GoodShape)
21 CanDo(G,RUNNING)
29 ShouldDo(x,RUNNING)
31 Ans(RUNNING)

31: (12,29); 29: (21,27); ((21: (9,19); 19: (8,17); 17: (5,6);) 27: (10,23); 23: (1,13); 13: (3,6);

Colleghiamo con 'connettivi linguistici'

Ans(RUNNING) because
CanDo(G,RUNNING) as
(Healthy(G) and Young(G) and Sport(RUNNING)) and
Likes(G, GoodShape) and Implies(Running,GoodShape)

… oppure in ordine inverso

Notare che non tutte le clausole sono rilevanti:
(alcune corrispondono
a risultati intermedi non significativi).

Quesito:

Prova a delineare un algoritmo per individuare,
a partire dalla conclusione,
quali dati hanno contribuito in modo ***rilevante***
al processo di inferenza
E generare la 'struttura' del messaggio da generare.

Esercizio

Estendi l'esempio della persuasione inserendo:

- L'appello alle conseguenze negative
- Diversi sport, con caratteristiche diverse
- Maggiori informazioni sull'utente

Trasforma la conoscenza in clausole e risolvi,
ponendo uno o più quesiti di tipo 'fill-in the blank'.

Riferimenti

M.R.Genesereth e N J Nilsson: Logical Foundations of Artificial Intelligence. Morgan Kaufman, 1986.
Capitoli 5 e 6