Tecniche di Decomposizione dei Problemi

Sommario

- Costruzione di algoritmi
- Approccio alla soluzione di problemi complessi
- Sequenza; Selezione; Iterazione; Ricorsione

Algoritmi: Costruzione

- Esaminare una specifica realtà o problema
- Costruirne un'astrazione
- Rappresentarla (più o meno) formalmente
- Individuare una sequenza di azioni che, eseguite, risolvano il problema nel mondo dell'astrazione
  - Il processo di analisi e astrazione è difficile da dominare per problemi complessi

Tipi di problemi

- Primitivi
  - Risolubili mediante
    - Un'azione primitiva
    - Una sequenza di azioni primitive
- Complessi
  - Non risolubili tramite un'azione nota
    - Al solutore
    - All'esecutore
- Ha senso occuparsi dei problemi complessi!

Problemi complessi: Soluzione

- Decomposizione del problema in sottoproblemi
  - fino a trovare un insieme di sottoproblemi primitivi che risulti equivalente al problema di partenza
- Individuazione del procedimento che porta alla soluzione
  - Processo di cooperazione tra sottoproblemi dei quali si è in grado di fornire facilmente una soluzione

Problemi complessi: Approccio

- Principio DIVIDE ET IMPERA
  - Ridurre la complessità del problema
  - decomposizione in sottoproblemi, la cui risoluzione è più semplice
    - Individuare le opportune strutture e relazioni
  - Realizzare un algoritmo per ogni sottoproblema
  - Se un sottoproblema continua ad essere troppo complesso, riapplicare ad esso la decomposizione fino ad arrivare a problemi primitivi
    - risolubili tramite azioni note

Decomposizione di problemi (1)

- Tecnica per raffinamenti successivi
  - Uno dei principi della programmazione strutturata
  - Basata sulla trasformazione di un problema in una gerarchia di problemi di difficoltà decrescente
    - Di un problema si affronta, prima, l'aspetto più generale e si passa, poi, a livelli sempre più dettagliati di descrizione sino ad arrivare agli elementi fondamentali

Decomposizione di problemi (2)

- Conseguenze
  - Aumento del numero di problemi da risolvere
  - Diminuzione della complessità di ciascuno
- Può capitare che, per risolvere il problema complessivo, uno stesso sottoproblema debba essere risolto più volte
  - considerando ogni volta valori diversi dei dati

Decomposizione di Problemi: Requisiti

- Ogni passo della suddivisione deve garantire che
  - la soluzione dei sottoproblemi conduca alla soluzione generale
  - la successione di passi da eseguire abbia senso e sia possibile
  - la suddivisione scelta dia luogo a sottoproblemi "più vicini" agli strumenti disponibili
    - Risolubili direttamente con gli operatori a disposizione

Decomposizione di Problemi: Quando fermarsi?

- Bisogna arrivare al punto in cui
  - Tutti i problemi sono primitivi
  - È fissato l'ordine di esecuzione dei sottoproblemi
  - È previsto il modo in cui un sottoproblema utilizza i risultati prodotti dalla soluzione dei sottoproblemi che lo precedono
    - Livello di cooperazione

Decomposizione di Problemi: Livelli

- I problemi ottenuti dalla decomposizione sono
  - Indipendenti
    - Si può progettare un algoritmo per ciascuno
    - La soluzione può essere delegata a solutori diversi (in parallelo)
  - Cooperanti
    - Ciascuno può usare il risultato della soluzione di altri
- Più complesso è il problema, più livelli saranno necessari in profondità
- Uno stesso problema può essere scomposto in modi diversi

Decomposizione di Problemi: Tecniche (1)

- È possibile scomporre I problemi secondo le seguenti tecniche
  - Sequenziale
    - Suddividere un problema in parti disgiunte
  - Selettiva
    - Trattamento differenziato in casi differenti
  - Iterativa
  - Ricorsiva

Decomposizione di Problemi: Tecniche (2)

