Teoria della Computazione

UniShare

Davide Cozzi
@dlcgold

Indice

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/dlcgold/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 il 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:

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

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 $T : \mathbb{N} \to \mathbb{N}$ una funzione calcolabile da TM e $L\pi$ un linguaggio di decisione (dove π 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. *Un **problema di decisione** π 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 π 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 $L\pi$.*

Approfondiamo ora lo studio della **classe P**.

Definizione 3. *La classe dei linguaggi di decisione accettati in tempo $T(n) = cn^p$, $p \in \mathbb{N}$, $p \neq 0$ da una TM deterministica è detta **classe P**, quindi in un tempo polinomiale sulla dimensione dell'input n, è detta **classe P**. Quindi P è una classe di **problemi di decisione**.*
Potenzialmente p potrebbe anche non essere un intero in quanto si potrebbero avere tempi frazionari.

Definizione 4. *Si definisce che $L\pi$ è accettato da una TM in tempo $T(n)$ se $\exists\, T : \mathbb{N} \to \mathbb{N}$ calcolabile da TM e $\forall x \in L\pi$, con $|x| = n$, la TM accetta x e risponde 1 (yes) in al più $T(n)$ mosse di calcolo (dette anche configurazioni). Nel caso del modello della macchina RAM si ha la stessa situazione con però $T(n)$ **istruzioni RAM** e si dice che $L\pi$ è accettato dalla macchina RAM (si può dire che è anche deciso dell'algoritmo A della macchina RAM). In caso contrario la macchina RAM restituisce no, in quanto si parla di "decisione" oltre che di "accettazione" (a differenza della TM, dove però si può ottenere lo stesso discorso parlando di TM complementare M', che in $T(n)$ mi risponderà yes alla richiesta che un input non appartenga a $L\pi$, altrimenti bisogna fissare un limite di tempo per ottenere yes).*
È dimostrabile che se $L\pi$ è accettabile in tempo polinomiale allora nello stesso tempo è anche decidibile.
*La differenza tra accettazione e decisione sarà fondamentale nel **modello non deterministico**.*

Si ricordi che il **modello RAM (*Random Access Machine*)** è usato per studiare il tempo di calcolo di uno pseudocodice. È un modello teorico (una macchina teorica "simile" a quelle reali) dotato di istruzioni come *load, store, add, etc...* dove un codice (ipoteticamente in qualsiasi linguaggio incluso lo pseudocodice) viene tradotto in una sorta di linguaggio macchina (linguaggio RAM), dove n è un intero rappresentante il numero di istruzioni RAM necessarie per ottenere l'output (n è detto **tempo uniforme**). Sul linguaggio RAM si può studiare anche lo spazio calcolato come numero di bit necessari per la computazione (è detto **costo logaritmico**). In questo secondo punto il costo di un'istruzione, come ad esempio *load(n)*, è logaritmico rispetto all'operando n ($\log_2 n$), studia quindi la *dimensione* dell'input.

Consideriamo ora un modello basato su algoritmi.

Definizione 5. *Sia $L\pi$ un linguaggio di decisione e $t : \mathbb{N} \to \mathbb{N}$ una funzione calcolabile, allora un algoritmo A accetta $L\pi$ in tempo $T(n)$ sse $\forall x \in L\pi$, con $|x| = n$, A termina su input x dopo $T(|x|)$ passi di calcolo, ovvero istruzioni esguite, producendo 1 come output. Quindi P è la classe dei linguaggi di decisione accettati in tempo $T(n) = cn^p$, $p \in \mathbb{N}$ (con lo stesso discorso di sopra su p) da un algoritmo A in tempo polinomiale.*