Esercitazione 1

(1) $f(n) = n \log n \leq n \cdot n \quad \forall n \geq 10$

① $= O(n^2)$ ✓ perché O è un upper bound perché $\log n$ cresce meno di n

② $= \Omega(\log^2 n)$ ✓ $c = 1$ $\quad n_0 \geq 10$ $\quad n \log n \geq \log n^2$

③ $= o(n^2)$ non possono crescere uguali

ovvero $\lim \frac{n \log n}{n^2} = 0$ ✓

La funzione ha queste 3 proprietà

(4) $A1 \in P$ complessità di A $O(n \log n)$ $O(n^k)$ $k = 2$ o anche $k = 1.1$ k dipende solo dalla taglia del problema

(2.) Studiamo 1B

$\exists$ A1 per B $\Theta(2^n)$ possiamo dire $B \notin P$

No non possiamo affermarlo con sicurezza perché potrebbe esistere un altro algoritmo

(3) A* : il miglior algoritmo possibile ha $\Theta(n^{\log n})$ allora $B \notin P$ ✓

- $n^{\log n}$ non è polinomiale perché non c'è nessuna costante k fissata che mi permette di risolvere il problema in tempo n^k dove k non dipende dalla taglia del problema

Essendo questo il miglior algoritmo possibile $B \notin P$

Per qualche n $\log n$ è peggio di k costante

(4) istanze sono triple di numeri

A $\quad I(A1) = \{x, y, z \mid x, y, z \in \mathbb{N}\} \quad A1(x, y, z) = f(x, y, z)$

$A_{\text{Algoritmo}} \quad T^A(x, y, z) = O(y \log(x+z))$

a. $A \in P$?

b. $A \notin P$?

Ovvero $\exists$ A per A1 $\Rightarrow A1 \in P$ *

$\exists$ A per A1 $\Rightarrow A1 \notin P$ Si coramente No

4

No perché la complessità di A è esponenziale

- Se esiste algoritmo polinomiale l'implicazione è falsa
- Se non è polinomiale potrei comunque averne un altra che è polinomiale

* Devo dire se:

$O(y \log(x+z)) = O((\text{taglia dell'input})^k)$

questo è ciò che mi serve per scrivere quei numeri

taglia $\|(x, y, z)\| = \log x + \log y + \log z$

non è polinomiale ho y e non log y

$y \log(x+z)$

(5) Se ci dicono che so che questo è il miglior algoritmo so che $A1 \notin P$ perché l'algoritmo è esponenziale

La taglia è logaritmica nei numeri

L'algoritmo funziona in tempo esponenziale nella taglia

Se y = un miliardo taglia è 30 perché servono 30 cifre l'algoritmo ci mette un miliardo di passi invece

MIDSIZE CLIQUE

Input: $G = (V, E)$

Output: yes $\iff$ $\exists$ clique Q di G

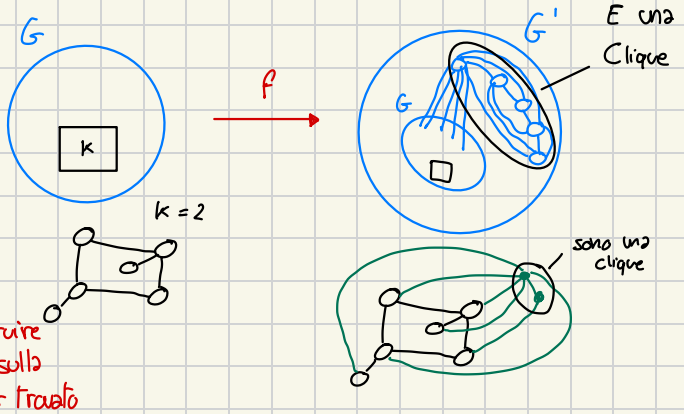
$$|Q| \geq \frac{|V|}{2}$$

Proviamo la riduzione a NP-completo

CLIQUE $\leq$ MID-SIZE CLIQUE

$$G = (V, E), K \xrightarrow{f} G' = (V', E')$$

$|V| = 2r$ ovvero pari



non posso costruire la riduzione sulla base di aver trovato la soluzione qui perché sto assumendo che il problema è NP e quindi che calcolarne la soluzione è difficile (ovvero non polinomiale)

