Funzioni e Calcolabilità

Sommario

- Riepilogo nozioni matematiche di base
- Il concetto di funzione
- Funzioni e algoritmi
- Decidibilità
- Funzioni parziali e calcolabilità

Funzione

- Una funzione è una corrispondenza tra due insiemi, che fa corrispondere ad ogni elemento del primo insieme (dominio) uno e un solo elemento del secondo (codominio)

$$f: D \rightarrow C$$

- In generale D e C sono diversi,
  - noi considereremo quasi sempre $D = C = \mathbb{N}$ = insieme dei naturali =$\{0, 1, 2, \ldots\}$ o un suo sottoinsieme

Argomenti e Valori

- L'elemento di D a cui la funzione f fa corrispondere l'elemento in C è detto argomento
- L'elemento in C corrispondente attraverso la funzione f a un argomento è detto valore

Esempi

- $f_1(x) = 2x$
- $f_2(x) = 3$
- $f_3(x) = \begin{cases} 0 & \text{se } x \text{ è pari} \\ 1 & \text{altrimenti} \end{cases}$

Funzioni a più argomenti (1)

- In generale, una funzione può avere più di un argomento
- Ad esempio la funzione sum seguente ha 3 argomenti (sum: $\mathbb{N}^3 \rightarrow \mathbb{N}$):

  $$\text{sum}(x_1, x_2, x_3) = x_1+x_2+x_3$$
- In questo caso il dominio è l'insieme delle terne ordinate su $\mathbb{N}$ ($\mathbb{N} \times \mathbb{N} \times \mathbb{N} = \mathbb{N}^3$)

Funzioni a più argomenti (2)

- Più in generale, un'espressione $f(x_1, x_2, \ldots, x_n)$ con n argomenti tale che,
  - se $x_1 \in D_1$, $x_2 \in D_2$, …, $x_n \in D_n$, allora $f(x_1, x_2, \ldots, x_n) \in C$

  individua una funzione $f: D_1 \times D_2 \times \ldots \times D_n \rightarrow C$ che ha come dominio l'insieme delle n-ple ordinate $D_1 \times D_2 \times \ldots \times D_n$ e come codominio C

Rango di una funzione

- Talvolta l'insieme di valori di una funzione (detto rango) coincide con tutto il codominio, talaltra con un suo sottoinsieme
  - Es: se $f: \mathbb{N} \rightarrow \mathbb{N}$, t.c. $f(x)= x^2$ il rango è il sottoinsieme di $\mathbb{N}$ costituito dai soli quadrati perfetti

Funzioni suriettive

- f: D → C è suriettiva sse il suo rango coincide con C
  - ∀ y∈ C esiste un x ∈ D t.c. f(x)=y
- Tutte le funzioni non suriettive possono essere trasformate in suriettive restringendo C al solo rango

Funzioni iniettive

- f: D → C è iniettiva sse a elementi distinti del dominio fa corrispondere elementi distinti del codominio
  - ∀ x_1, x_2∈ D, se $x_1 \neq x_2$ allora $f(x_1) \neq f(x_2)$

Funzioni biiettive

- f: D → C è biiettiva sse è sia suriettiva che iniettiva
  - Una funzione biiettiva è anche detta corrispondenza biunivoca

Funzioni inversa e identità

- Sia f: D → C una funzione biiettiva. La funzione inversa di f è la funzione f^{-1} : C→ D t.c. f^{-1} (y) = x sse f(x)=y
- Dato un qualsiasi insieme D, la funzione f: D→ D t.c. ∀x∈D f(x)=x è detta funzione identità
  - È indicata con i_D

Funzione composta

- Date due funzioni f: A → B e g: B → C, si può definire la funzione composta di f con g (in simboli g • f) come h: A → C t.c. ∀a∈A h(a)=g(f(a))

Funzioni e Algoritmi (1)

- Ogni algoritmo può essere assimilato a una funzione
  - I dati in input corrispondono agli argomenti
  - I dati in output ai valori
- Un algoritmo A calcola una funzione f: D→ C sse ∀x∈D f(x)=y sse A con input x produce come output y

