Fondamenti di informatica

UniVR - Dipartimento di Informatica

Fabio Irimie

1° Semestre 2025/2026

Indice

1 Introduzione

1.1 Cos'è l'informatica?

È una scienza che studia la calcolabilità, cioè cerca di capire che problemi si possono risolvere con un programma. Nasce dall'unione di matematica, ingegneria e logica. Il computer è solo uno strumento, mentre la matematica è il linguaggio con cui si creano algoritmi che permettono di risolvere i problemi.

1.2 Origini dell'informatica

Hilbert, nel 1900, si pose l'obiettivo di formalizzare tutta la matematica con un insieme finito e non contraddittorio di assiomi. Nel 1931, invece, Gödel dimostrò che l'informatica non potrà mai rappresentare tutta la matematica, perché ci saranno sempre proposizioni vere ma non dimostrabili tramite il calcolo. Ci si iniziò a chiedere se esistessero modelli di calcolo meccanici in grado di risolvere tutti i problemi. Nel 1936, Turing propose la macchina di Turing, una **sola** macchina programmabile in grado di risolvere tutti i problemi risolvibili:

$$\mathrm{Int}(P, x) = \begin{cases} P(x) & \text{se } P(x) \text{ termina} \\ \uparrow & \text{se } P(x) \text{ non termina} \end{cases}$$

dove P è un programma e x è un input. La macchina di Turing è un modello teorico di calcolatore, che non esiste fisicamente, ma è in grado di simulare qualsiasi altro calcolatore. Da questo modello deriva la concezione di calcolabilità, cioè se un problema è intuitivamente calcolabile, allora esiste un programma in grado di risolverlo.

Altri modelli di calcolo che sono stati proposti sono:

- Lambda-calcolo
- Funzioni ricorsive
- Linguaggi di programmazione (Turing-completi)

Definizione utile 1.1. La Turing-completezza è la proprietà di un linguaggio di programmazione di essere in grado di simulare una macchina di Turing, cioè di poter risolvere qualsiasi problema risolvibile.

1.2.1 Calcolabilità

Un programma è calcolabile se termina, ma non è detto che termini in un tempo ragionevole. Non esistono algoritmi che possono dire se un programma termina o meno. Questo è un esempio di problema non calcolabile.

I problemi non calcolabili sono infinitamente più numerosi di quelli calcolabili

2 Funzioni calcolabili

Un insieme è una proprietà ed è rappresentato da una funzione che indica se un elemento appartiene o meno all'insieme. I problemi da risolvere (in questo corso) hanno come soluzione una funzione sui naturali:

$$f : \mathbb{N} \to \mathbb{N}$$

Questo tupo di funzione è un **insieme** di associazioni input-output. Un esempio è la funzione quadrato:

$$f = \text{quadrato} = \{(0,0),(1,1),(2,4),(3,9),\ldots\} = \{(n, n^2) | n \in \mathbb{N}\}$$

Quindi f è un insieme di coppie in $\mathbb{N} \subseteq \mathbb{N} \times \mathbb{N}$ la cui cardinalità è: $|\mathbb{N} \times \mathbb{N}| = |\mathbb{N}|$. Di conseguenza la funzione è un sottoinsieme di $\mathbb{N} \times \mathbb{N}$:

$$f \subseteq \mathbb{N} \times \mathbb{N} \quad f \in \mathcal{P}(\mathbb{N} \times \mathbb{N})$$

Dove $\mathcal{P}(\mathbb{N} \times \mathbb{N})$ è l'insieme delle parti di $\mathbb{N} \times \mathbb{N}$, cioè l'insieme di tutti i sottoinsiemi di $\mathbb{N} \times \mathbb{N}$.

Il numero di funzioni è:

$$f : \mathbb{N} \to \mathbb{N} = |\mathcal{P}(\mathbb{N} \times \mathbb{N})| = |\mathcal{P}(\mathbb{N})|$$

Esempio 2.1. Un esempio di insieme delle parti per l'insieme $A = \{1,2,3\}$ è:

$$\mathcal{P}(A) = \{\emptyset, \{1\}, \{2\}, \{3\}, \{1,2\}, \{1,3\}, \{2,3\}, \{1,2,3\}\}$$

E le cardinalità sono:

$$|A| = 3 \quad |\mathcal{P}(A)| = 8 = 2^3$$

$$\downarrow$$

$$|\mathbb{N}| = \omega < \underbrace{|\mathcal{P}(\mathbb{N})| = 2^{\omega}}_{\substack{\text{Insieme delle funzioni,} \\ \text{non numerabile}}} = |\mathbb{R}|$$

Si ha quindi che **l'insieme delle funzioni non è numerabile**

Ci si chiede se queste funzioni sono tutte calcolabili:

Definizione 2.1. Una funzione **intuitivamente calcolabile** è una funzione descrivibile attraverso un algoritmo, cioè una sequenza finita di passi discreti elementari.