$G' = (V', E')$

$V' = V \cup T$

Aggiungiamo questi super nodi

$|T| = t = \frac{V}{2} - K + 1$

$$G' = \begin{cases} G & K > \frac{|V|}{2} \\ \bullet & K \leq \frac{|V|}{2} \end{cases}$$

Trasformazione vera e propria

$T = 3 - 2 + 1 = 2$ nodi aggiunti

Ogni clique ora è una clique assieme a quella formata dai super nodi perché sono interamente connesse tra loro (la clique dei supernodi ha taglia t)

Perciò esiste una clique di taglia $\frac{|V|}{2} + 1$ in G se e solo se ne esiste una di taglia K in G

poiché $|t| = \frac{V}{2} - K + 1$ nodi (taglia supernodi aggiunti)

Errore sul numero di nodi ne uno aggiunti:

dimensione reale clique nel nuovo grafo

$$t + k = \frac{V + t}{2} + 1$$

ma allora per avere taglia $\frac{V}{2} + 1$ della clique devo avere una clique di taglia K in G perché in t da sola non ci può stare

$$t = |V| + 2 - 2K$$

Ecco perché

$$|t + k| = \frac{V + t}{2} + 1$$

$$2t + 2k = V + t + 2$$

$$t = V + 2 - 2k$$

(2) Provare che se $P = NP$ allora

reachability è NP completo

Input $G=(V,E)$ $s,t \in V$
Output yes $\iff \exists\ s \rightsquigarrow t$

Reach $\in$ NP
$\forall A \in NP \quad A \leq Reach$

Se $P = NP$ io so dire sempre sì e no

Reach $\in P$ (BFS) $\subseteq NP$ ✓ indipendentemente
Reach $\in P = NP \ni$ Reach

$x \in \mathcal{I}(A) \rightarrow (G, s, t)$
t.c
se $A(x)$ è yes $\Rightarrow \exists\ s \rightsquigarrow t$
se $A(x) =$ no $\Rightarrow \nexists\ s \rightsquigarrow t$

$$f(x) = \begin{cases} G=(V,E),\ V=\{s,t\},\ E=\{s \rightsquigarrow t\} & \text{se } A(G)=\text{yes} \\ G=(V,E),\ V=\{s,t\},\ E=\{\} & \text{se } A(x)=\text{no} \end{cases}$$

$A(G)$=yes polytime
$A(x)$=no polytime se $P=NP$

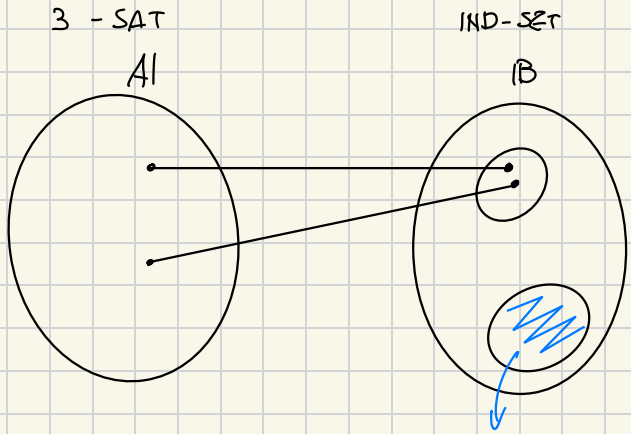
funzione di riduzione → polinomiale, i grafi sono due nodi

A è NP che è uguale a P e quindi calcolabile polinomialmente

Qui usiamo la soluzione perché assumo che il problema è in P

3. Fatto a lezione

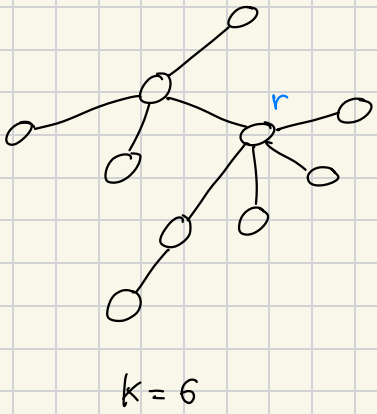
4.



