Corso di Laurea Ingegneria Informatica

Fondamenti di Informatica 2

Dispensa 04

Introduzione ai Tipi astratti di dato

A. Miola

Febbraio 2007

Contenuti

- **Soluzione automatica di problemi**
  - Specifica e rappresentazione di un algoritmo
  - Tipi astratti e tipi concreti
  - Specifica di un tipo astratto
- **Il tipo astratto Insieme**
  - Specifica del tipo astratto Insieme
  - Operazioni primitive e non
- **Rappresentazione del tipo astratto Insieme**
  - Rappresentazione mediante interfacce
  - La classe InsConArray
  - Efficienza della rappresentazione

Soluzione automatica di problemi . . .

- La soluzione automatica di un problema richiede di svolgere attività piuttosto articolate che comprendono:
  - la specifica di un algoritmo risolutivo
  - l'implementazione dell'algoritmo nel linguaggio di programmazione prescelto
- In effetti, come abbiamo già avuto modo di vedere anche nei casi di semplici problemi, ciascuna delle due attività coinvolge molteplici fasi (in particolare la valutazione della correttezza della sua implementazione)

. . . Soluzione automatica di problemi

- **In particolare è da osservare che la specifica dell'algoritmo risolutivo comprende due attività strettamente correlate tra loro:**
  - **la specifica dei dati e delle loro proprietà**
  - **la specifica delle operazioni da compiere su di essi al fine di pervenire alla soluzione del problema posto**
- **Questa osservazione dovrebbe essere ormai evidente**
  - **Pensiamo ad esempio al caso del problema della ricerca: diversi insiemi di dati e diverse loro proprietà determinano la scelta di algoritmi diversi, più o meno efficienti.**

Rappresentazione di un algoritmo

- **Altrettanto possiamo dire circa l'attività di implementazione dell'algoritmo in un linguaggio di programmazione**
  - **si tratta infatti di implementare (rappresentare) sia i dati che le operazioni su di essi, come un'unica attività di rappresentazione di un algoritmo**
    - **Si pensi alla specifica dell'algoritmo per il calcolo del Massimo Comun Divisore di due numeri interi x e y che prevede la specifica e la determinazione di:**
      - **l'insieme X dei divisori di x e l'insieme Y dei divisori di y**
      - **l'insieme Z intersezione di X e di Y**
      - **il massimo valore dell'insieme Z**
- **Si pone quindi l'esigenza di rappresentare in particolare gli insiemi di numeri interi con relative operazioni**

Rappresentazione di dati e operazioni . . .

- Questa attività di rappresentazione di dati e operazioni consiste essenzialmente nel determinare una corrispondenza tra i dati, così come specificati nell'algoritmo, e le tipologie di dati, o di strutture di dati, disponibili nel linguaggio di programmazione prescelto
  - Si pensi ad esempio ai tipi primitivi di Java e alla struttura di array di Java
- In maniera analoga si tratta di ragionare circa la rappresentazione delle operazioni sulle rappresentazioni dei dati

. . . Rappresentazione di dati e operazioni

- Riferiamoci al tipo Java `int`. Quando dichiariamo un dato di tipo `int` , numero intero, non solo stiamo indicando la modalità della sua rappresentazione binaria nelle celle della memoria nonché il suo valore massimo consentito, ma anche quelle che saranno le operazioni consentite, ammissibili, su di esso

- Nella dichiarazione del tipo di un dato c'è, per così dire, un aspetto statico e un aspetto dinamico

Tipi astratti e tipi concreti . . .

