

Informatica Teorica

Massimo Perego

Contents

Introduzione	2
1 Teoria della Calcolabilità	4
1.1 Notazione	4
1.1.1 Funzioni	4
1.1.2 Prodotto Cartesiano	6
1.1.3 Funzione di Valutazione	7
1.2 Sistemi di Calcolo	7
1.3 Potenza Computazionale	8
1.4 Relazioni di Equivalenza	9
1.4.1 Partizione indotta dalla relazione di equivalenza	9
1.4.2 Classi di equivalenza e Insieme quoziente	9
1.5 Cardinalità	10
1.5.1 Isomorfismi	10
1.5.2 Cardinalità finita	11
1.5.3 Cardinalità infinita	11
1.5.4 Insieme delle Parti	13

Introduzione

Si “contrappone” all’informatica applicata, ovvero qualsiasi applicazione dell’informatica atta a raggiungere uno scopo, dove l’informatica è solamente lo strumento per raggiungere in maniera efficace un obiettivo.

Con “*informatica teorica*” l’oggetto è l’informatica stessa, si studiano i fondamenti della disciplina in modo rigoroso e scientifico. Può essere fatto ponendosi delle questioni fondamentali: il *cosa* e il *come* dell’informatica, ovvero cosa è in grado di fare l’informatica e come è in grado di farlo.

Cosa: L’informatica è “la disciplina che studia l’informazione e la sua elaborazione automatica”, quindi l’oggetto sono l’informazione e i dispositivi di calcolo per gestirla; scienza dell’informazione. Diventa lo studio come risolvere automaticamente un problema. Ma tutti i problemi sono risolvibili in maniera automatica? Cosa è in grado di fare l’informatica?

La branca dell’informatica teorica che studia cosa è risolvibile si chiama **Teoria della Calcolabilità**, studia cosa è calcolabile per via automatica. Spoiler: non tutti i problemi sono risolvibili per via automatica, e non potranno mai esserlo per limiti dell’informatica stessa. Cerchiamo una caratterizzazione generale di cosa è calcolabile e cosa no, si vogliono fornire strumenti per capire ciò che è calcolabile. La caratterizzazione deve essere fatta matematicamente, in quanto il rigore e la tecnica matematica permettono di trarre conclusioni sull’informatica.

Come: Una volta individuati i problemi calcolabili, come possiamo calcolarli? Il dominio della **Teoria della Complessità** vuole descrivere le risoluzioni dei problemi tramite mezzi automatici in termini di risorse computazionali necessarie. Una “risorsa computazionale” è qualsiasi cosa che viene consumata durante l’esecuzione per risolvere il problema, come pos-

sono essere elettricità o numero di processori, generalmente i parametri più importanti considerati sono tempo e spazio di memoria. Bisognerà definire in modo preciso cosa si intende con “tempo” e “spazio”. Una volta fissati i parametri bisogna definire anche cosa si intende con “risolvere efficientemente” un problema, in termini di tempo e spazio.

La teoria della calcolabilità dice quali problemi sono calcolabili, la teoria della complessità dice, all'interno dei problemi calcolabili, quali sono risolvibili efficientemente.

Capitolo 1

Teoria della Calcolabilità

1.1 Notazione

1.1.1 Funzioni

Funzione: Una funzione f dall'insieme A all'insieme B è una legge che dice come associare a ogni elemento di A un elemento di B . Si scrive

$$f : A \rightarrow B$$

E chiamiamo A dominio e B codominio. Per dire come agisce su un elemento si usa $f(a) = b$, b è l'immagine di a secondo f (di conseguenza a è la controimmagine).