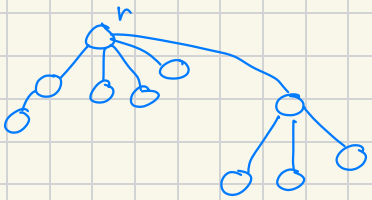
non posso usare questa istanza per risolvere 3-SAT perché sono semplici, ad esempio in SUBSET SUM

- Tree indipendendent set

Abbiamo un grafo senza cicli (quero un albero) vogliamo mostrare che in questo caso la riduzione a 3-SAT non esiste e che quindi il problema è risolvibile in politime



radichiamo l'albero

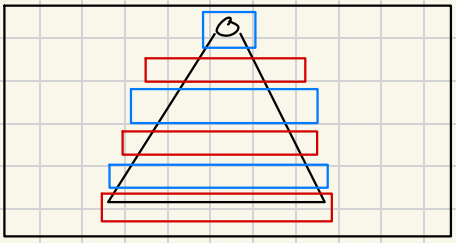


- Trovare un indipendent set I

$|I|$ max

$|I| \geq k$ yes

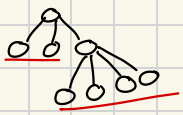
$|I| < k$ no



Somma dei rossi è indipendent set

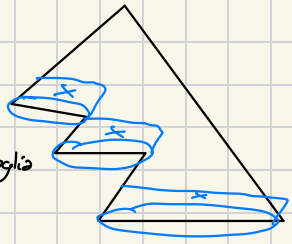
Somma dei blu è indipendent set

ma non funziona perché potrei avere



avrei 6 ma mi dà 4. Mi offre uno spunto:

Tutte le foglie fanno parte dell' independent set di taglia massima



Infatti:
- Se nell' IS di taglia massima prendo il padre e non la foglia li scambio
- Se non prendo nessuno dei due posso estendere l' IS con la foglia

$\exists I^*$ t.c I^* contiene tutte le foglie

I^* ↓ indipendent set di taglia massima

In I^* non avrò i genitori

Li rimuovo dall' albero e riparto

In sostanza:

faccio BFS quelli che non mi danno figli li includo nell' indipendent set rimuovo il padre e loro dall' albero poi rilancio sul nuovo albero

Indipendent set sugli alberi è un problema facile

Only Small Independent Set

Input G, k
Output yes $\iff \forall I$ ind set di G $|I| \leq k$

Vogliamo mostrare che se so risolvere in tempo polinomiale questo problema allora $P = NP$

- Ho $P = NP$ se $\exists A$ per Only small Independent Set con A polytime

$$\frac{A|}{OSIS} \quad \quad \frac{|B}{} = \text{INDSET}$$

$$A|(G,k) = \overline{B}(G,k)$$

Questo è l'opposto di: indipendent set.

Se ho un algoritmo che dice sì e no in tempo polinomiale a OSIS per assunzione
Ne ho uno che mi dice pure sì e no per l'indipendent set che è l'opposto.
Ma quindi indipendent set è P ed essendo NPC tutte collassa

⑥ Proviamo che se $\underbrace{P = NP}_{\text{Decisione}} \Rightarrow \exists A$ polytime $A(G=(V,E)) = k^*$ t.c. $\exists$ ind. set. $I \subseteq V$ $|I| = k^*$

$\forall I'$ ind. set di G $|I'| \leq k^*$

se ho $B(G=(V,E), k) = \begin{cases} \text{yes} & \text{se } G \text{ ha } I \text{ di taglia } k \\ \text{no} & \text{altrimenti} \end{cases}$

Algoritmo(G):

```
i = 1
while (B(G, i) = yes)
        i = i + 1
return i - 1
```

per ottenere l'indipendent set se ho questo algoritmo prendo un grafo e mi metto su un nodo e lo tolgo, mi chiedo se c'è ancora indset di massima, se sì tolgo il nodo, se no faccio una nuova chiamata mantenendo questo nodo e rimuovendone un altro

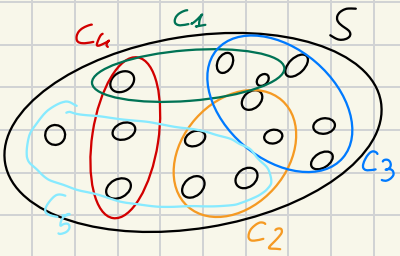
7 Min Weight set Cover

Abbiamo un insieme di elementi con vari sottoinsieme di tale insieme

Ogni set ha un costo

Vogliamo una sotto famiglia che copra tutti gli elementi dell'universo che abbia peso minimo

