Progetto e analisi di algoritmi

Roberto Cordone

DTI - Università degli Studi di Milano

Polo Didattico e di Ricerca di Crema

Tel. 0373 / 898**089**

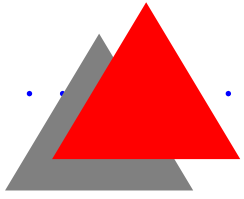
E-mail: **cordone@dti.unimi.it**

Ricevimento: **su appuntamento**

Web page: **http://www.dti.unimi.it/~cordone**

Lezioni: **Martedì dalle 11.00 alle 13.00**

Giovedì dalle 11.00 alle 13.00



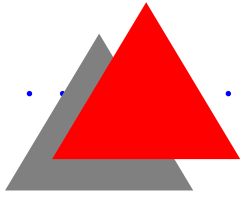
Formule logiche

- Variabile logica x_j è una variabile che assume solo i valori *Vero* o *Falso*
- Letterale è una variabile logica (x_j) o la sua negazione ($\bar{x}_j$)
- Formula logica è una funzione ottenuta combinando letterali tramite somme (*OR*) o prodotti (*AND*)

Esempio: $f = (x_1\bar{x}_2 + x_1x_2\bar{x}_3)(\bar{x}_1 + x_3)$

Forma congiuntiva normale (*CNF*) è una formula logica costituita da un prodotto di somme di letterali

Esempio: $f' = x_1(\bar{x}_1 + \bar{x}_2)(\bar{x}_1 + x_3)$



Soddisfacibilità (SAT)

Data una *CNF* $f(x_1, \ldots, x_n)$, esiste un assegnamento di valori alle variabili x_j tale che f sia vera?

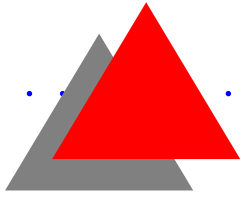
È un problema di decisione (S =*Sì* o *No*)

Forza bruta: valutare f per ogni assegnamento

Ma gli assegnamenti possibili sono esponenziali (2^n)!

x_1	0	0	0	0	1	1	1	1
x_2	0	0	1	1	0	0	1	1
x_3	0	1	0	1	0	1	0	1
f	0	0	0	0	0	1	0	0

$$f = x_1 (\bar{x}_1 + \bar{x}_2)(\bar{x}_1 + x_3)$$



Verifica di una soluzione

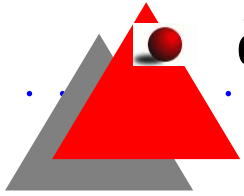
Molti problemi riguardano l'esistenza o meno di sottosistemi del sistema dato, che godano di opportune proprietà

- data una *CNF*, esiste un assegnamento di verità che la soddisfa?
- dato un grafo, esiste un ciclo?
- dato un grafo, esiste un ciclo che visita tutti i nodi?

Impropriamente li chiamiamo "soluzioni" (S =*Sì* o *No*)

Per molti di questi problemi

- può essere difficile risolvere il problema
- è facile verificare che una data "soluzione" è valida



Riconoscimento di relazioni

Relazione fra insiemi A e B è un sottoinsieme $R \subseteq A \times B$

Una macchina riconosce una relazione quando

- riceve la stringa (a, b), con $a \in A$ e $b \in B$
- restituisce *Sì* quando $(a, b) \in R$, *No* altrimenti

Sfruttiamo il concetto per definire una classe di problemi per i quali è facile verificare una "soluzione" (o certificato):

- a è l'istanza del problema
- b è il certificato da verificare
- una macchina verifica che a ha soluzione *Sì* a causa di b

Esempio

$$a = x_1 \left(\bar{x}_1 + \bar{x}_2\right) \left(\bar{x}_1 + x_3\right) \qquad b = (1\,0\,1)$$

Diciamo che $(a, b) \in R$ quando si può interpretare a come CNF e b come assegnamento alle variabili di a e quando $a\,(b) = 1$

