Università Roma Tre – Dipartimento di Matematica e Fisica

Percorso Abilitante Speciale
Classe A048 – Matematica Applicata

Corso di Informatica

Complessità computazionale

Marco Liverani
(liverani@mat.uniroma3.it)

Sommario

- Problema e istanze del problema, dimensione dell'istanza
- Complessità di un algoritmo come misura della sua efficienza
- La complessità computazionale è una funzione
- Complessità di un problema
- Problemi trattabili e intrattabili
- La classe P e la classe NP
- La classe dei problemi NP-completi e il problema "P=NP"

Problemi e istanze

- Un problema viene generalmente formulato in **forma astratta**
  - dati i nominativi degli studenti del corso, scriverli in ordine alfabetico
  - dato l'indirizzo di partenza e quello di arrivo, trovare il percorso più breve in auto
  - ecc.
- Un'istanza di un problema è una **formulazione concreta**, corredata da tutte le informazioni necessarie a risolverlo
  - il corso è frequentato da 5 studenti: Rossi, Bianchi, Verdi, Abate e Giacomini; scrivere questi nominativi in ordine alfabetico
  - partendo da p.zza Mazzini, qual è il percorso più breve per raggiungere in automobile l.go Murialdo, muovendosi in automobile a Roma?
  - ecc.

Dimensione dell'istanza di un problema

- L'istanza di un problema è caratterizzata da una **dimensione**, ossia dal *numero di informazioni* che la caratterizzano e che devono essere elaborate per risolvere il problema
- Problema astratto:

$$a_0x^n+a_1x^{n-1}+a_2x^{n-2}+\ldots+a_{n-1}x+a_n = 0$$

- Istanza concreta del problema:

$$n=4,\ a_0=-5,\ a_1=3,\ a_2=7,\ a_3=-1,\ a_4=-3$$

$$-5x^4+3x^3+7x^2-x-3 = 0$$

- Dimensione dell'istanza: 6

Complessità di un algoritmo 1

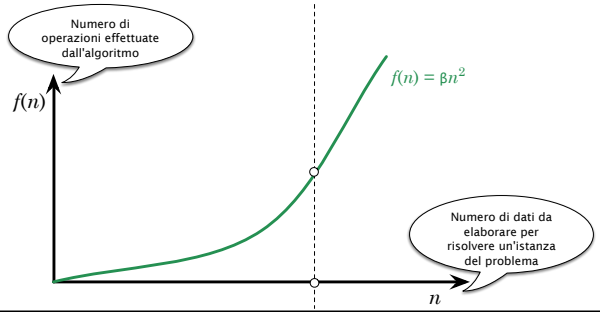
- L'algoritmo A risolve il problema P se, per ogni istanza di P, l'algoritmo termina dopo un numero finito di passi, producendo la soluzione per tale istanza del problema
- È possibile contare il numero di operazioni elementari effettivamente eseguite dall'algoritmo per risolvere una determinata istanza di dimensione n
- Il numero di operazioni eseguite dipende dal numero di informazioni da elaborare (la dimensione dell'istanza) oltre che dalla struttura dell'algoritmo

Complessità di un algoritmo 2

- Il numero di operazioni eseguite dall'algoritmo per risolvere un'istanza del problema di dimensione n **è una funzione**

$$f : \mathrm{N} \rightarrow \mathrm{N}$$

- Per risolvere un'istanza di dimensione n l'algoritmo esegue $f(n)$ operazioni

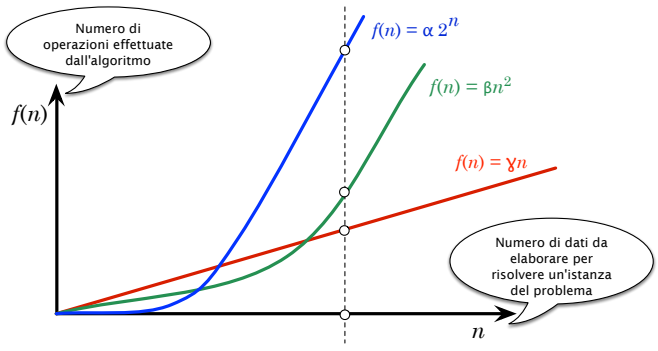


Complessità di un algoritmo 3

- Algoritmi diversi per la soluzione del medesimo problema possono eseguire un numero di operazioni differente per risolvere la stessa istanza del problema
- Ad esempio:
  - l'algoritmo A_1 impiega $2n$ operazioni per risolvere una certa istanza del problema P di dimensione n
  - l'algoritmo A_2 impiega n^2 operazioni per risolvere la stessa istanza del problema P
  - Se n=10 allora A_1 esegue 20 operazioni, mentre A_2 ne esegue 100
  - **Dunque A_1 è più veloce, più efficiente, di A_2**