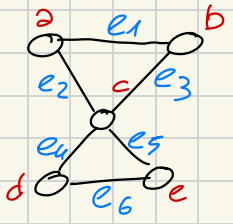
ad es: $\{c_2, c_3, c_4, c_5\}$

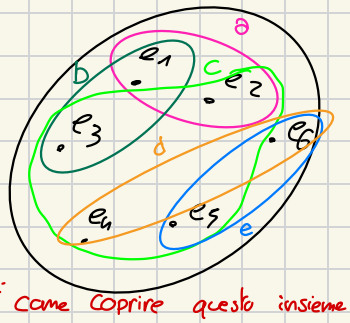


Vogliamo provare che è NP completo

Similare a vertex cover: Vedo gli insiemi come i vertici

Posso costruire da questo problema una soluzione per vertex cover





Per ogni insieme do peso 1

W lo metto a K Se ho una coperfura K allora il peso totale degli insiemi presi sarà K che è uguale a W

Coprire con i vertici questi archi

È come coprire questo insieme con i sottoinsiemi

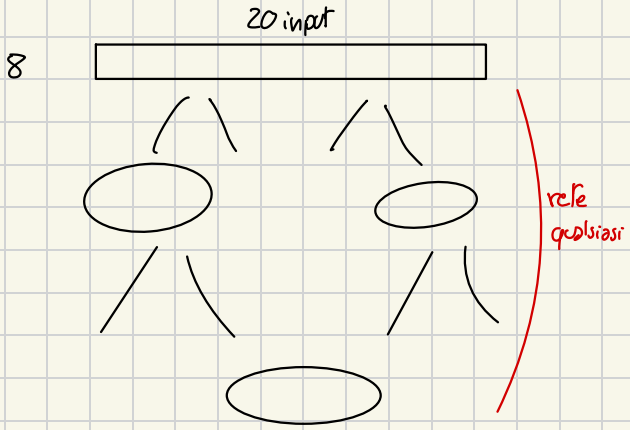
Ho trasformati gli archi in punti e i vertici negli insiemi di punti che riesco a coprire con quei vertici

(punti: 1 o archi)

Ho un istanza di Vertex Cover e voglio trasformarla in un istanza di Min Weight Subset Sum prendo l'istanza e la trasformo come appena detto

Min Weight SUBSET SUM è la versione più completa di vertex Cover

(8) Ho Circuit 20-SAT voglio mostrare che sta in P e perché



provo tutti gli input

$2^{20} \cdot n$

costante → non dipende dalla taglia del problema

costo di propagazione dei valori (svolgere la formula) è la taglia del problema → ovvero la verifica

Ho che A NPC e che C NPC

Da ① $A \leq B$ $\Rightarrow$ B NPC
② $B \leq C$

Quindi voglio che • $B \in NP$ ✓ $B \leq C$ con C che è NPC e quindi pure NP

• $\forall E \in NP$ $E \leq B$ ✓ Se so che A è NPC ogni problema NPC si riduce ad A ma allora se B si riduce ad A per transitività ogni problema si riduce a B perciò è NP hard

Esercizi a fine lezione 30-05-2024

Dire si o no alle affermazioni

① TWO-PLAYERS-SAT $\in$ NPSPACE($\log(n)$) $\Rightarrow$ Tautology $\leq$ SAT

Se Two-players-SAT $\in$ NL

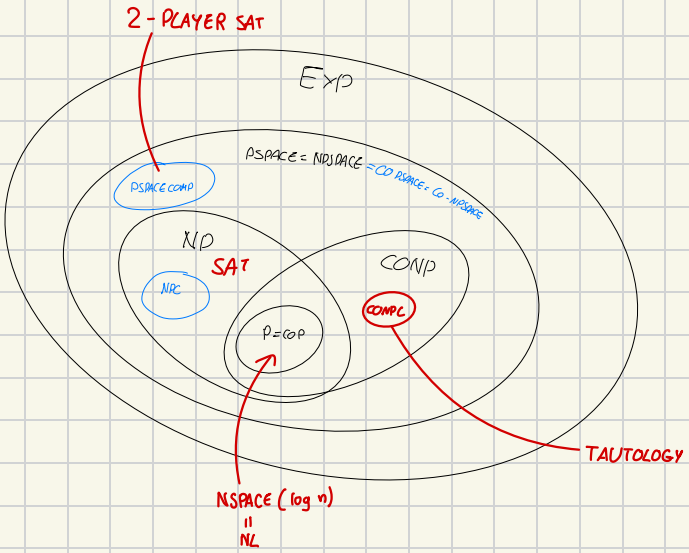
ho: PSPACE = P (tutto ciò che sta in mezzo collassa) $\Rightarrow$ CONP = P

