Fondamenti d’Informatica: Funzioni ricorsive

Barbara Re, Phd

Calcolabilità secondo Church

- Approccio alla calcolabilità che **si basa sulla nozione di funzione ricorsiva**
  - Contemporaneo a quello di Turing
  - Formalmente diverso da quello di Turing
  - Completamente equivalente a quello di Turing

Funzione Ricorsiva

- La **ricorsione** è connessa alla nozione matematica di **induzione**
  - Un **insieme** è definito in modo induttivo se è definito in termini di se stesso
  - Una **funzione** è definita in modo induttivo se è definita in termini di se stessa

- La definizione di una funzione f si dice ricorsiva se il valore di f su alcuni argomenti dipende dal valore di f su altri argomenti, solitamente più "semplici" rispetto ad una opportuna metrica da definire
  - Nella definizione induttiva di una funzione è possibile identificare
    - Uno o più casi base
    - Uno o più casi induttivi

Esempi

- **Fattoriale**

$$\begin{cases} fatt(0) = 1 & \text{caso base} \\ fatt(n+1) = (n+1) * fatt(n) & \text{caso induttivo} \end{cases}$$

- **Fibonacci** (è una successione di numeri interi positivi in cui ciascun numero è la somma dei due precedenti e i primi due termini della successione sono dati per definizione)

$$\begin{cases} fibo(0) = 1 \\ fibo(1) = 1 \\ fibo(n+2) = fibo(n) + fibo(n+1) \end{cases}$$

Metodi ricorsivi

- La **ricorsione** è una tecnica di programmazione utile ed efficace nell'implementazione di operazioni (metodi) su insiemi e tipi di dato definiti in modo induttivo
- Un metodo ricorsivo è un metodo che, direttamente o indirettamente, può invocare se stesso

Definizione di metodi ricorsivi: esempio

```
/* Calcola il fattoriale del numero naturale n. */
public static int fattoriale (int n) {
    // pre: n>=0
    int nfatt;        // il fattoriale di n

    if (n==0)         // caso base
        nfatt = 1;
    else              // caso induttivo
        nfatt = n * fattoriale(n-1);
    return nfatt;
}
```

Funzione Primitiva Ricorsiva

Nella teoria della calcolabilità, **le funzioni ricorsive primitive** sono una classe di funzioni che possono essere definite **applicando un numero finito di volte la ricorsione** e la composizione a partire da particolari **funzioni base** (funzioni zero, funzione successore e funzioni selettive o proiettive) e costituiscono un passo fondamentale nella costruzione di una completa formalizzazione della calcolabilità

Le funzioni ricorsive primitive possono essere calcolate dalle macchine che terminano sempre, mentre le funzioni ricorsive richiedono sistemi con la stessa potenza di calcolo delle macchine di Turing.

Funzione Primitiva Ricorsiva

- Definiamo **primitiva ricorsiva** una funzione che o fa parte delle funzioni base, oppure può essere ottenuta a partire dalle funzioni base applicando la **composizione** e la **ricorsione primitiva** un numero finito di volte

- L'insieme delle funzioni ricorsive primitive è definito come il più piccolo insieme contenente le funzioni ricorsive di base e che sia chiuso per composizione e ricorsione primitiva

- Le funzioni ricorsive vanno dai naturali ai naturali: N → N
  - In generale prendono come argomento una tupla di n numeri naturali e restituiscono un numero naturale

Funzioni base

- Le funzioni **zero**, che prendono n argomenti (con $n \geq 0$), e che restituiscono sempre 0. Sono perciò funzioni costanti

$$Z(x_1, x_2, \ldots, x_n) = 0$$

- La funzione **successore** S, che prende un argomento e restituisce il numero successivo

$$S(x) = x + 1$$

- Le funzioni di **proiezione** (dette anche funzioni selettive o funzioni identità), che prendono n argomenti ($n \geq 1$) e restituiscono l'i-esimo tra essi ($1 \leq i \leq n$).

$$P_i^n(x_1, x_2, \ldots, x_n) = x_i$$

Operatori

- È possibile ottenere funzioni ricorsive primitive più complesse di quelle base, applicando a queste ultime i seguenti operatori:
  - Composizione
  - Ricorsione Primitiva

Operatore di composizione (o sostituzione)

Se f è una funzione a n argomenti, e $g_1, \ldots, g_n$ sono funzioni tutte a m argomenti, allora la funzione h a m argomenti:

$$h(x_1, \ldots, x_m) = f(g_1(x_1, \ldots, x_m), \ldots, g_n(x_1, \ldots, x_m))$$

è ottenuta per composizione (o sostituzione) da f e $g_1, \ldots, g_n$

Operatore di ricorsione primitiva

Se f è una funzione a n argomenti, e g è una funzione a n+2 argomenti, allora la funzione h a n+1 argomenti così definita:

$$h(x_1, ..., x_n, 0) = f(x_1, ..., x_n)$$

$$h(x_1, ..., x_n, y+1) = g(x_1, ..., x_n, y, h(x_1, ..., x_n, y))$$

è ottenuta per ricorsione primitiva da f e g

Cosa si può costruire!

- Addizione
- Moltiplicazione
- Fattoriale
- …

Progettazione di metodi ricorsivi

Linee guida per la progettazione di metodi ricorsivi

- Determina i casi base
- Determina i casi induttivi
- Verifica che i casi base e induttivi partizionano linsieme dei dati di ingresso per il metodo
- Usa una istruzione condizionale per selezionare qual è il caso corrente
- Può essere utile dichiarare una variabile per memorizzare il risultato del metodo
- Talvolta è necessario scrivere anche un metodo per avviare la ricorsione (mostrato più avanti)

Errori comuni

- Alcuni errori comuni nella scrittura di metodi ricorsivi
  - errori nella gestione dei casi base e induttivi
  - errori nella definizione del caso base
  - errori nella definizione del caso induttivo

Quando usare (o non usare) la ricorsione

- Per ogni metodo ricorsivo, è sempre possibile implementare un metodo iterativo che si comporta in modo equivalente
- In quali casi è opportuno risolvere un problema in modo ricorsivo e in quali casi è opportuno risolverlo in modo iterativo?
- **Alcuni criteri per scegliere:**
  - La ricorsione può essere usata in modo naturale nel calcolo di funzioni definite induttivamente e nell'elaborazione di dati di tipo ricorsivo e di insiemi definiti in modo induttivo
  - Tempo di esecuzione e occupazione di memoria
  - Costo (umano) della soluzione del problema