Informatica teorica

Indice

1. Lezione 01

1.1. Introduzione

1.1.1. Definizione

L'**informatica teorica** è quella branca dell'informatica che si "contrappone" all'informatica applicata: in quest'ultima, l'informatica è usata solo come uno *strumento* per gestire l'oggetto del discorso, mentre nella prima l'informatica diventa l'*oggetto* del discorso, di cui ne vengono studiati i fondamenti.

1.1.2. Cosa e come

Analizziamo i due aspetti fondamentali che caratterizzano ogni materia:

1. il **cosa**: l'informatica è la scienza che studia l'informazione e la sua elaborazione automatica mediante un sistema di calcolo. Ogni volta che ho un *problema* cerco di risolverlo automaticamente scrivendo un programma. *Posso farlo per ogni problema? Esistono problemi che non sono risolubili?* Possiamo chiamare questo primo aspetto con il nome di **teoria della calcolabilità**, quella branca che studia cosa è calcolabile e cosa non lo è, a prescindere dal costo in termini di risorse che ne deriva. In questa parte utilizzeremo una caratterizzazione molto rigorosa e una definizione di problema il più generale possibile, così che l'analisi non dipenda da fattori tecnologici, storici...
2. il **come**: è relazionato alla **teoria della complessità**, 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 parte 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`.

Possiamo dire che il *cosa* è uno studio **qualitativo**, mentre il *come* è uno studio **quantitativo**.

Grazie alla teoria della calcolabilità individueremo le funzioni calcolabili, di cui studieremo la complessità.

1.2. Richiami matematici

1.2.1. Funzioni

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

Abbiamo due tipi di funzioni:

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

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

  in cui 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:

- **iniettività**: 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.*

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

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

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

Se definiamo l'**insieme immagine**:

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

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

Infine, una funzione $f : A \longrightarrow B$ si dice **biiettiva** se e solo se è iniettiva e suriettiva, quindi vale: $\forall b \in B \quad \exists! a \in A \mid f(a) = b$.

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

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

Per definire la funzione inversa f^{-1}, la funzione f deve essere biiettiva: se così non fosse, la sua inversa avrebbe problemi di definizione.

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

La composizione *non è commutativa*, quindi $g \circ f \neq f \circ g$ in generale, ma è *associativa*, quindi $(f \circ g) \circ h = f \circ (g \circ h)$.

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

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:

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

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

2. Lezione 02

2.1. Richiami matematici

2.1.1. Funzioni totali e parziali

Prima di fare un'ulteriore classificazione per le funzioni, introduciamo la notazione $f(a)\downarrow$ per indicare che la funzione f è definita per l'input a, mentre la notazione $f(a)\uparrow$ per indicare la situazione opposta.

Ora, data $f : A \longrightarrow B$ diciamo che f è:
- **totale** se è definita *per ogni elemento* $a \in A$, ovvero $f(a)\downarrow \ \forall a \in A$;
- **parziale** se è definita *per qualche elemento* $a \in A$, ovvero $\exists a \in A \mid f(a)\uparrow$.

Chiamiamo **dominio** (o *campo*) **di esistenza** di f l'insieme

$$\mathrm{Dom}_f = \{a \in A \mid f(a)\downarrow\} \subseteq A.$$

Notiamo che:
- $\mathrm{Dom}_f \subsetneq A \Longrightarrow f$ parziale;
- $\mathrm{Dom}_f = A \Longrightarrow f$ totale.

2.1.2. Totalizzare una funzione

È possibile *totalizzazione* una funzione parziale definendo una funzione a tratti $f : A \longrightarrow B \cup \{\perp\}$ tale che

$$\overline{f}(a) = \begin{cases} f(a) & a \in \mathrm{Dom}_f(a) \\ \perp & \text{altrimenti} \end{cases}.$$

Il nuovo simbolo introdotto è il *simbolo di indefinito*, e viene utilizzato per tutti i valori per cui la funzione di partenza f non è appunto definita.

