1 Introduction

1.1 Logistica del Corso

Il corso è strutturato in due moduli disconnessi: algoritmi di approssimazione e algoritmi distribuiti (affrontati dal punto di vista applicato nella sezione del prof. Cabri).

Esame: orale (2/3) e progetto (1/3)

Slide non sempre sufficienti, vedi libro, vecchie edizioni rilasciate pubblicamente.

1.2 Problemi

Ciò che vogliamo risolvere, generalizzando il più possibile.

I problemi possono essere classificati in 3 categorie, in base al tipo di soluzione cercata:

- Problemi decisionali: output Y/N
- Problemi di ricerca: cerca una *soluzione ammissibile* (ammissibile = che soddisfa una proprietà) tra tutte le possibili soluzioni.
- Problemi di ottimizzazione: ad ogni soluzione ammissibile è associato un costo; si cerca la *soluzione ottima*, ovvero quella che massimizza o minimizza tale costo

1.3 Algoritmi

Def. Un algoritmo è una procedura generale generale per risolvere un problema, con le seguenti caratteristiche: sequenza finita di passi (altrimenti il sorgente non sarebbe finito), non ambigua, effettivamente realizzabile (eg, no divisioni per zero), in grado di terminare in tempo finito. L'algoritmo deve sempre ritornare un risultato che appartiene all'insieme delle soluzioni; una soluzione può essere anche un errore. (Knuth)

Un algoritmo non corretto può non terminare o non ritornare un risultato errato (sollevando un errore).

Gli algoritmi efficienti usano poche risorse in termini di spazio e tempo. Il tempo è più prezioso rispetto allo spazio, in quanto non è riutilizzabile.

Le misure di efficienza devono essere generalizzabili, non specifico per un dato esecutore. Per il tempo si usa il numero di operazioni di elementari in funzione dell'input, necessarie per il problema nel caso peggiore. Questo funge da upper bound per le altre esecuzioni. Per lo spazio si usa il numero di celle di memoria, in funzione della dimensione dell'input.

Def. La dimensione del problema misura la quantità di informazione necessaria per codificare l'istanza di un problema.

Nel costo logaritmico la dimensione dipende dal numero di bit necessari per rappresentare un input. Nel costo uniforme la dimensione dipende dal numero di elementi necessari per rappresentare l'input. E.g. fattorizzazione di un numero.

Il costo computazionale di un *problema* è il costo dell'algoritmo più efficiente (di minimo costo) che risolve il problema. L'UB di un problema può essere impostato trovando un algoritmo in grado di risolvere il problema. Il LB di un problema è il minimo costo necessario per un algoritmo per risolvere il problema. Dimostrare il LB non è scontato; eg. per i problemi di ordinamento il LB è $nlog(n)$.

1.4 Classificazione dei Problemi

La difficoltà di un problema dipende dal costo dell'algoritmo risolutivo.

- Problemi trattabili: esiste un algo di costo polinomiale
- Problemi presumilmente intrattabili: non abbiamo un algo di costo polinomiale ma non è stato dimostrato che non esista
- Problemi intrattabili: si può dimostrare non esista un algo di costo polinomiale
- Problemi irrisolvibili: si può dimostrare non esiste un algo per il problema (o meglio, esiste con un numero irragionevole di risorse)

Il costo polinomiale è stato scelto in quanto rappresenta un costo che permette al problema di essere risolvibile nella realtà. La maggior parte dei problemi di costo polinomiale conosciuti hanno un esponente basso. I problemi intrattabili possono richiedere un output di dimensione non polinomiale rispetto all'input (e.g. generare tutti i numeri rappresentabili con un certo numero di cifre).

La teoria della complessità categorizza i problemi nelle diverse classi.

Tesi di Curch-Turing (1936): modelli di calcolo diversi sono simulabili a vicenda con slowdown polinomiale. Ovvero, sono tutti polinomialmente equivalenti alla Macchina di Turing.

La trattazione formale più agevole per trattare alcuni problemi è quella in cui si considerano la variante decisionale del problema. Per esempio, il problema dello shortest path ammette la versione di ottimizzazione, in cui dato un grafo e due vertici ci si chiede la lunghezza del cammino minima tra i due vertici, e la versione decisionale in cui si aggiunge un valore k e ci si chiede se esista un cammino migliore (lunghezza minore o uguale) di k.