- Nota: le precedenti sono tecniche di decomposizione
- I linguaggi di programmazione mettono a disposizione costrutti linguistici per realizzare tali tecniche
  - Tramite operatori linguistici espliciti (sequenza, selezione, iterazione)
  - Tramite meccanismi ad hoc (ricorsione)

Decomposizione Sequenziale

- La soluzione si ottiene tramite una sequenza di passi
  - I passi sono eseguiti uno alla volta
  - Ogni passo è eseguito una sola volta
    - Nessuno è ripetuto o omesso
  - L'ordine in cui i passi vanno eseguiti è lo stesso in cui sono scritti
  - L'ultimo passo equivale alla terminazione del procedimento

Decomposizione Sequenziale: Esempio (1)

- Problema: Preparare una tazza di tè
- Soluzione: un algoritmo per la preparazione della tazza di tè
  - Passo1: Bollire l'acqua
  - Passo2: Mettere il tè nella tazza
  - Passo3: Versare l'acqua nella tazza
  - Passo4: Lasciare in infusione
- Non sono problemi primitivi
  - Ciascuno va ulteriormente scomposto

Decomposizione Sequenziale: Esempio (2)

- Bollire l'acqua:
  - Riempire d'acqua il bollitore
  - Accendere il gas
  - Aspettare che l'acqua bolla
  - Spegnere il gas
- Mettere il tè nella tazza:
  - Aprire la scatola del tè
  - Prendere un sacchetto-filtro
  - Chiudere la scatola del tè
  - Mettere il sacchetto nella tazza
- Versare l'acqua nella tazza:
  - Versare l'acqua bollente nella tazza finché non è piena
- Lasciare in infusione:
  - Aspettare 3 minuti
  - Estrarre il sacchetto-filtro

Decomposizione Selettiva

- Soluzione ottenuta tramite scelte multiple
  - Strutture di selezione all'interno di altre strutture di selezione
- La decomposizione fa ricorso a strutture multiple
  - Nidificazione (o annidamento) di strutture

Decomposizione Selettiva: Esempio (1)

- Problema: trovare il maggiore fra tre numeri a, b, c (diversi tra di loro)
  - Supponiamo che verificare se un numero è maggiore di un altro sia un'azione primitiva

Decomposizione Selettiva: Esempio (2)

- Algoritmo:

```
se a>b
   allora se a > c
            allora a è la soluzione
            altrimenti c è la soluzione
   altrimenti se b > c
            allora b è la soluzione
            altrimenti c è la soluzione
```


Decomposizione Iterativa

- Successione di problemi tutti dello stesso tipo
  - Ripetere un numero finito di volte un passo di soluzione in modo da ottenere, alla fine, la soluzione completa

Decomposizione Iterativa: Requisiti

- Individuare una catena di sottoproblemi che:
  - sono uguali, oppure
  - differiscono solo per i dati su cui agiscono
- ed inoltre i dati su cui agiscono
  - sono uguali, oppure
  - sono in una qualche relazione d'ordine
    - ciascun problema differisce dal precedente perché opera sul dato successivo

Decomposizione Iterativa: Esempio (1)

- Trovare il più grande fra n > 3 numeri tutti diversi fra loro
  - Supponiamo che i dati siano in relazione d'ordine
    - Si può parlare di 1° dato, 2° dato, … , n-esimo dato

Decomposizione Iterativa: Esempio (2)

Lo stesso passo (trovare il più grande) è ripetuto n-1 volte, su diverse coppie di dati

- Trova il più grande tra i primi 2 numeri
- Trova il più grande fra il risultato del problema precedente e il 3° numero
- …
- Trova il più grande fra il risultato del problema precedente e l'ultimo numero (n-esimo dato)

- Più concisamente:
  - Trova il più grande fra i primi 2 numeri
  - Finché ci sono numeri da esaminare esegui
    - Esamina il primo numero non ancora considerato
    - Trova il più grande tra questo e il più grande precedentemente trovato