- **Supponiamo che venga posto un problema relativo ai numeri interi**
  - **ad esempio il semplice problema di determinare il prodotto di due numeri**
- **Sappiamo bene che volendo risolvere un problema come questo, magari anche manualmente, non possiamo operare "astrattamente" sui numeri interi intesi come tipi astratti, ma dobbiamo passare ad una rappresentazione dei tipi astratti mediante tipi concreti come i numerali corrispondenti tipicamente rappresentati in base 10**

. . . Tipi astratti e tipi concreti

- Creiamo quindi una corrispondenza tra il tipo astratto dei numeri interi e il tipo concreto dei numerali in base 10
- Se invece vogliamo risolvere automaticamente il problema posto, ad esempio utilizzando Java, dobbiamo creare una corrispondenza tra il tipo astratto dei numeri interi e il tipo `int` di Java
  - Anche nel caso di linguaggi di programmazione parliamo di tipi concreti per riferirci ai tipi del linguaggio
  - Il tipo `int` è un tipo concreto di Java

Tipi astratti . . .

- **Consideriamo il tipo astratto dei numeri naturali con le relative operazioni, in genere denotato da**

$$\mathcal{N}at = \langle \mathcal{N}, +, -, *, =, <, 0, 1 \rangle$$

  - **l'insieme di tutti i numeri naturali:** $\mathcal{N}$
  - **un insieme di operazioni (funzioni o predicati):**

$$+, -, *, =, <$$

  - **un insieme di costanti: 0 e 1 , che consentono di generare tutti gli elementi di** $\mathcal{N}$ **mediante le operazioni disponibili**

. . . Tipi astratti

- **In generale un tipo astratto, inteso come ente matematico, è definito come**
  - un insieme di oggetti (detto il dominio del tipo)
  - un insieme di operazioni (funzioni o predicati del tipo)
  - un insieme di costanti (che denotano alcuni elementi del dominio del tipo)
- **Nell'esempio del tipo astratto dei naturali**
  - un insieme di oggetti { $\mathcal{N}$ }
  - un insieme di operazioni { + , - , * , = , < }
  - un insieme di costanti { 0 , 1 }

Un esempio: il tipo astratto Insieme

- **Tipo astratto Insieme di oggetti di un certo dominio V**
  - **1. Il dominio Ins**

    (tutti gli insiemi di oggetti di un certo dominio V)
    - **Il dominio Ins, dominio del tipo, è detto anche dominio di interesse;**
    - **V è detto dominio di sostegno**
  - **2. Le operazioni**
    - **Vuoto (verifica se un insieme è vuoto)**
    - **Inserisci (inserisce un elemento)**
    - **Cancella (cancella un elemento)**
    - **Contiene (verifica se un insieme contiene un elemento)**
  - **3. La costante Insieme_Vuoto**

La specifica delle operazioni . . .

Vuoto

Dato un insieme H (cioè un elemento di Ins, dominio di interesse), verifica se esso contiene o meno elementi

Vuoto : Ins ⟶ Boolean

Inserisci

Dato un insieme H e un oggetto E, appartenente all'insieme V, fornisce come risultato l'insieme costituito da tutti gli elementi di H con l'aggiunta dell'elemento E (se non presente), oppure l'insieme H

Inserisci : Ins x V ⟶ Ins

. . . La specifica delle operazioni

Cancella

Dato un insieme H e un elemento E di V, fornisce come risultato l'insieme costituito da tutti gli elementi di H tranne l'elemento E (se presente), oppure l'insieme H

Cancella : Ins x V ⟶ Ins

Contiene

Dato un insieme H e un elemento E di V, verifica se E e' contenuto in H

Contiene : Ins x V ⟶ Boolean

Il tipo astratto Insieme

- Va notato che nella specifica delle operazioni si fa riferimento a vari domini: oltre al dominio Ins, si fa infatti riferimento anche ai domini V e Boolean
- La denotazione del tipo di dato astratto Insieme è quindi data dalla terna

$$< S , F , C >$$

  - S = { Ins , V , Boolean }
  - F = { Vuoto , Inserisci , Cancella , Contiene }
  - C = { Insieme_Vuoto }

Operazioni primitive e non

- **Le operazioni specificate in un tipo astratto di dato, vengono dette operazioni primitive**
  - **Nel tipo astratto dei numeri naturali le operazioni + , - e * sono operazioni primitive**
  - **Nel tipo astratto Insieme le operazioni Vuoto, Inserisci, Cancella e Contiene sono operazioni primitive**
- **A partire dalle operazioni primitive è possibile definire altre operazioni, come ad esempio l'operazione Unione per calcolare l'unione di due insiemi**