- a è una CNF con 3 variabili
- b contiene 3 assegnamenti
- sostituendo $x_1 = 1$, $\bar{x}_1 = 0 \ldots$ si ottiene

$$a\,(b) = 1\ (0 + 1)\,(0 + 1) = 1$$

- quindi $(a, b) \in R$ (b è un certificato valido per a)

$\Rightarrow a$ ha soluzione *Sì*

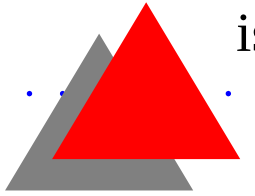
Certificati e $\mathcal{NP}$

$\mathcal{NP}$ è l'insieme dei problemi P per i quali esistono

- un insieme di certificati Δ
- una relazione $R \subseteq P \times \Delta$
- un polinomio $p(\cdot)$

tali che

1. la relazione R è riconoscibile in tempo polinomiale
2. ogni istanza positiva è in relazione con almeno un certificato:
   per ogni $I \in P^+$, $\exists D \in \Delta : (I, D) \in R$
3. nessuna istanza negativa I è in relazione con un certificato:
   per ogni $I \in P^-$ e per ogni $D \in \Delta$, è $(I, D) \notin R$
4. la dimensione di un certificato è al più $p(\cdot)$ nella dimensione delle istanze relative: $|D| \leq p(|I|)$ per ogni $(I, D) \in R$



Problemi in $\mathcal{NP}$

$SAT \in \mathcal{NP}$

- Δ è l'insieme delle stringhe binarie $\{0,1\}^*$
- $(f, D) \in R$: interpretato come assegnamento, D soddisfa f
- $p(x) = x$ (polinomio identità)

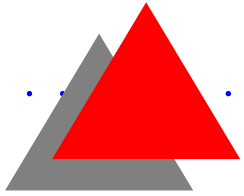
1. esiste una macchina che, per ogni $D \in \Delta$ e per ogni $f \in SAT$, verifica in tempo polinomiale se D soddisfa f (descriverla)
2. per ogni f positiva, esiste un D che soddisfa f (quindi la macchina risponde *Sì* sulla coppia (f, D))
3. per ogni f negativa, nessun D soddisfa f (quindi la macchina risponde *No* sulla coppia (f, D) per ogni $D \in \Delta$)
4. per ogni D che soddisfa f, $|D| \leq |f|$ (dunque è polinomiale)

Problemi in $\mathcal{NP}$

SAT $\in \mathcal{NP}$

1. $f_1 = x_1 \, (\bar{x}_1 + \bar{x}_2) \, (\bar{x}_1 + x_3)$ ammette certificato $D_1 = 101$
2. $f_2 = (x_1 + x_2) \, x_3 \, (\bar{x}_1 + \bar{x}_3) \, (\bar{x}_2 + \bar{x}_3)$ non ammette alcun certificato
3. $f_3 = (x_1 + x_2) \, (x_1 + x_3) \, (\bar{x}_1 + \bar{x}_2 + \bar{x}_3)$ ammette i certificati $D_1 = 101$ e $D_2 = 011$, $D_3 = 100$, $D_4 = 110$

D_1 è certificato di f_1 e f_3

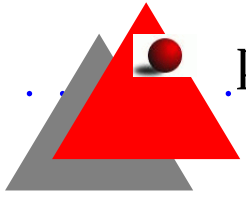


Problemi in $\mathcal{NP}$

GrafoCiclico $\in \mathcal{NP}$

L'insieme dei certificati è l'insieme delle stringhe di numeri naturali: $\Delta = \mathbb{N}^*$ (improprio... perché?)

- interpretiamo D come sequenza degli indici dei vertici di un ciclo
- esiste una macchina che, per ogni $D \in \Delta$ e per ogni G, verifica in tempo polinomiale se D è un ciclo di G (descriverla)
- per ogni G positiva, esiste un D che rappresenta un ciclo in G (quindi la macchina risponde *Sì* sulla coppia (G, D))
- per ogni G negativa, nessun D rappresenta un ciclo in G (quindi la macchina risponde *No* sulla coppia (G, D) per ogni $D \in \Delta$)
- per ogni grafo G e per ogni ciclo D in G, $|D| \leq |f|$ (polinomiale)



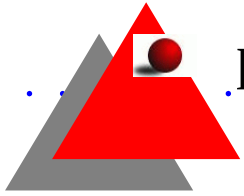
Problemi in $\mathcal{NP}$

TSP $\in \mathcal{NP}$

L'insieme dei certificati è l'insieme delle stringhe di numeri naturali:

$\Delta = \mathbb{N}^*$ (improprio... perché?)