Per definizione di funzione, è possibile che elementi del codominio siano raggiungibili da più elementi del dominio, ma non il contrario. Possiamo classificare le funzioni in base a questa caratteristica:

- **Iniettiva:** $f : A \rightarrow B$ è iniettiva sse $\forall a, b \in A, a \neq b \implies f(a) \neq f(b)$
- **Suriettiva:** $f : A \rightarrow B$ è suriettiva sse $\forall b \in B, \exists a \in A : f(a) = b$: un altro modo per definirla è tramite l'insieme immagine di f , definito come

$$\text{Im}_f = \{b \in B : \exists a, f(a) = b\} = \{f(a) : a \in A\}$$

Solitamente $\text{Im}_f \subseteq B$, ma f è suriettiva sse $\text{Im}_f = B$;

- **Biiettiva:** $f : A \rightarrow B$ è biiettiva sse è sia iniettiva che suriettiva, ovvero

$$\begin{aligned} \forall a, b \in A, a \neq b : f(a) \neq f(b) \\ \forall b \in B, \exists a \in A : f(a) = b \end{aligned} \implies \forall b \in B, \exists! a \in A : f(a) = b$$

Inversa: Per le funzioni biettive si può naturalmente associare il concetto di “inversa”: dato $f : A \rightarrow B$ biettiva, si definisce inversa la funzione $f^{-1} : B \rightarrow A$ tale che $f^{-1}(b) = a \Leftrightarrow f(a) = b$.

Composizione di funzioni: Date $f : A \rightarrow B$ e $g : B \rightarrow C$, f composto g è la funzione $g \circ f : A \rightarrow C$ definita come $g \circ f(a) = g(f(a))$. Generalmente non commutativo, $f \circ g \neq g \circ f$, ma è associativo.

Funzione identità: Dato l'insieme A , la funzione identità su A è la funzione $i_A : A \rightarrow A$ tale che $i_A(a) = a, \forall a \in A$.

Un'altra possibile definizione per l'inversa diventa:

$$f^{-1} \circ f = i_A \wedge f \circ f^{-1} = i_B$$

Funzioni Parziali: Se una funzione $f : A \rightarrow B$ è definita per $a \in A$ si indica con $f(a) \downarrow$ e da questo proviene la categorizzazione: una funzione è **totale** se definita $\forall a \in A$, **parziale** altrimenti (definita solo per qualche elemento di A).

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

$$\text{Dom}_f = \{a \in A | f(a) \downarrow\} \subseteq A$$

Quindi se $\text{Dom}_f = A$ la funzione è totale, se $\text{Dom}_f \subsetneq A$ allora è una funzione parziale.

Totalizzazione: Si può **totalizzare una funzione parziale** f definendo una funzione a tratti $\bar{f} : A \rightarrow B \cup \{\perp\}$ tale che

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

Dove \perp è il **simbolo di indefinito**, per tutti i valori per cui la funzione di partenza f non è definita. Da qui in poi B_\perp significa $B \cup \{\perp\}$.

Insieme delle funzioni: L'insieme di tutte le funzioni che vanno da A a B si denota con

$$B^A = \{f : A \rightarrow B\}$$

La notazione viene usata in quanto la cardinalità di B^A è esattamente $|B|^{|A|}$, con A e B insiemi finiti.

Volendo includere anche tutte le funzioni parziali:

$$B_{\perp}^A = \{f : A \rightarrow B_{\perp}\}$$

Le due definizioni coincidono, $B^A = B_{\perp}^A$, ma quest'ultima permette di mettere in evidenza che tutte le funzioni presenti sono totali o totalizzate.

1.1.2 Prodotto Cartesiano

Chiamiamo **prodotto cartesiano** l'insieme

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

Rappresenta l'insieme di tutte le coppie ordinate di valori in A e B . In generale non è commutativo, a meno che $A = B$.

Può essere esteso a n -uple di valori:

$$A_1 \times \cdots \times A_n = \{(a_1, \dots, a_n) | a_i \in A_i\}$$

Il prodotto di n volte lo stesso insieme verrà, per comodità, indicato come

$$A \times \cdots \times A = A^n$$

Proiettore: Operazione “opposta”, il proiettore i -esimo è una funzione che estrae l' i -esimo elemento di una tupla, quindi è una funzione

$$\pi_i : A_1 \times \cdots \times A_n \rightarrow A_i \quad \text{t.c.} \quad \pi_i(a_1, \dots, a_n) = a_i$$

La proiezione sull'asse in cui sono presenti i valori dell'insieme a_i .