Da qui in avanti utilizzeremo $B_\perp$ come abbreviazione di $B \cup \{\perp\}$.

2.1.3. Prodotto cartesiano

Chiamiamo **prodotto cartesiano** l'insieme

$$A \times B = \{(a, b) \mid a \in A \wedge b \in B\},$$

che rappresenta l'insieme di tutte le *coppie ordinate* di valori in A e in B.

In generale, il prodotto cartesiano **non è commutativo**, a meno che $A = B$.

Possiamo estendere il concetto di prodotto cartesiano a n-uple di valori, ovvero

$$A_1 \times ... \times A_n = \{(a_1, ..., a_n) \mid a_i \in A_i\}.$$

L'operazione "*opposta*" è effettuata dal **proiettore** i-esimo: esso è una funzione che estrae l'i-esimo elemento di un tupla, ovvero è una funzione $\pi_i : A_1 \times ... \times A_n \longrightarrow A_i$ tale che

$$\pi_i(a_1, ..., a_n) = a_i$$

Per comodità chiameremo $\underbrace{A \times ... \times A}_{n} = A^n$

2.1.4. Insieme di funzioni

L'insieme di tutte le funzioni da A in B si indica con

$$B^A = \{f : A \longrightarrow B\}.$$

Viene utilizzata questa notazione in quanto la cardinalità di B^A é esattamente $|B|^{|A|}$, se A e B sono insiemi finiti.

In questo insieme sono presenti anche tutte le funzioni parziali da A in B: mettiamo in evidenza questa proprietà, scrivendo

$$B_\perp^A = \{f : A \to B_\perp\}.$$

Sembrano due insiemi diversi ma sono la stessa cosa: nel secondo viene messo in evidenza il fatto che tutte le funzioni che sono presenti sono totali oppure parziali che sono state totalizzate.

2.1.5. Funzione di valutazione

Dati A, B e $B_\perp^A$ si definisce **funzione di valutazione** la funzione

$$\omega : B_\perp^A \times A \longrightarrow B$$

tale che

$$w(f, a) = f(a).$$

In poche parole, è una funzione che prende una funzione f e la valuta su un elemento a del dominio.

Abbiamo due possibili approcci:
- tengo fisso a e provo tutte le funzioni f: sto eseguendo un *benchmark*, quest'ultimo rappresentato da a;
- tengo fissa f e provo tutte le a del dominio: sto ottenendo il grafico di f.

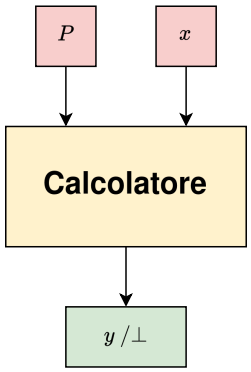
2.2. Teoria della calcolabilità

2.2.1. Sistema di calcolo

Quello che faremo ora è *modellare teoricamente un sistema di calcolo.*

Un **sistema di calcolo** lo possiamo vedere come una *black-box* che prende un programma P, una serie di dati x e calcola il risultato di P sull'input x.

La macchina ci restituisce:
- y se è riuscita a calcolare un risultato;
- $\perp$ se è entrata in un loop.



Un sistema di calcolo quindi è una funzione del tipo

$$\mathcal{C} : \text{PROG} \times \text{DATI} \longrightarrow \text{DATI}_\perp.$$

Come vediamo, è molto simile ad una funzione di valutazione: infatti, la parte dei dati x la riusciamo a "mappare" sull'input a del dominio, ma facciamo più fatica con l'insieme dei programmi, questo perché non abbiamo ancora definito cos'é un programma.

Un **programma** P è una **sequenza di regole** che trasformano un dato di input in uno di output (o in un loop): in poche parole, un programma è una funzione del tipo

$$P : \mathrm{DATI} \longrightarrow \mathrm{DATI}_\perp,$$

ovvero una funzione che appartiene all'insieme $\mathrm{DATI}_\perp^{\mathrm{DATI}}$.