- D è la sequenza degli indici dei vertici di un ciclo hamiltoniano
- esiste una macchina che, per ogni $D \in \Delta$ e per ogni G, verifica in tempo polinomiale se D è un ciclo hamiltoniano di G (quale?)
- per ogni G positiva, esiste un D che rappresenta un ciclo hamiltoniano in G (quindi la macchina risponde *Sì* su (G, D))
- per ogni G negativa, nessun D rappresenta un ciclo hamiltoniano in G (quindi la macchina risponde *No* su (G, D) per ogni D)
- per ogni grafo G e per ogni ciclo hamiltoniano D in G, $|D| \leq |f|$

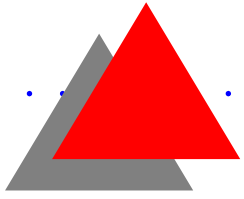


Problemi in $\mathcal{NP}$

TSP $\in \mathcal{NP}$

1. $G_1 = \{(1,2),(1,3),(1,4),(3,1),(4,2),(4,3)\}$ non ammette certificati
2. $G_2 = \{(1,2),(2,3),(3,1),(3,4),(4,1)\}$ ammette certificato $D_1 = 1234$
3. $G_3 = \{(1,2),(2,1),(2,3),(3,4),(4,1),(4,2)\}$ ammette certificato $D_1 = 1234$ e $D_2 = 1342$

D_1 è certificato di G_2 e G_3



Problemi in $\mathcal{NP}$

Hanoi $\notin \mathcal{NP}$

- Un'istanza di Hanoi è $I = n$
- Proviamo a usare come certificato la sequenza di mosse che risolve il problema ($D = A \rightarrow B\ A \rightarrow C\ B \rightarrow C \ldots$)
- il certificato non è di dimensione polinomiale rispetto all'istanza ($D = 2^n - 1$)

Le sequenze di mosse non sono certificati validi

Le coppie (I, D) sono riconoscibili in tempo polinomiale rispetto alla propria lunghezza, ma non a quella di I

Proprietà di $\mathcal{NP}$ (1)

$\mathcal{P} \subseteq \mathcal{NP}$

Ogni problema polinomiale è in $\mathcal{NP}$

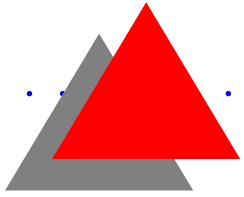
Basta un solo certificato "fasullo" ($\Delta = \{D\}$, con $D = ()$)

- Definiamo la relazione $(I, D) \in R$ se e solo se $I \in P^+$
- Il certificato è ovviamente polinomiale

$$|D| = 0 \leq |I| \qquad \text{per ogni } I \in P$$

- Una macchina che ignora D e risolve l'istanza I, in tempo polinomiale risponde *Sì* se $I \in P^+$, *No* se $I \notin P^+$

Quindi riconosce la relazione R!

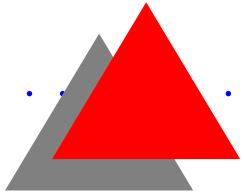


Proprietà di $\mathcal{NP}$ (2)

Per ogni problema in $\mathcal{NP}$ esiste una macchina che lo risolve in tempo $O\left(2^{f(n)}\right)$, dove $f(\cdot)$ è un un polinomio

- si legge l'istanza I e se ne valuta la dimensione $n = |I|$
- si generano tutti i $2^{p(n)}$ certificati potenziali D per l'istanza I (tutte le stringhe con $|D| \leq p(n)$)
- per ogni D, si applica a (I, D) la macchina che riconosce in tempo $q(n')$ se D è valido ($n' = |I| + |D| \leq n + p(n)$)
- la macchina risultante risolve il problema in tempo $O\left(2^{p(n)} q(n + p(n))\right)$

Ma questo algoritmo (esaustivo) è esponenziale...