Unione : Ins x Ins ⟶ Ins

Operazioni primitive e non

Unione

Dato un insieme A e un insieme B fornisce come risultato l'insieme costituito da tutti gli elementi di A ampliato con tutti gli elementi dell'insieme B (non presenti in A), oppure l'insieme A

. . .

```
if   Vuoto(B) risultato = A
  else considera un qualunque elemento b di B
      if Contiene(A,b) risultato = Unione  (A, Cancella (B, b) )
      else risultato = Inserisci (Unione  (A, Cancella (B, b) ), b) ;
```

. . .

Rappresentazione di tipo un astratto . . .

- I tipi astratti stanno ai tipi concreti, propri di un linguaggio di programmazione, come gli algoritmi stanno ai programmi in quel linguaggio di programmazione
- Per poter operare effettivamente (automaticamente) su dati astratti è necessario determinare una loro rappresentazione nel contesto linguistico di un linguaggio di programmazione scelto

Rappresentazione di tipo un astratto

- **Per rappresentare un tipo astratto < S , F , C > è necessario rappresentare:**
  - **il dominio di interesse e ciascuno degli altri domini in S**
  - **ciascuna delle operazioni in F**
  - **ciascuna delle costanti in C**
- **Riprendiamo l'esempio del tipo astratto Insieme di oggetti di V e poniamoci l'obiettivo di determinare una sua rappresentazione in Java**

Rappresentazione del tipo astratto Insieme

- Consideriamo il caso degli insiemi costituiti da oggetti che siano numeri interi appartenenti al dominio $\mathcal{Z}$
- Partiamo dalla rappresentazione dei domini di S = { Ins , $\mathcal{Z}$, Boolean }
- In questo caso il dominio $\mathcal{Z}$ può essere direttamente rappresentato dal tipo primitivo Java `int`, così come Boolean è rappresentato dal tipo primitivo Java `boolean`
- Si tratta ora di determinare una rappresentazione per il dominio di interesse Ins

Rappresentazione del dominio di interesse

- **Per essere ancora più concreti, consideriamo il caso degli insiemi costituiti da numeri interi compresi tra 1 e 5**
  - **In questo caso gli elementi che possono comporre i nostri insiemi sono quindi appartenenti al dominio di sostegno** $V = \{1, 2, 3, 4, 5\}$
- **Consideriamo l'insieme** $H = \{2, 3, 5\}$

Rappresentazione del dominio di interesse

- Un qualsiasi insieme H del dominio Ins può essere rappresentato da un array di 5 elementi di tipo `boolean[5]`

```
class Insieme  {
   private boolean[] ins; //array degli elementi}
```