Con questa definizione riusciamo a "mappare" l'insieme PROG sull'insieme delle funzioni che ci serviva per definire la funzione di valutazione.

Formalmente, un sistema di calcolo è quindi la funzione

$$\mathcal{C} : \mathrm{DATI}_\perp^{\mathrm{DATI}} \times \mathrm{DATI} \longrightarrow \mathrm{DATI}.$$

Con $\mathcal{C}(P, x)$ indichiamo la funzione calcolata da P su x, ovvero la sua **semantica**, il suo *significato*.

Il modello classico che viene considerato quando si parla di calcolatori è quello di **Von Neumann**.

2.2.2. Potenza computazionale

Prima di definire la potenza computazionale facciamo una breve premessa: indichiamo con

$$\mathcal{C}(P, \_) : \mathrm{DATI} \longrightarrow \mathrm{DATI}_\perp$$

la funzione che viene calcolata dal programma P, ovvero la semantica di P.

Fatta questa premessa, definiamo la **potenza computazionale** di un calcolatore come l'insieme di tutte le funzioni che quel sistema di calcolo può calcolare grazie ai programmi, ovvero

$$F(\mathcal{C}) = \{\mathcal{C}(P, \_) \mid P \in \mathrm{PROG}\} \subseteq \mathrm{DATI}_\perp^{\mathrm{DATI}}.$$

In poche parole, $F(\mathcal{C})$ rappresenta l'insieme di tutte le possibili semantiche di funzioni che sono calcolabili con il sistema $\mathcal{C}$.

Stabilire *cosa può fare l'informatica* equivale a studiare quest'ultima inclusione: in particolare, se
- $F(\mathcal{C}) \subsetneq \mathrm{DATI}_\perp^{\mathrm{DATI}}$ allora esistono compiti **non automatizzabili**;
- $F(\mathcal{C}) = \mathrm{DATI}_\perp^{\mathrm{DATI}}$ allora l'informatica può fare tutto.

Se ci trovassimo nella prima situazione dovremmo individuare una sorta di "recinto" per dividere le funzioni calcolabili da quelle che non lo sono.

Calcolare una funzione vuole dire *risolvere un problema*: ad ognuno di questi associo una **funzione soluzione**, che mi permette di risolvere in modo automatico il problema.

Grazie a questa definizione, calcolare le funzioni equivale a risolvere problemi.

In che modo possiamo risolvere l'inclusione?

Un primo approccio è quello della **cardinalità**: viene definita come una funzione che associa ad ogni insieme il numero di elementi che contiene.

Sembra un ottimo approccio, ma presenta alcuni problemi: infatti, funziona solo quando l'insieme è finito, mentre è molto fragile quando si parla di insiemi infiniti.

Ad esempio, gli insiemi
- $\mathbb{N}$ dei numeri naturali e
- $\mathbb{P}$ dei numeri naturali pari

sono tali che $\mathbb{P} \subsetneq \mathbb{N}$ ma hanno cardinalità $|\mathbb{N}| = |\mathbb{P}| = \infty$.

Dobbiamo dare una definizione diversa di cardinalità, visto che possono esistere "*infiniti più fitti/densi di altri*", come abbiamo visto nell'esempio precedente.

3. Lezione 03

3.1. Relazioni di equivalenza

3.1.1. Definizione

Una **relazione** R su due insiemi A, B è un sottoinsieme $R \subseteq A \times B$ di coppie ordinate. Una relazione particolare che ci interessa è quella **binaria**, ovvero $R \subseteq A^2$. Due elementi $a, b \in A$ sono in relazione R se e solo se $(a, b) \in R$. Indichiamo la relazione tra due elementi tramite notazione infissa aRb.

Una classe molto importante di relazioni é quella delle **relazioni di equivalenza**: una relazione $R \subseteq A^2$ é una relazione di equivalenza se e solo se R é RST, ovvero:
- **riflessiva**: $\forall a \in A \quad aRa$;
- **simmetrica**: $\forall a, b \in A \quad aRb \implies bRa$;
- **transitiva**: $\forall a, b, c \in A \quad aRb \land bRc \implies aRc$.