Perché $\mathcal{NP}$?

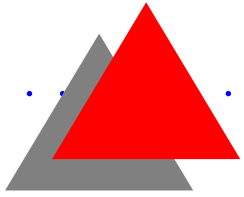
La sigla $\mathcal{NP}$ fa pensare a “non polinomiale”…

È vero che per alcuni di questi problemi non si conosce un algoritmo polinomiale, ma

- un algoritmo polinomiale potrebbe esistere
- per altri problemi si sa che esiste ($\mathcal{P} \subseteq \mathcal{NP}$)

La maggior parte dei problemi in $\mathcal{NP}$ sono aperti e il loro gap algoritmico è fra polinomiale ed esponenziale

$\mathcal{NP}$ significa non-deterministico polinomiale, cioè risolubile in tempo polinomiale da una macchina di Turing non deterministica



Macchina di Turing non deterministica (1)

Si definisce come la Macchina di Turing

$$NTM = (Q, q_0, A, \Sigma, -, \Gamma, \delta)$$

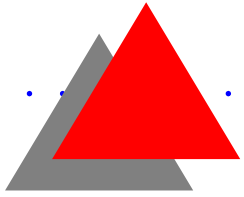
ma la funzione di transizione restituisce

un insieme di terne (stato,uscita,spostamento)

$$\delta : Q \times \Gamma \rightarrow 2^{Q \times \Gamma \times \{S,D\}}$$

anziché una singola terna

Che significa?



Macchina di Turing non deterministica (2)

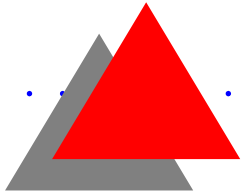
Ad ogni passo la *NTM* si "clona" e ogni copia assume una delle configurazioni indicate da δ

Ne risulta un'arborescenza computazionale, le cui foglie sono le configurazioni di arresto

Se tutte le computazioni terminano, la *NTM* risponde

- *Sì* quando almeno una foglia è in uno stato accettante
- *No* quando nessuna foglia è in uno stato accettante

La complessità è il massimo numero di passi compiuti



$\mathcal{NP}$ e *le* NTM

$\mathcal{NP}$ è l'insieme dei problemi risolubili in tempo polinomiale da una *NTM*

- la *NTM* genera clonandosi tutti i certificati potenziali (in tempo $p(n)$)
- ogni clone verifica uno dei certificati (in tempo $q(n + p(n))$)
- In complesso, impiega tempo $p(n) + q(n + p(n))$

Viceversa, se la NTM risolve P in tempo polinomiale, si può costruire un certificato per ogni computazione accettante, cioè sequenza di configurazioni che terminano in uno stato accettante

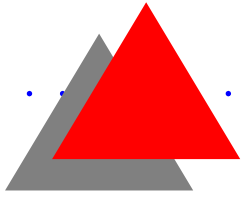
Perché "non deterministico"?

Si immagini una Macchina di Turing classica che decide la propria evoluzione in base a

- stato corrente
- ingresso corrente
- un "suggerimento" che indichi quale terna (stato,uscita,spostamento) realizzare per raggiungere una configurazione accettante (se esiste)

Funzionerebbe esattamente come la *NTM*

Per questo si parla di "macchina non deterministica"



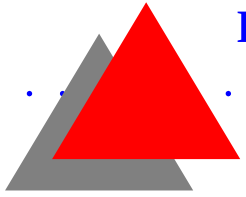
Asimmetria di $\mathcal{NP}$

Le definizioni di $\mathcal{NP}$ sono asimmetriche

1. almeno un certificato per le istanze positive
   nessuno per le negative
2. almeno una foglia accettante per le istanze positive
   nessuna per le negative