Funzioni e Algoritmi (2)

- Una funzione è calcolabile (o computabile) sse esiste (almeno) un algoritmo che la calcola
- Una funzione è calcolabile sse esiste (almeno) una MdT che realizza il corrispondente algoritmo
- Esistono funzioni non calcolabili

Decidibilità

- Una proprietà di un dominio è decidibile se esiste un algoritmo che permette di stabilire se la proprietà sussiste per ogni elemento del dominio
  - Es: x è pari è decidibile in ℕ
- Una relazione tra due (o più) elementi in un insieme è decidibile se esiste un algoritmo per stabilire se la relazione sussiste tra i due (o più) elementi

Funzioni Totali e Parziali

- Le funzioni che associano a ogni elemento del Dominio un elemento del codominio sono dette totali
- Esistono funzioni parziali che associano un elemento del codominio solo a elementi in un sottoinsieme del dominio (detto campo di esistenza)
- Se D è il dominio e CE il campo di esistenza, allora ∀ x ∈ D\CE f(x)=⊥ (indefinito)

Calcolabilità di Funzioni Parziali

- Una funzione f:D→C parziale con campo di esistenza CE è computabile sse esiste un algoritmo A t.c.:
  - se x ∈ CE e f(x)=y allora A con input x produce l'output y
  - se x ∈ D\CE (cioè f(x)=⊥) allora A con input x non produce alcun output

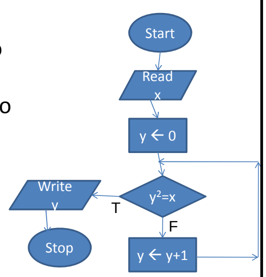
Esempio: Radice quadrata (1)

- La funzione f:ℕ→ℕ che calcola la radice quadrata di x è parziale:

$$f(x) = \begin{cases} \sqrt{x} & \text{se x è un quadrato} \\ \perp & \text{altrimenti} \end{cases}$$

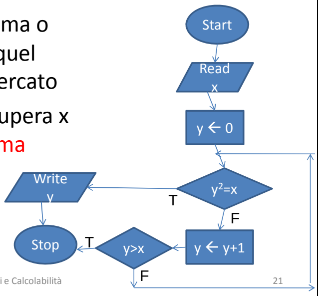
Esempio: Radice quadrata (2)

- Algoritmo 1
  - Si incrementa y
  - Se x è un quadrato, prima o poi y^2 diventa = a x, e quel valore di y è il valore cercato
  - Altrimenti la computazione prosegue all'infinito
- È diverso dalla figura riportata nel libro!!!



Esempio: Radice quadrata (3)

- Algoritmo 2
  - Si incrementa y
  - Se x è un quadrato, prima o poi y^2 diventa = a x, e quel valore di y è il valore cercato
  - Altrimenti, quando y supera x la computazione si ferma
- È diverso dalla figura riportata nel libro!!!



Esempio: Radice quadrata (4)

- Entrambi gli algoritmi:
  - sono poco efficienti
  - calcolano il valore della radice quadrata in numero finito di passi solo se il valore in input è un quadrato perfetto
- Se l'input non è un quadrato perfetto, allora hanno due comportamenti diversi:
  - uno continua a calcolare all'infinito
  - l'altro finisce la computazione dopo un numero finito di passi senza alcun messaggio

Generalizzazione dell'esempio (1)

- Se, per una funzione parziale f, è possibile scrivere un algoritmo che termina, allora è possibile estendere la funzione f ad una funzione totale f' che calcola un valore convenzionale (ad es 0) per gli argomenti per cui f è indefinita:

$$f'(x) = \begin{cases} f(x) \text{ se } x \in CE \\ 0 \text{ se } x\ D \setminus CE \end{cases}$$

Generalizzazione dell'esempio (2)

- Esistono funzioni parziali per le quali NON è possibile l'estensione a funzioni totali

Nota di Programmazione

- I moduli di Controllo dell'Input visti in precedenza hanno lo scopo di gestire questi eventi
  - Anziché usare il valore di output 0 si usano opportuni codici di errore