1.1.3 Funzione di Valutazione

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

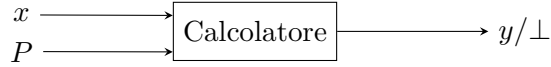
$$\omega : B_{\perp}^A \times A \rightarrow B \quad \text{t.c.} \quad \omega(f, a) = f(a)$$

Prende una funzione f e la valuta su un elemento a del dominio. Si possono fare due tipi di analisi su questa funzione:

- Fisso a e provo tutte le f , ottenendo un *benchmark* di tutte le funzioni su a
- Fisso f e provo tutte le a del dominio, ottenendo il *grafico* di f

1.2 Sistemi di Calcolo

Vogliamo modellare teoricamente un **sistema di calcolo**; quest'ultimo può essere visto come una black box che prende in input un programma P , dei dati x e calcola il risultato y di P su input x . La macchina restituisce y se è riuscita a calcolare un risultato, \perp (indefinito) se è entrata in un loop.



Quindi, formalmente, possiamo definire un sistema di calcolo come una funzione

$$C : \text{PROG} \times \text{DATI} \rightarrow \text{DATI}_{\perp}$$

Possiamo vedere un sistema di calcolo come una funzione di valutazione:

- i dati x corrispondono all'input a
- il programma P corrisponde alla funzione f

Formalmente, un programma $P \in \text{PROG}$ è una sequenza di regole che trasformano un dato input in uno di output, ovvero l'espressione di una funzione secondo una sintassi

$$P : \text{DATI} \rightarrow \text{DATI}_{\perp}$$

e di conseguenza $P \in \text{DATI}_{\perp}^{\text{DATI}}$. In questo modo abbiamo mappato l'insieme PROG sull'insieme delle funzioni, il che ci permette di definire il sistema di calcolo come la funzione

$$C : \text{DATI}_{\perp}^{\text{DATI}} \times \text{DATI} \rightarrow \text{DATI}$$

Analoga alla funzione di valutazione. Con $\mathcal{C}(P, x)$ indichiamo la funzione calcolata da P su x dal sistema di calcolo \mathcal{C} , che viene detta **semantica**, ovvero il suo “significato” su input x .

Il modello solitamente considerato quando si parla di calcolatori è quello di **Von Neumann**.

1.3 Potenza Computazionale

Indicando con

$$\mathcal{C}(P, _) : \text{DATI} \rightarrow \text{DATI}$$

la funzione che viene calcolata dal programma P (semantica di P).

La **potenza computazionale** di un calcolatore è definita come l’insieme di tutte le funzioni che quel sistema di calcolo è in grado di calcolare, ovvero

$$F(\mathcal{C}) = \{\mathcal{C}(P, _) | P \in \text{PROG}\} \subseteq \text{DATI}_{\perp}^{\text{DATI}}$$

Ovvero, l’insieme di tutte le possibili semantiche di funzioni calcolabili con il sistema \mathcal{C} . Stabilire il carattere di quest’ultima inclusione equivale a stabilire *cosa può fare l’informatica*:

- se $F(\mathcal{C}) \subsetneq \text{DATI}_{\perp}^{\text{DATI}}$ allora esistono compiti **non automatizzabili**
- se $F(\mathcal{C}) = \text{DATI}_{\perp}^{\text{DATI}}$ allora l’informatica *può fare tutto*

Calcolare funzioni vuol dire risolvere problemi *in generale*, a ogni problema è possibile associare una funzione soluzione che permette di risolverlo automaticamente.

Un possibile approccio per risolvere l’inclusione è tramite la **cardinalità** (funzione che associa ogni insieme al numero di elementi che contiene) dei due insiemi. Potrebbe però presentare dei problemi: è efficace solo quando si parla di insiemi finiti. Ad esempio, l’insieme dei numeri naturali contiene l’insieme dei numeri pari $\mathbb{P} \subsetneq \mathbb{N}$, ma $|\mathbb{N}| = |\mathbb{P}| = \infty$.

Serve una diversa definizione di cardinalità che considera l’esistenza di infiniti *più densi di altri*.

1.4 Relazioni di Equivalenza

Dati due insiemi A, B , una relazione binaria R è un sottoinsieme $R \subseteq A \times B$ di coppie ordinate. Data $R \subseteq A^2$, due elementi sono in relazione sse $(a, b) \in R$. Indichiamo la relazione tra due elementi anche con la notazione infissa aRb .