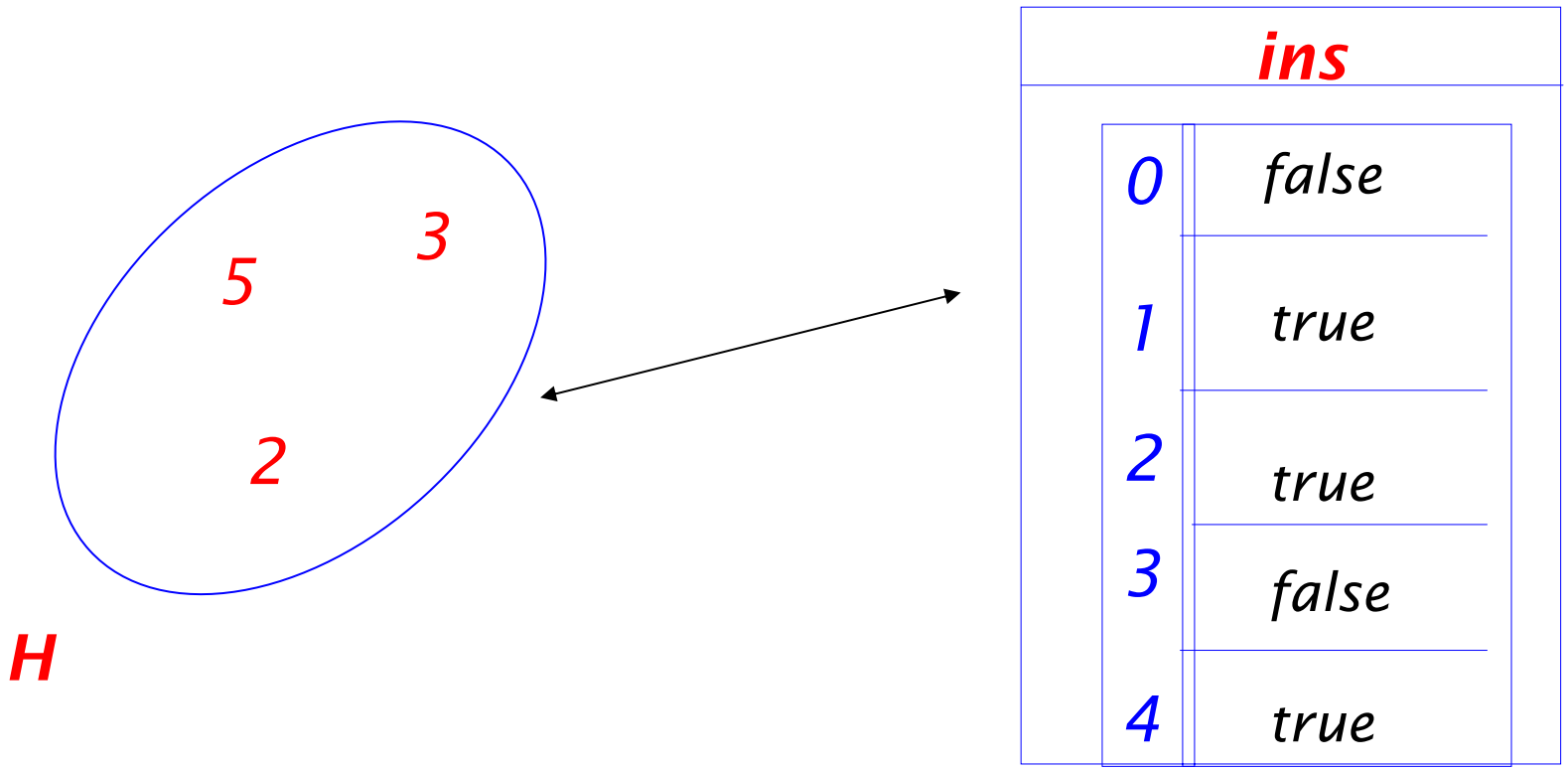
- Si stabilisce quindi una corrispondenza biunivoca tra l'insieme dei valori del tipo astratto Insieme di elementi di

$$V = \{1, 2, 3, 4, 5\}$$

e l'insieme dei valori di tipo array di 5 elementi di tipo `boolean`

Rappresentazione del tipo astratto Insieme

Il valore del generico elemento ins[k] dell'array sarà true se il valore dell'intero k+1 appartiene ad H, false altrimenti.



Rappresentazione dei domini

- **Rappresentazione dei domini del tipo astratto Insieme**

$\mathcal{Z}$	⟶	`int`
Boolean	⟶	`boolean`
Ins	⟶	`Insieme`

Definizione di Tipo astratto di dato

- **Un tipo astratto di dato è una tripla**

$$< S , F , C >$$

dove

- **S è un insieme di domini $\{ V_1, V_2, \ldots , V_n \}$, tra cui un dominio speciale chiamato "dominio di interesse"**
- **F è un insieme di funzioni (o operazioni) $\{ F_1, F_2, \ldots , F_m \}$, ciascuna delle quali è del tipo**

$$F_i : V_{i1} \times V_{i2} \times \ldots \times V_{ih} \longrightarrow V_{ik}$$

dove ogni elemento di $\{V_{i1}, V_{i2}, \ldots , V_{ih}, V_{ik}\}$ è un elemento di S, e uno tra essi è il dominio di interesse

  - **il dominio di interesse o è il codominio di F_i oppure è uno dei domini del dominio (prodotto cartesiano) di F_i**
- **C è un insieme di elementi del dominio di interesse**

Esempio . . .