3.1.2. Classi di equivalenza e insieme quoziente

Ad ogni relazione di equivalenza viene associata una **partizione**: infatti, ogni relazione di equivalenza R su A^2 *induce* una partizione su A formata da $A_1, A_2, \dots$ sottoinsiemi tali che:
- $\forall i \geq 1 \quad A_1 \neq \varnothing$;
- $\forall i, j \geq 1 \quad i \neq j \implies A_i \cap A_j = \varnothing$;
- $\bigcup_{i \geq 1} A_i = A$.

Dato un elemento $a \in A$, la sua **classe di equivalenza** é l'insieme

$$[a]_R = \{b \in A \mid aRb\},$$

ovvero tutti gli elementi che sono in relazione con a, detto anche *rappresentante della classe*.

Si può dimostrare che:
- non esistono classi di equivalenza vuote, garantito dalla proprietà *riflessiva*;
- dati $a, b \in A$ allora $[a]_R \cap [b]_R = \varnothing$ oppure $[a]_R = [b]_R$;
- $\bigcup_{a \in A} [a]_R = A$.

Ma allora l'insieme delle classi di equivalenza é una partizione indotta dalla relazione R sull'insieme A. Questa partizione é detta **insieme quoziente** di A rispetto a R ed é denotato con A/R.

3.2. Cardinalità

3.2.1. Isomorfismi

Due insiemi A e B sono **isomorfi** se esiste una biiezione tra essi. Formalmente scriviamo

$$A \sim B.$$

Sinonimi di *isomorfi* sono *equinumerosi* o *insiemi che hanno la stessa cardinalità*.

Notiamo come $\sim$ sia una relazione: infatti, due insiemi appartengono alla relazione $\sim$ se e solo se sono isomorfi. Detto $\mathcal{U}$ l'insieme di tutti gli insiemi, la relazione $\sim$ é sottoinsieme di $\mathcal{U}^2$.

Dimostriamo che $\sim$ é una relazione di equivalenza:
- *riflessiva*: $A \sim A$ se la biiezione é i_A;
- *simmetrica*: $A \sim B \implies B \sim A$ se la biiezione é la funzione inversa usata per $A \sim B$;
- *transitiva*: $A \sim B \land B \sim C \implies A \sim C$ se la biiezione é la composizione della funzione usata per $A \sim B$ con la funzione usata per $B \sim C$.

Essendo $\sim$ una relazione di equivalenza posso partizionare l'insieme $\mathcal{U}$: la partizione creata contiene classi di equivalenza che contengono insiemi con la stessa cardinalità, ovvero isomorfi tra loro.

La **cardinalità** é questo: l'insieme quoziente di $\mathcal{U}$ rispetto alla relazione $\sim$.

La comodità di questo approccio é che possiamo utilizzare la nozione di *cardinalità* anche con gli insiemi infiniti.

3.2.2. Cardinalità finita

La prima classe di cardinalità che vediamo é quella delle **cardinalità finite**: prima di tutto definiamo la famiglia di insiemi

$$J_n = \begin{cases} \varnothing & \text{se } n = 0 \\ \{1, ..., n\} & \text{se } n > 0 \end{cases}.$$

Un insieme A ha cardinalità finita se $A \sim J_n$ per qualche $n \in \mathbb{N}$, e in tal caso scriviamo "tranquillamente" $|A| = n$.

La classe di equivalenza $[J_n]_\sim$ riunisce tutti gli insiemi di $\mathcal{U}$ contenenti n elementi.

3.2.3. Cardinalità infinita

Un insieme che non sia finito, ovvero non in relazione con J_n, si dice a **cardinalità infinita**.

3.2.3.1. Insiemi numerabili

La prima classe di insiemi a cardinalità infinita é quella degli **insiemi numerabili**: un insieme A é numerabile se e solo se $\mathbb{N} \sim A$, ovvero $A \in [\mathbb{N}]_\sim$.