2.1 Quante funzioni numerabili ci sono?

Consideriamo Σ come un alfabeto finito, cioè una seguenza di simboli utilizzabili per scrivere un programma o algoritmo.

$$\Sigma = \{s_1, s_2, s_3, \ldots, s_n\}$$

$$\downarrow$$

$$\text{Programma} \subseteq \text{sqeuenze di simboli in } \Sigma$$

Con Σ^* si descrive l'insieme di tutte le sequenze finite di simboli in Σ, quindi l'insieme di tutti i possibili programmi è:

$$\text{Programmi} \subseteq \Sigma^*$$

Esempio 2.2. Se $\Sigma = \{a, b, c\}$ allora la sequenza di tutti i possibili simboli è:

$$\Sigma^* = \{\epsilon, a, b, c, aa, ab, ac, ba, bb, bc, ca, cb, cc, aaa, aab, \ldots\}$$

dove ϵ è la stringa vuota.

La cardinalità di Σ^* è infinita numerabile (anche se Σ è finito).

$$|\Sigma^*| = |\mathbb{N}|$$

Si ha quindi che l'insieme dei programmi è numerabile:

$$|\text{Programmi in } \Sigma| \leqslant |\Sigma^*| = |\mathbb{N}|$$

e questo implica che l'insieme delle **funzioni calcolabili è numerabile**

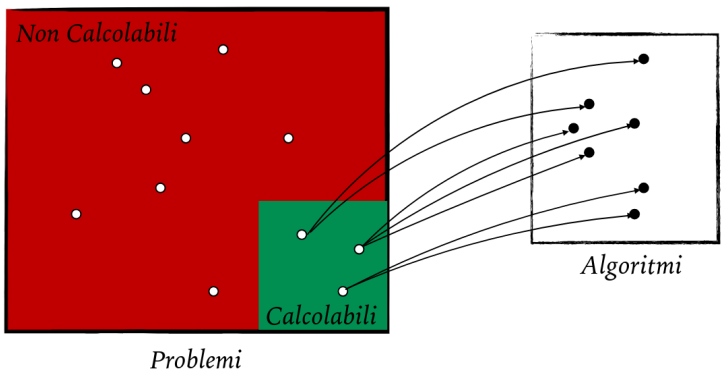

Figura 1: Cardinalità delle funzioni

3 Principio di induzione

Il principio di induzione è un meccanismo di definizione e dimostrazione che funziona **solo su insiemi infiniti**. Esistono due metodi di induzione:

- Induzione matematica
- Induzione strutturale

In questo corso tratteremo solo l'induzione matematica.

Un insieme A infinito con una relazione di ordine non riflessiva (senza l'uguale perchè l'elemento non è in relazione con sè stesso): $<$: $(A, <)$. $A = \mathbb{N}$ e $<$ è

l'ordinamento stretto tra numeri naturali. La relazione di ordine deve essere **ben fondata**, quindi non devono esserci catene discendenti infinite, cioè una sequenza di elementi in ordine decrescente infinita:

$$a_0 > a_1 > a_2 > a_3 > \ldots \rightarrow \text{non ben fondata}$$

Una relazione di ordine riflessiva non è ben fondata, perchè esistono catene infinite:

$$a_0 \geqslant a_1 \geqslant a_2 \geqslant a_3 \geqslant a_3 \geqslant a_3 \geqslant \ldots \rightarrow \text{non ben fondata}$$

b minimale in $A : b \in A$ b è minimale se $\forall b' < b \;.\; b' \notin A$ Ad esempio: $\{1, 2, 3\}$ ha come minimali (di contenimento) $\{1, 2\}$ e $\{2, 3\}$.