Una classe importante di relazioni è quella delle **relazioni di equivalenza**: una relazione $R \subseteq A^2$ è una relazione di equivalenza sse rispetta le proprietà di

- riflessività: $\forall a \in A, (a, a) \in R$
- simmetria: $\forall a, b \in A, (a, b) \in R \Leftrightarrow (b, a) \in R$
- transitività: $\forall a, b, c \in A, (a, b) \in R \wedge (b, c) \in R \implies (a, c) \in R$

1.4.1 Partizione indotta dalla relazione di equivalenza

A ogni relazione di equivalenza $R \subseteq A^2$ si può associare una **partizione**, ovvero un insieme di sottoinsiemi $A_i \subseteq A$ tali che

- $\forall i \in \mathbb{N}^+, A_i \neq \emptyset$
- $\forall i, j \in \mathbb{N}^+, \text{ se } i \neq j \text{ allora } A_i \cap A_j = \emptyset$
- $\bigcup_{i \in \mathbb{N}^+} A_i = A$

La relazione R definita su A^2 induce una partizione $\{A_1, A_2, \dots\}$ su A .

1.4.2 Classi di equivalenza e Insieme quoziente

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

$$[a]_R = \{b \in A \mid (a, b) \in R\}$$

Ovvero, tutti gli elementi in relazione con a , chiamato **rappresentante** della classe.

Si può dimostrare che

- non esistono classi di equivalenza vuote, per riflessività

- dati $a, b \in A$, allora $[a]_R \cap [b]_R = \emptyset$, oppure $[a]_R = [b]_R$, i due elementi o sono in relazione o non lo sono
- $\bigcup_{a \in A} [a]_R = A$

L'insieme delle classi di equivalenza, per definizione, è una partizione indotta da R su A , detta **insieme quoziente** di A rispetto ad R , denotato con A/R .

1.5 Cardinalità

1.5.1 Isomorfismi

Due insiemi A e B sono **isomorfi** (*equi-numerosi*) se esiste una biezione tra essi, denotato come $A \sim B$. Chiamando \mathcal{U} l'insieme di tutti gli insiemi, la relazione \sim è $\sim \subseteq \mathcal{U}^2$.

Dimostriamo che \sim è una relazione di equivalenza:

- riflessività: $A \sim A$, la biezione è data dalla funzione identità i_A
- simmetria: $A \sim B \Leftrightarrow B \sim A$, la biezione è data dalla funzione inversa
- transitività: $A \sim B \wedge B \sim C \implies A \sim C$, la biezione è data dalla composizione delle funzioni usate per $A \sim B$ e $B \sim C$

Dato che \sim è una relazione di equivalenza, permette di partizionare l'insieme \mathcal{U} , risultando in classi di equivalenza contenenti insiemi isomorfi, ovvero con la stessa cardinalità. Possiamo quindi definire la **cardinalità** come l'insieme quoziente di \mathcal{U} rispetto alla relazione \sim .

Questo approccio permette il *confronto delle cardinalità di insiemi infiniti*, basta trovare una funzione biettiva tra i due insiemi per poter affermare che sono isomorfi.

1.5.2 Cardinalità finita

La prima classe di cardinalità è quella delle cardinalità finite. Definiamo la seguente famiglia di insiemi:

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

Un insieme A ha **cardinalità finita** sse $A \sim J_n$ per qualche $n \in \mathbb{N}$; in tal caso possiamo scrivere $|A| = n$. La classe di equivalenza $[J_n]_{\sim}$ identifica tutti gli insiemi di \mathcal{U} contenenti n elementi.

1.5.3 Cardinalità infinita

L'altra classe di cardinalità è quella delle **cardinalità infinite**, ovvero gli insiemi non in relazione con J_n . Si possono dividere in **numerabili** e **non numerabili**.

Insiemi numerabili

Un insieme A è numerabile sse $A \sim \mathbb{N}$, ovvero $A \in [\mathbb{N}]_{\sim}$. Vengono anche detti **listabili**, in quanto è possibile elencare tutti gli elementi dell'insieme A tramite una funzione f biettiva tra \mathbb{N} e A ; grazie ad f possiamo elencare gli elementi di A , formando l'insieme

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

Ed è esaustivo, in quanto elenca tutti gli elementi di A .

Questi insiemi hanno cardinalità \aleph_0 (*aleph*).