- **Consideriamo il tipo astratto dei numeri naturali con le relative operazioni, in genere denotato da**

$$\mathcal{N}at = \langle \mathcal{N}, +, -, *, =, <, 0, 1 \rangle$$

  - **l'insieme di tutti i numeri naturali:** $\mathcal{N}$
  - **un insieme di operazioni (funzioni o predicati):**

$$+, -, *, =, <$$

  - **un insieme di costanti: 0 e 1 , che consentono di generare tutti gli elementi di** $\mathcal{N}$ **mediante le operazioni disponibili**

. . . Esempio

- **Secondo la definizione data con la tripla**

$$< S , F , C >$$

il tipo astratto dei naturali è quindi definito come

- **S è l'insieme di domini {$\mathcal{N}$, Boolean }, con $\mathcal{N}$ "dominio di interesse"**
- **F è l'insieme di funzioni { + , - , * , = , < }**
- **C è l'insieme di costanti { 0 , 1 }**

Il tipo astratto Insieme

- **Secondo la definizione data con la tripla**

< S , F , C >

la denotazione del tipo di dato astratto

Insieme è quindi data da

- **S = { Ins , V , Boolean }**
- **F = { Vuoto , Inserisci , Cancella , Contiene }**
- **C = { Insieme_Vuoto }**

Rappresentazione di un tipo astratto . . .

- Per poter operare effettivamente (automaticamente) su dati astratti è necessario determinare una loro rappresentazione nel contesto linguistico di un linguaggio di programmazione scelto
- I tipi astratti stanno ai tipi concreti, propri di un linguaggio di programmazione, come gli algoritmi stanno ai programmi in quel linguaggio di programmazione
  - Ad esempio in Java i tipi concreti sono i tipi primitivi e quelli riferimento (già disponibili come String e Array oppure definiti dall'utente come NodoLista)

. . . Rappresentazione di un tipo astratto

- **Per rappresentare un tipo astratto < S , F , C > è necessario rappresentare:**
  - **il dominio di interesse e ciascuno degli altri domini in S**
  - **ciascuna delle operazioni in F**
  - **ciascuna delle costanti in C**

Il tipo di dato astratto Pila

- **Tipo astratto Pila = <S, F, C> con**
  - **S = {P, V, boolean}**
    - **P = dominio d'interesse**
    - **V= dominio di sostegno (elementi della pila)**
  - **F = {Null, Push, Pop, Top}**
  - **C = {Empty}**

dove

- Empty: → P
  - **crea la pila vuota**
- Null: P → boolean
  - **test pila-vuota**
- Push: P x V → P
  - **Inserisce un elemento nella pila**
- Pop: P → P
  - **cancella da P l'elemento x in cima alla pila P e riporta x stesso**
- Top: P → V
  - **riporta l'elemento in cima alla pila S, senza modificare la pila**

Il tipo di dato astratto coda

- **Tipo astratto Coda = <S, F, C> con**
  - **S = {Q, V, boolean}**
    - **Q = dominio d'interesse**
    - **V= dominio di sostegno (elementi della coda)**
  - **F = {Null, Enqueue, Dequeue, Front}**
  - **C = {Empty}**

dove

- Empty: → Q
  - **crea la coda vuota**
- Null: Q → boolean
  - **test coda-vuota**
- Enqueue: Q x V → Q
  - **Inserisce un elemento nella coda**
- Dequeue: Q → Q
  - **cancella da Q l'elemento x in testa alla coda q e riporta x stesso**
- Front: Q → V
  - **riporta l'elemento in testa alla coda Q, senza modificare la coda**