Definizione 3.1 (Principio di induzione)**.** Se A è un insisme ben fondato (con ordinamento $<$), e Π è una proprietà definita sugli elementi di $A : \Pi \subseteq A$, allora:

$$\forall a \in A \;.\; \underbrace{\Pi(a)}_{\substack{a \text{ soddisfa } \Pi}} \iff \underbrace{\forall a \in A \;.\; [[\forall b < a \;.\; \Pi(b)] \Rightarrow \Pi(a)]}_{\substack{\text{Se dimostriamo } \Pi \text{ per ogni elemento più piccolo di } a, \\ \text{allora } \Pi \text{ vale anche per } a}}$$

Consideriamo come caso base gli elementi minimali di A:

$$\text{Base}_A = \{a \in A \mid a \text{ minimale}\}$$

Se si dimostra che Π vale per tutti gli elementi minimali di A (la base):

$$\overbrace{\underbrace{\forall a \in \text{Base}_A \;.\; \Pi(a)}_{\substack{\text{Dimostriamo } \Pi \text{ per ogni} \\ \text{elemento minimale di } A}}}^{\text{Base}} \wedge \underbrace{\forall a \in A \setminus \text{Base}_A}_{\text{Passo induttivo}} \;.\; \underbrace{\forall b < a \;.\; \Pi(b)}_{\text{Ipotesi induttiva}} \quad \underbrace{\Rightarrow \Pi(a)}_{\text{Tesi da dimostrare}}$$

Esempio 3.1. Dimostriamo che:

$$\forall n \in \mathbb{N} \sum_{i=1}^{n} i = \frac{n(n+1)}{2}$$

L'insieme è:

$$A = \mathbb{N} \setminus \{0\} = \{1, 2, 3, \ldots\}$$

- **Basse**

$$\text{Base}_A = \{1\}$$

  - dimostriamo la base

$$\sum_{i=1}^{1} i = n(n+1)/2 = 1(1+1)/2 = 1$$

- **Passo induttivo**: Prendo $n \in \mathbb{N}$

– Ipotesi induttiva, cioè per ogni $m < n$ vale la proprietà:

$$\forall \underset{\in \mathbb{N}}{m} < n \; . \; \sum_{i=1}^{m} i = \frac{m(m+1)}{2}$$

Dobbiamo dimostrare la proprietà per n:

$$\sum_{i=1}^{n} i = \sum_{i=1}^{n-1} i + n$$

$n - 1 < n$ quindi vale l'ipotesi induttiva

$$\sum_{i=1}^{n-1} i = \frac{(n-1)(n-1+1)}{2} = \frac{(n-1)n}{2}$$

$$\downarrow$$

$$\begin{aligned} \sum_{i=1}^{n} i = \sum_{i=1}^{n-1} i + n \\ &= \frac{(n-1)n}{2} + n \\ &= \frac{(n-1)n + 2n}{2} \\ &= \frac{n^2 - n + 2n}{2} \\ &= \frac{n(n+1)}{2} \quad \square \end{aligned}$$

È quindi dimostrato che:

$$\forall n \in \mathbb{N} \sum_{i=1}^{n} i = \frac{n(n+1)}{2}$$

3.1 Linguaggi formali

Definizione 3.2. Un linguaggio formale è un insieme di stringhe costruite su un alfabeto finito Σ .

Solitamente un linguaggio formale $\mathcal{L}$ è un sottoinsieme di Σ^*, tipicamente infiniti, ma non necessariamente:

$$\mathcal{L} \subseteq \Sigma^*$$

I linguaggi sono divisi in:

- Linguaggi finiti
- Linguaggi regolari, il modello utilizzato è l'automa a stati finiti.

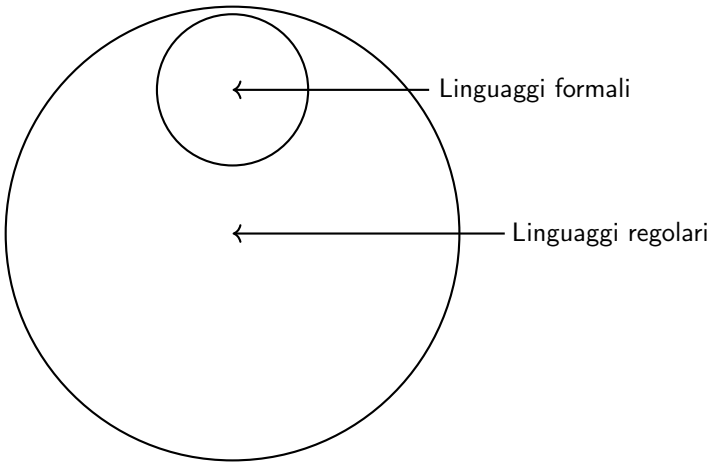

Figura 2: Linguaggi formali e linguaggi regolari