Complessità di un algoritmo 4

- Algoritmi diversi per la soluzione del medesimo problema possono eseguire un numero di operazioni differente per risolvere la stessa istanza del problema



Complessità di un algoritmo 5

- Uno stesso algoritmo su due istanze diverse ma di uguale dimensione, dello stesso problema, può avere comportamenti differenti e impiegare tempi diversi
- Si studia quindi il **caso peggiore**, quello in cui l'algoritmo risulta meno efficiente
- Si considera inoltre solo il **comportamento asintotico** della funzione al crescere di n
- $O(f(n)) = \{g(n)$: esiste n_0, $c>0$ tali che $g(n) < c\ f(n)$ per ogni $n>n_0\}$

Complessità di un problema

- La **complessità di un problema** è la complessità più bassa che può avere un *algoritmo risolutore* del problema stesso
- Non devo necessariamente trovare un simile algoritmo, basta *dimostrare* che nessun algoritmo che risolve il problema può avere una complessità più bassa
- Esempio: $O(n \log_2 n)$ è la minima complessità di un algoritmo che risolve il problema dell'ordinamento degli elementi di un insieme attraverso confronti e scambi e senza informazioni aggiuntive sugli elementi dell'insieme

Problemi trattabili e intrattabili 1

- Se traduco l'algoritmo in un programma per il computer la complessità dell'algoritmo fornisce un'indicazione del **tempo impiegato dalla macchina**, ipotizzando che per eseguire ciascuna operazione elementare la macchina impieghi *un'unità di tempo* (es.: un milionesimo di secondo)
  - l'algoritmo A ha una complessità $O(n^2)$
  - quindi per risolvere un'istanza del problema con n=2.000 esegue circa n^2=4.000.000 operazioni
  - se per eseguire un'operazione impiega un milionesimo di secondo, per risolvere il problema impiegherà 4 secondi

Problemi trattabili e intrattabili 2

- Attenzione alle funzioni complessità!
- Se la macchina esegue 100 milioni di operazioni al secondo:

$f(n)$	n=10	n=20	n=40	n=100
n	10^{-7} sec	2×10^{-7} sec	4×10^{-7} sec	10^{-6} sec
$n \log_2 n$	3×10^{-7} sec	8×10^{-7} sec	2×10^{-6} sec	6×10^{-6} sec
n^2	10^{-6} sec	4×10^{-6} sec	0,000016 sec	0,0001 sec
n^4	0,0001 sec	0,0015 sec	0,0256 sec	1 sec
2^n	0,00001 sec	0,01 sec	**3 ore**	**miliardi di anni**
$n!$	0,036 sec	**7 secoli**	**miliardi di anni**	**...**

- Problemi con complessità *super-polinomiale* sono **intrattabili**!

Raggruppiamo i problemi in classi

- **P** = insieme dei problemi che possono essere *risolti* con un algoritmo di complessità polinomiale
- **NP** = insieme dei problemi la cui soluzione può essere *verificata* da un algoritmo di complessità polinomiale
- **NP-completi** = insieme dei problemi NP a cui possono essere *ricondotte facilmente* (con un algoritmo di complessità polinomiale) le istanze di *tutti* gli altri problemi NP

Problemi difficili (NP-completi)

- SUBSET-SUM: Dato un insieme di numeri interi $X = \{x_1, x_2, ..., x_n\}$ è possibile costruire una partizione in due componenti di uguale somma?
- PARTITION: Dato un insieme di numeri interi $X = \{x_1, x_2, ..., x_n\}$ e un intero k, esiste un sottoinsieme di X la cui somma degli elementi è k?
- ecc.

... ricordiamo infatti che l'insieme delle parti di un insieme con n elementi ha cardinalità 2^n