

# Teoria della Computazione

UniShare

Davide Cozzi  
@dlcgold

# Indice

<b>1</b>	<b>Introduzione</b>	<b>2</b>
<b>2</b>	<b>Complessità computazionale</b>	<b>3</b>
2.1	Tempo di calcolo di una TM . . . . .	3

# Capitolo 1

## Introduzione

Questi appunti sono presi a lezione. Per quanto sia stata fatta una revisione è altamente probabile (praticamente certo) che possano contenere errori, sia di stampa che di vero e proprio contenuto. Per eventuali proposte di correzione effettuare una pull request. Link: <https://github.com/dlccgold/Appunti>.

## Capitolo 2

# Complessità computazionale

L'informatica è costruita su una logica matematica. Il punto di partenza è stato dettato da Turing (con la **macchina di Turing** ( $TM$ )) e questo pensiero si è poi sviluppato nel tempo. Turing, con la sua macchina logica, ha dimostrato che ci sono funzioni non calcolabili, verità logiche non dimostrabili.

Subito dopo la macchina di Turing nasce la teoria della **complessità computazionale**, col fine di classificare i problemi in base alla difficoltà delle soluzioni mediante macchine di calcolo. Tale *difficoltà* viene stimata rispetto a **spazio** e **tempo**. La teoria della **complessità computazionale** si riferisce a varie **classi di complessità** che classificano, in un primo approccio, *problemi decisionali* descritti da funzioni binarie che hanno in input una stringa sull'alfabeto  $\{0, 1\}$  e restituiscono un bit (o 0 o 1). Questo perché le macchine di Turing ragionano in binario. Si ha quindi:

$$f : \{0, 1\}^* \rightarrow 0, 1$$

**Esistono problemi che si è dimostrato non essere risolvibili in tempo efficiente.**

Tra le classi abbiamo i **problemi NP** e **problemi P**. Inoltre i problemi NP sono a loro volta classificabili tra loro cercando i più difficili, ottenendo **problemi NP-hard** e **problemi NP-complete** (esistono varie dimostrazioni per la *NP-completezza*).

### 2.1 Tempo di calcolo di una TM

**Definizione 1.** Sia  $T : \mathbb{N} \rightarrow \mathbb{N}$  una funzione calcolabile da TM e  $L\pi$  un linguaggio di decisione (dove  $\pi$  sta per “problema” e “di decisione” ci ricorda che il risultato sarà binario) allora una **TM deterministica**  $M$  accetta  $L\pi$

in tempo  $T(n)$  se,  $\forall x \in L\pi$ , con  $|x| = n$ ,  $M$  accetta  $x$  in  $T(n)$  mosse o configurazioni

**Definizione 2.** La classe dei linguaggi di decisione accettati in tempo  $T(n) = cn^p$ ,  $p \in \mathbb{N}$  da una TM deterministica è detta **classe P**, quindi in un tempo polinomiale sulla dimensione dell'input. Quindi  $P$  è una classe di **problemi di decisione**.

**Definizione 3.** Un **problema di decisione**  $\pi$  riceve in input un'istanza  $x$  e l'output è:

- 0 che vuole dire no
- 1 che vuole dire yes

Un linguaggio  $L\pi$  restituisce 1 per tutti gli  $x$  che appartengono al linguaggio. Quindi  $L\pi$  è l'insieme degli input di  $\pi$  su cui l'output è 1 (è l'analogo della funzione caratteristica di un insieme, ovvero la funzione che risponde 1 sse un certo elemento appartiene all'insieme di riferimento).

La **funzione associata al problema** si chiama  $f\pi$  ed è la funzione che dato un input restituisce 1 sse l'input appartiene al  $L\pi$ .