$\Rightarrow \forall A \in$ CO-NP essendo che CONP = P

$A \in NP \Rightarrow A \leq SAT$

A = TAUTOLOGY

Perciò l'implicazione è vera

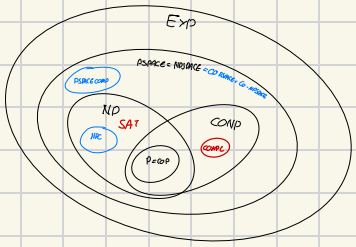


② Se TAUTOLOGY $\leq$ TWO-PLAYER-SAT $\Rightarrow$ CO-NP = PSPACE

essendo Tautology $\in$ CO-NP e PSPACE e two player sat è PSPACE-complete l'affermazione è sempre vera indipendentemente

Perciò se questa può essere falsa l'implicazione è falsa

Questo può essere non vero banalmente anche nel caso che riteniamo più plausibile, ovvero $\rightarrow$



Perciò l'implicazione è falsa

③ Se $\exists$ Al t.c. Al $\in$ NPC e Al è PSPACE-hard $\Rightarrow$ PSPACE = P

Se vera la premessa ho PSPACE = NP perché tutti i problemi PSPACE si riducono a un problema NPC ma continuo a non sapere se P=NP e nulla vieta che P $\neq$ NP com'è più plausibile pensare

Perciò PSPACE = NP $\neq$ P $\Rightarrow$ P = PSPACE

Perciò l'implicazione è falsa

④ Se $\exists$ Al t.c. Al $\in$ NPC e Al è PSPACE-hard $\Rightarrow$ PSPACE = NP

Perciò l'implicazione è vera per quanto detto prima

⑤ Se A é NPC e B é NPC con $\mathcal{I}(A) = \mathcal{I}(B)$ e $\mathcal{C} = A(x) \wedge B(x) \Rightarrow \mathcal{C}$ é NPC

Se penso a: CLIQUE $(G, \frac{|V|}{2}+1) = A(x)$
INDSET $(G, \frac{|V|}{2}+1) = B(x)$

L'and tra le due e quindi l'intersezione é sempre falsa perció il problema $\mathcal{C}$ non é NPC
perché allo stesso tempo non posso avere sia una clique di più della metà dei vertici
sia un indipendent set di più della metà dei vertici

$$|V| - \left(\frac{|V|}{2}+1\right) < \frac{|V|}{2}$$

Perció l'implicazione é falsa

Se avessi l'OR possiamo arrivare ad un assurdo simile

⑥ Se Indipendent Set é PSPACE completo allora NP = CO-NP

PSPACE diventa uguale a NP essendo Indipendent Set NPC, tutto ció tra PSPACE e NP collasa perció ho

NP = PSPACE = CO-NP e quindi NP = CO-NP

Perció l'implicazione é vera

$$\text{Clique} = \overline{\text{Ind Set}}$$

$$\text{SAT} \leq \text{Ind Set}$$

Esercizio 4

3-SAT-K

3-SAT-2 $\in P$

n = num_variabili

```
for i in range m
    for j in variables_in_clause_m
        if xj = negato:
            arr[j].add(!)
        else
            arr[j].add(?)

for i in range m:
    Se almeno una delle tre variabili nella clausola appare sempre
    normale metto quella a T

    Se almeno una delle tre variabili nella clausola appare sempre
    negata metto quella a F

    Se tutte e tre appaiono una negata e una a true metto
    a false quelle negate
```

SAT $\longrightarrow$ 4

$x_1 \wedge (x_1 \wedge x_4) \wedge (x_4$

$\overline{x_1} \vee \overline{x_4}$