Gli insiemi numerabili sono "**listabili**", ovvero posso elencare *tutti* gli elementi dell'insieme A tramite una regola, che in questo caso é la funzione f biiezione tra $\mathbb{N}$ e A.

Infatti, grazie alla funzione f posso elencare gli elementi di A formando l'insieme

$$A = \{f(0),\ f(1),\ ...\}.$$

Questo insieme é *esaustivo*, ovvero elenco ogni elemento dell'insieme A senza perderne alcuno.

Gli insiemi numerabili più famosi sono:
- numeri pari $\mathbb{P}$ e numeri dispari $\mathbb{D}$;
- numeri interi $\mathbb{Z}$ generati con la biiezione $f(n) = (-1)^n \left(\frac{n + (n \bmod 2)}{2} \right)$;
- numeri razionali $\mathbb{Q}$.

3.2.3.2. Insiemi non numerabili

La classe degli **insiemi non numerabili** raccoglie gli insiemi a cardinalità infinita ma che non sono listabili come gli insiemi numerabili, ovvero sono "più fitti" di $\mathbb{N}$.

Il *non poter listare gli elementi* si traduce in *qualunque lista generata mancherebbe di qualche elemento*, ovvero non é una lista esaustiva di tutti gli elementi presenti nell'insieme.

Il più famoso insieme non numerabile é l'insieme dei numeri reali $\mathbb{R}$.

In generale, qualsiasi *insieme continuo* é un insieme non numerabile.

Teorema 3.2.3.2.1 L'insieme $\mathbb{R}$ non é numerabile

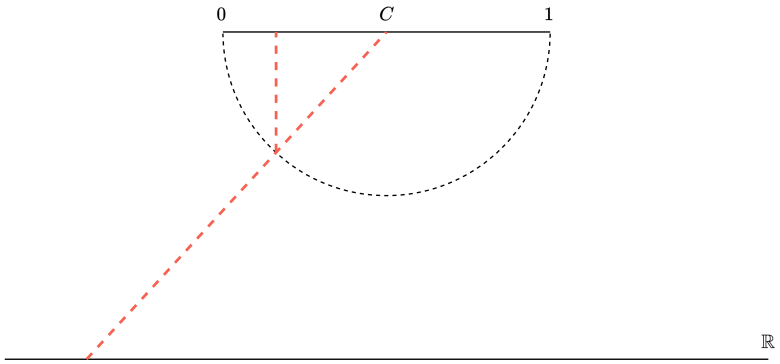
Dimostrazione
Suddividiamo la dimostrazione in tre punti:

1. dimostriamo che $\mathbb{R} \sim (0,1)$;
2. dimostriamo che $\mathbb{N} \nsim (0,1)$;
3. dimostriamo che $\mathbb{R} \nsim \mathbb{N}$.

[1] Partiamo con il dimostrare che $\mathbb{R} \sim (0,1)$: mostro che esiste una biiezione tra $\mathbb{R}$ e $(0,1)$. Usiamo come biiezione "grafica" quella che:

- disegna la circonferenza di raggio $\frac{1}{2}$ centrata in $\frac{1}{2}$;
- disegna la perpendicolare al punto da mappare che interseca la circonferenza;
- disegna la retta passante per il centro C e l'intersezione precedente.

L'intersezione tra l'asse reale e la retta finale é il punto mappato.



In realtà, $\mathbb{R}$ é isomorfo a qualsiasi segmento di lunghezza maggiore di 0.

La stessa biiezione vale anche sull'intervallo chiuso $[0,1]$ utilizzando la "compattificazione" $\mathring{\mathbb{R}} = \mathbb{R} \cup \{\pm\infty\}$ di $\mathbb{R}$, mappando 0 su $-\infty$ e 1 su $+\infty$.

[2] Continuiamo dimostrando che $\mathbb{N} \nsim (0,1)$: ...

[3] Terminiamo dimostrando che $\mathbb{R} \nsim \mathbb{N}$: ... □