Informatica teorica

Indice

1. Introduzione

1.1. Definizione

L'**informatica teorica** è una branca dell'informatica e della matematica che si "contrappone" all'informatica applicata: infatti, in quest'ultima, l'informatica è usata solo come uno strumento per gestire l'oggetto del discorso

In questo caso, invece, vogliamo rendere l'informatica l'oggetto del discorso, studiandone i fondamenti

1.2. Cosa e come

Per studiare una nuova materia dobbiamo studiarne i *cosa* e i *come*:

- *cosa*: l'informatica è la scienza che studia l'informazione e la sua elaborazione automatica, fatta da un sistema di calcolo
  Ogni volta che ho un *problema* cerco di risolverlo automaticamente scrivendo un programma, ma posso farlo per ogni problema? Esistono problemi che non sono risolubili?
  Il *cosa* riguarda la **teoria della calcolabilità**, ovvero quella branca che studia tutto ciò che è calcolabile o meno, a prescindere dal costo in termini di risorse che ne deriva
  In questa parte utilizzeremo una caratterizzazione molto rigorosa e una definizione il più generale possibile di un problema, così che non dipenda da fattori tecnologici, storici, eccetera
- *come*: riguarda la **teoria della complessità**, ovvero quella branca che studia la quantità di risorse computazionali richieste nella soluzione automatica di un problema
  Con *risorsa computazionale* si intende qualsiasi cosa venga consumato durante l'esecuzione di un programma, ad esempio
  - elettricità
  - numero di processori
  - tempo
  - spazio di memoria

  In questa fase daremo una definizione rigorosa di tempo, spazio e di problema efficientemente risolubile in tempo e spazio, oltre che uno sguardo al famoso problema `P = NP`

In poche parole, il *cosa* fa uno studio **qualitativo**, il *come* fa uno studio **quantitativo**

Studieremo prima la teoria della calcolabilità per individuare le funzioni calcolabili, poi di queste ultime ne studieremo la complessità

2. Linguaggio matematico

2.1. Funzioni

Una **funzione** da un insieme A ad un insieme B è una *legge*, spesso indicata con f, che spiega come associare ad ogni elemento di A un elemento di B

Abbiamo due livelli di funzioni

- **generale**: la funzione è definita in modo generale come $f : A \longrightarrow B$, dove A è detto **dominio** di f e B è detto **codominio** di f
- **locale/puntuale**: la funzione riguarda i singoli valori a e b

$$f(a) = b \quad | \quad a \overset{f}{\longmapsto} b$$

  b è detta **immagine** di a rispetto ad f e a è detta **controimmagine** di b rispetto ad f

Possiamo categorizzare le funzioni in base ad alcune proprietà

- **iniettiva**: una funzione $f : A \longrightarrow B$ si dice *iniettiva* se e solo se

$$\forall a_1, a_2 \in A \quad a_1 \neq a_2 \Longrightarrow f(a_1) \neq f(a_2)$$

  In poche parole, non ho *confluenze*, ovvero *elementi diversi finiscono in elementi diversi*
- **suriettiva**: una funzione $f : A \longrightarrow B$ si dice *suriettiva* se e solo se

$$\forall b \in B \quad \exists a \in A \ | \ f(a) = b$$

  In poche parole, *ogni elemento del codominio ha almeno una controimmagine*

Se definiamo l'**insieme immagine**

$$\mathrm{Im}_f = \{b \in B \ | \ \exists a \in A \text{ tale che } f(a) = b\} = \{f(a) \ | \ a \in A\} \subseteq B$$

possiamo dare una definizione alternativa di funzione suriettiva, ovvero una funzione è *suriettiva* se e solo se $\mathrm{Im}_f = B$

L'ultima proprietà è l'intersezione delle due precedenti: una funzione $f : A \longrightarrow B$ si dice **biiettiva** se e solo se è iniettiva e suriettiva, ovvero

$$\forall b \in B \quad \exists! a \in A \ | \ f(a) = b$$

Se $f : A \longrightarrow B$ è una funzione biiettiva, si definisce **inversa** di f la funzione $f^{-1} : B \longrightarrow A$ tale che

$$f(a) = b \Longleftrightarrow f^{-1}(b) = a$$

La funzione f deve essere biiettiva: se così non fosse, la sua inversa avrebbe problemi di definizione

Un'operazione importante è la **composizione**: date $f : A \longrightarrow B$ e $g : B \longrightarrow C$, la funzione *f composto g* è la funzione $g \odot f : A \longrightarrow C$ definita come $(g \odot f)(a) = g(f(a))$

In generale, la composizione *non è commutativa*, quindi $g \odot f \neq f \odot g$ in generale, ma è *associativa*, quindi $(f \odot g) \odot h = f \odot (g \odot h)$

La composizione *f composto g* la possiamo leggere come *prima agisce f poi agisce g*

Una funzione molto importante è la **funzione identità**: dato l'insieme A, la funzione identità su A è la funzione $i_A : A \longrightarrow A$ tale che

$$i_A(a) = a \quad \forall a \in A,$$

ovvero è una funzione che mappa ogni elemento su se stesso

Grazie alla funzione identità possiamo dare una definizione alternativa di funzione inversa: data la funzione $f : A \longrightarrow B$ biiettiva, la sua inversa è l'unica funzione $f^{-1} : B \longrightarrow A$ che soddisfa

$$f \odot f^{-1} = f^{-1} \odot f = \mathrm{id}_A$$

Possiamo vedere f^{-1} come l'unica funzione che ci permette di *tornare indietro*