Ribaltandole, si ottiene un'altra classe di problemi per cui

1. almeno un certificato per le istanze negative
   nessuno per le positive
2. almeno una foglia accettante per le istanze negative
   nessuna per le positive



co-$\mathcal{NP}$ e complementarietà

co-$\mathcal{NP}$ è la classe ottenuta ribaltando la definizione di $\mathcal{NP}$

Contiene i problemi complementari ai problemi di $\mathcal{NP}$

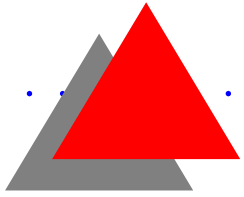
- "n è un numero composto?" è in $\mathcal{NP}$
- "n è un numero primo?" è in co-$\mathcal{NP}$

Dato $n \in \mathbb{N}$ ($|I| = \log_2 n$)

- un certificato polinomiale che n è composto
- un certificato polinomiale che n non è primo

è una coppia di fattori di n ($|D| = 2 \log_2 n \leq 2\,|I|$)

Esempio: $I = (24)$, $D = (3\ 8)$



Problemi ben caratterizzati

$\mathcal{P} \subseteq$ co-$\mathcal{NP}$: Ogni problema polinomiale è in co-$\mathcal{NP}$

Si dimostra col certificato "fasullo" come $\mathcal{P} \subseteq$ co-$\mathcal{NP}$

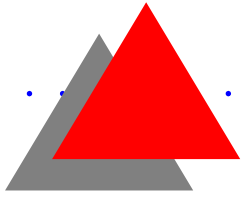
Problemi ben caratterizzati sono quelli in $\mathcal{NP} \cap$ co-$\mathcal{NP}$

Le istanze positive e negative hanno in genere certificati diversi

Esempio: "Nel grafo G, c'è un accoppiamento perfetto?"

- un accoppiamento perfetto è un certificato di esistenza
- il certificato di Tutte è un certificato di non esistenza

Attualmente, i problemi ben caratterizzati sono tutti polinomiali, ma è possibile che $P \subset \mathcal{NP} \cap$ co-$\mathcal{NP}$



Relazioni fra spazio e tempo (1)

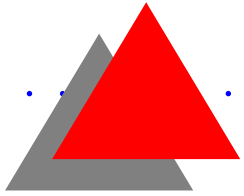
Sia $\mathcal{P}_{\text{space}}$ la classe dei problemi che si risolvono usando un numero polinomiale di celle

$$\mathcal{NP} \subseteq \mathcal{P}_{\text{space}}$$

perché l'algoritmo esaustivo usa

- n celle per l'istanza
- $p(n)$ celle per il certificato
- $\phi(n)$ celle per riconoscer la relazione (tempo polinomiale!)

In un tempo polinomiale si usa un numero polinomiale di celle,
per cui anche ϕ è un polinomio!



Relazioni fra spazio e tempo (2)

Sia *Exp* la classe dei problemi che si risolvono in un tempo esponenziale

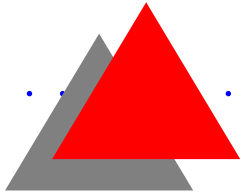
$$\mathcal{P}_{\text{space}} \subseteq Exp$$

Se la macchina ripete una configurazione, entra in ciclo infinito, e non termina: ci sono meno passi che configurazioni!

Quante configurazioni esistono?

- $|Q|$ stati
- $\phi(n)$ posizioni per la testina
- 2 simboli per ciascuna delle $\phi(n)$ posizioni

Il numero dei passi è $T(n) \leq |Q|\,\phi(n)\,2^{\phi(n)}$



$\mathcal{P}$ e $\mathcal{NP}$ per altri problemi

- problemi di ricerca: $\mathcal{FP}$ e $\mathcal{FNP}$ ($\mathcal{F}$ sta per *Find*)
- problemi di ottimizzazione: $\mathcal{OP}$ e $\mathcal{ONP}$
- problemi di conteggio: $\#\mathcal{P}$ e $\#\mathcal{NP}$

Se il contesto è chiaro, spesso si omette il prefisso