TEORIA della COMPLESSITA'

Esistono problemi irrisolvibili a prescindere dal tempo di risoluzione (equazioni diofantee, problema dell'arresto)

$\longrightarrow$ PROBLEMI DECIDIBILI

risolvibili che richiedono tempi esponenziali nella dimensione dell'istanza (torri di Hanoi)

$\longrightarrow$ PROBLEMI INTRATTABILI

risolvibili con algoritmi di costo polinomiale

"FACILI"

$\longrightarrow$ PROBLEMI TRATTABILI

il cui stato non è noto (si hanno solo algoritmi di costo esponenziale e lower bound polinomiale non dimostrato)

$\longrightarrow$ PROBLEMI PRESUMIBILMENTE INTRATTABILI

Suppongo: c_1, c_2 calcolatori t.c. c_2 k volte più veloce di c_1

n_1 dati trattabili da c_1 in tempo t

n_2 c_2

oss. c_2 usato per tempo t è come usare c_1 $k \cdot t$ tempo

Algoritmo polinomiale che risolve in $c \cdot n^s$ secondi (c, s cost.)

$C_1 \rightarrow$ risolve in $c \cdot n_1^s = t \Rightarrow n_1 = \sqrt[s]{t/c}$

$C_2 \rightarrow$ $c \cdot n_2^s = k \cdot t \Rightarrow n_2 = \sqrt[s]{k} \cdot \sqrt[s]{t/c}$

$\Rightarrow n_2 = \sqrt[s]{k} \cdot n_1$

esponenziale che risolve il problema in $c \cdot 2^n$ secondi

$$C_1 \rightarrow c \cdot 2^{n_1} = t \Rightarrow 2^{n_1} = t/c$$

$$C_2 \rightarrow c \cdot 2^{n_2} = k \cdot t \Rightarrow 2^{n_2} = k \cdot t/c = k \cdot 2^{n_1}$$

$$n_2 = \log k + n$$

k diventa un fattore additivo che dipende dal logaritmo

PROBLEMI

Problema Π,

Istanze in ingresso I

Insieme delle soluzioni S

Si distinguono:

problemi decisionali ($S = \{0,1\}$):

istanze positive $x \in I$: $\Pi(x) = 1$

negative $\Pi(x) = 0$

di ricerca (richiedono di trovare una soluzione ammissibile)

ottimizzazione (richiedono la soluzione migliore)

La teoria della complessità è definita in termini di problemi decisionali:

- il tempo richiesto per scrivere la soluzione è piccolo, tutto quello richiesto è tempo di calcolo
- un problema resta difficile anche in forma decisionale

Si possono esprimere problemi di ottimizzazione in forma decisionale

"Trovare la massima clique" $\rightarrow$ "Esiste una clique di almeno k vertici?"

Il problema di ottimizzazione è almeno difficile quanto quello decisionale (ne dà un lower bound)

CLASSI di COMPLESSITÀ

Π problema decisionale

A algoritmo

A risolve Π se

x istanza di input $A(x) = 1 \Leftrightarrow \Pi(x) = 1$

Lo risolve in $t(n)$, $s(n)$ se $t(n)$ e $s(n)$ sono tempo di esecuzione e spazio di occupazione.

Data $f(n)$:

Time $(f(n))$ problemi decisionali risolvibili in tempo $O(f(n))$

Space $(f(n))$ spazio $O(f(n))$

CLASSE P Problemi decisionali con tempo di risoluzione polinomiale

$\exists$ c, n_0 costanti positive t.c. il numero di passi elementari è al più n^c per ogni input di dimensione n e per ogni $n_0 > n$

PSPACE spazio

EXPTIME tempo esponenziale

$P \subseteq PSPACE$, (E' noto solo $P \subset EXPTIME$)

$PSPACE \subseteq EXPTIME$

esempi: Clique, cammino hamiltoniano, SAT e EXPTIME
(2^n sopra Clique, $n!$ sopra cammino hamiltoniano, 2^n sopra SAT)

CERTIFICATI Alcuni problemi non sono risolvibili in tempo polinomiale ma verificarne una istanza valida e' semplice

Per le istanze accettabili x e' possibile fornire un certificato y che dimostra che l'istanza soddisfi la proprieta' e sia dunque facile

CLIQUE → Insieme dei vertici

CAMMINO HAMILTONIANO → Permutazione

SAT → Assegnamento

E' un attestato breve di esistenza di una soluzione con determinate proprieta' definito solo su istanze accettabili

VERIFICA Π risulta <u>verificabile in tempo polinomiale</u> se:

- $\forall$ istanza accettabile di Π lunga n esiste un certificato y di lunghezza polinomiale in n
- $\exists$ algoritmo di verifica polinomiale in n applicabile a ogni coppia (x, y) che permette di attestare che x sia accettabile

La classe NP e' definita come "classe dei problemi verificabili in tempo polinomiale"
(equiv. risolvibile in tempo polinomiale non deterministico)

Vale banalmente $P \subseteq NP$

(Per ogni problema in P si esegue l'algoritmo che lo risolve e si ricava dunque il certificato)

Non e' noto se $P = NP$, $P \neq NP$

Esistono problemi NP completi ossia i problemi piu' difficili in NP: si ricorre alla tecnica della riduzione polinomiale, se esistesse un algoritmo polinomiale che risolve un problema NP completo allora $P = NP$, se esistesse un lower bound esponenziale allora varrebbe $P \neq NP$

Π_1, Π_2 problemi decisionali

I_1, I_2 istanze di input

Π_1 si riduce in tempo polinomiale a Π_2 se esiste una funzione calcolabile in tempo polinomiale t.c. per ogni istanza di Π_1 questa e' accettabile se e solo se la funzion applicata a quella medesima istanza e' istanza accettabile per Π_2:

$(\exists f : I_1 \mapsto I_2 \mid x$ istanza accettabile di Π_1

$\Longleftrightarrow$

$f(x)$ istanza accettabile di $\Pi_2)$

$\Rightarrow \Pi_1 \leq_p \Pi_2$

Definisco Π <u>NP-arduo</u> se $\forall \Pi' \in NP.\ \Pi' \leq_p \Pi$

NP completo se $\Pi \in NP \wedge \forall \Pi' \in NP.\ \Pi' \leq_p \Pi$

TEOREMA di COOK (1971) SAT e' NP Completo