Insiemi non numerabili

Gli insiemi non numerabili sono insiemi a cardinalità infinita ma non listabili, sono “più fitti” di \mathbb{N} ; ogni lista generata non può essere esaustiva.

Il più noto tra gli insiemi non numerabili è l'insieme \mathbb{R} dei numeri reali.

Teorema 1.5.1. *L'insieme \mathbb{R} non è numerabile ($\mathbb{R} \not\sim \mathbb{N}$)*

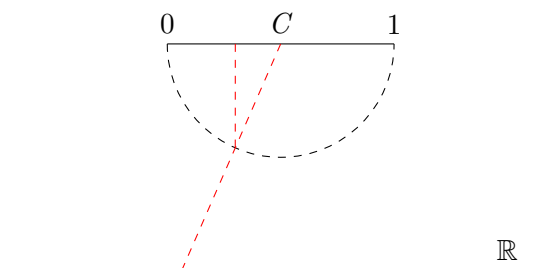
Dimostrazione. Suddividiamo la dimostrazione in 3 punti:

1. dimostriamo che $\mathbb{R} \sim (0, 1)$
2. dimostriamo che $\mathbb{N} \not\sim (0, 1)$
3. dimostriamo che $\mathbb{R} \not\sim \mathbb{N}$

Per dimostrare che $\mathbb{R} \sim (0, 1)$ serve trovare una biezione tra \mathbb{R} e $(0, 1)$. Usiamo una rappresentazione grafica:

- disegnare una semicirconferenza di raggio $1/2$, centrata in $1/2$, quindi con diametro 1
- disegnare la perpendicolare al punto da mappare che interseca la circonferenza
- disegnare la semiretta passante per il centro C e l'intersezione precedente

L'intersezione tra asse reale (parallela al diametro) e semiretta finale è il punto mappato.



Questo approccio permette di dire che \mathbb{R} è isomorfo a qualsiasi segmento di lunghezza maggiore di 0. La stessa biezione vale anche sull'intervallo chiuso $[0, 1]$ (e di conseguenza qualsiasi intervallo chiuso), usando la “compattificazione” $\mathbb{R} = \mathbb{R} \cup \{\pm\infty\}$ e mappando 0 su $-\infty$ e 1 su $+\infty$.

Continuiamo dimostrando che $\mathbb{N} \not\sim (0, 1)$: serve dimostrare che l'intervallo $(0, 1)$ non è listabile, quindi che ogni lista manca di almeno un elemento. Proviamo a “costruire” un elemento che andrà a mancare. Per assurdo, sia $\mathbb{N} \sim (0, 1)$, allora possiamo listare gli elementi di $(0, 1)$ come

$$\begin{array}{cccc} 0. & a_{00} & a_{01} & a_{02} & \dots \\ 0. & a_{10} & a_{11} & a_{12} & \dots \\ 0. & a_{20} & a_{21} & a_{22} & \dots \\ 0. & & & & \dots \end{array}$$

dove con a_{ij} indichiamo la cifra di posto j dell' i -esimo elemento della lista.

Costruiamo il numero $c = 0.c_0c_1\dots$ tale che

$$c_i = \begin{cases} 2 & \text{se } a_{ii} \neq 2 \\ 3 & \text{se } a_{ii} = 2 \end{cases}$$

Viene costruito “guardando” le cifre sulla diagonale principale, apparterrà sicuramente a $(0, 1)$ ma differirà per almeno una posizione (quella sulla diagonale principale) da ogni numero presente all'interno della lista. Questo è assurdo sotto l'assunzione che $(0, 1)$ è numerabile, quindi abbiamo provato che $\mathbb{N} \not\sim (0, 1)$.

Il terzo punto $\mathbb{R} \not\sim \mathbb{N}$ si dimostra per transitività.

Più in generale, non si riesce a listare nessun segmento di lunghezza maggiore di 0.

□

Questa dimostrazione (punto 2 in particolare) è detta **dimostrazione per diagonalizzazione**.

L'insieme \mathbb{R} viene detto **insieme continuo** e tutti gli insiemi isomorfi a \mathbb{R} si dicono continui a loro volta.

Gli insiemi continui hanno cardinalità \aleph_1 .

1.5.4 Insieme delle Parti