Oggetto Ricorsivo

- Definito in termini di se stesso
  - Esempi
    - Numeri naturali:
      - 1 è un numero naturale
      - Il successore di un numero naturale è un numero naturale
    - Strutture ad albero:
      - • è un albero (albero vuoto)
      - Se t1 e t2 sono alberi, allora anche la struttura
      -

è un albero

Definizione Ricorsiva

- Ottenuta in termini di versioni più semplici della stessa cosa che definisce
- 2 livelli:
  - Passo
    - Operazione che riconduce la definizione ad un dato livello a quella di livello inferiore
  - Base (o livello assiomatico)
    - Definizione di partenza
      - Necessario per ricostruire i livelli successivi
      - Consente di definire problemi di ordine superiore

Ricorsione: Esempio

Decomposizione Ricorsiva

- Un problema di ordine n è definito in termini del medesimo problema di ordine inferiore
  - Entità definita in termini più semplici della stessa entità
  - Nella definizione deve apparire il livello assiomatico (o di ricostruzione)
    - Problema risolubile tramite un'azione primitiva

Decomposizione Ricorsiva: Requisiti

- Almeno uno dei sottoproblemi è formalmente uguale al problema di partenza, ma di ordine inferiore
- Al momento della decomposizione è noto l'ordine del problema
  - decomposizione mediante regola di copiatura
    - Sostituire ad ogni occorrenza del problema la decomposizione ricorsiva
  - fino ad avere problemi primitivi (raggiungere il livello assiomatico)

Decomposizione Ricorsiva: Esempio (1)

- Invertire una sequenza di lettere

  – se la sequenza contiene una sola lettera allora
    - Scrivila (il problema è risolto)
  – altrimenti:
    - Rimuovi la prima lettera dalla sequenza
    - Inverti la sequenza rimanente
    - Appendi in coda la lettera rimossa

Decomposizione Ricorsiva Esempio (2)

- Invertire “roma”
  – “roma” (4 lettere): rimuovi ‘r’, inverti “oma”
    - “oma” (3 lettere): rimuovi ‘o’, inverti “ma”
      – “ma” (2 lettere): rimuovi ‘m’, inverti “a”
        » “a” (1 lettera): è già invertita
      – Appendi ‘m’: “am”
    - Appendi ‘o’: “amo”
  – Appendi ‘r’: “amor”
- Risultato: “amor”

Decomposizione Ricorsiva: Caratteristiche

- Un problema esprimibile ricorsivamente si può risolvere iterativamente
- Una funzione esprimibile ricorsivamente è computabile
  – Esiste un algoritmo che la calcola e termina sempre
- Ogni funzione computabile per mezzo di un programma è ricorsiva

Iterazione e Ricorsione: Esempio (1)

- Calcolo del prodotto di due interi n e m
  – Definizione iterativa:
    - n * m = m + m + … + m (n volte)
  – Definizione ricorsiva:

$$n * m = \begin{cases} 0 & \text{se } n = 0 \\ (n-1) * m + m & \text{se } n > 0 \end{cases}$$

Iterazione e Ricorsione: Esempio (2)

- Soluzione iterativa
  - inizialmente sia il risultato uguale a 0
  - ripeti per ogni intero da 1 a n
    - somma m al risultato per ottenere un nuovo risultato
- il prodotto è il risultato finale

- Soluzione ricorsiva
  - se n è uguale a 0
    - allora il prodotto è 0
    - altimenti
      - calcola il prodotto di n – 1 per m
      - somma m al prodotto

Decomposizione di Problemi: Conclusioni

- Può esserci più di una decomposizione di un problema in sottoproblemi
- Cause di difficoltà nella decomposizione
  - Comprensione intuitiva della complessità di un problema
  - Scelta tra diverse possibili scomposizioni
  - Necessità di una formulazione “adeguata” per ciascun sottoproblema