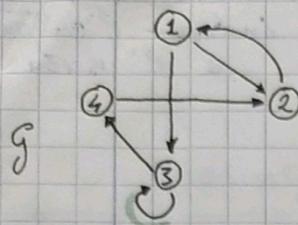


## GRAFO (ORIENTATO)

relazione binaria di vertici



coppie  $G = (V, E)$

$\rightsquigarrow V = \{1, \dots, n\}$  insieme nodi/vertici

$E \subseteq V \times V$

coppia/self loop!

$$G = (\underline{\{1, 2, 3, 4\}}, \{(1, 2), (2, 1), (1, 3), (3, 1), (2, 3), (3, 2)\})$$

## GRAFO (NON ORIENTATO)

$a R b \Rightarrow b Ra$

$G = (V, E)$  se  $(u, v) \in E \Leftrightarrow (v, u) \in E$

Indirizzi u e v

Fare da u a v

NON ci sono coppie!!!

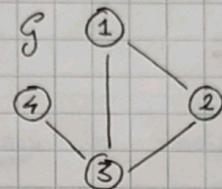
altra notazione

$$(3, 4) \neq (4, 3)$$

$$\{3, 4\} = \{4, 3\} \rightarrow G = (V, E) \quad E \subseteq \binom{V}{2} \text{ insieme d'insiemi}$$

$\rightarrow V = \{1, 2, 3\}$  i sottoinsiemi di 2

$\rightarrow \binom{V}{2} = \{\{1, 2\}, \{1, 3\}, \{2, 3\}\}$  Jornoli degli elementi di V



$G = (V, E)$

$$V = \{1, 2, 3, 4\}$$

$$E = \{\{1, 2\}, \{1, 3\}, \{2, 3\}, \{3, 4\}\}$$

G grafo orientato con  $|V| = n$  vertici  $\Rightarrow$  massimo numero di archi  $|E| = V \times V = n^2$

G grafo NON orientato con  $|V| = n$  vertici  $\Rightarrow$  massimo numero di archi  $|E| = \binom{n}{2} = \frac{n(n-1)}{2} = n^2$

due vertici sono ADIACENTI se esiste un arco che li collega

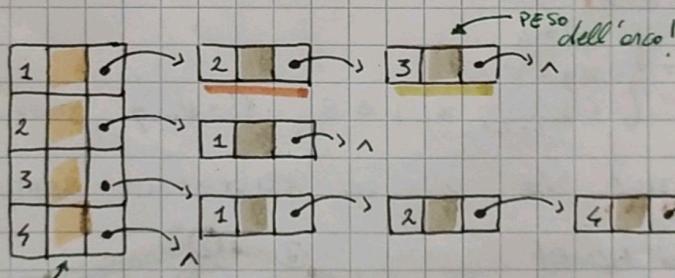
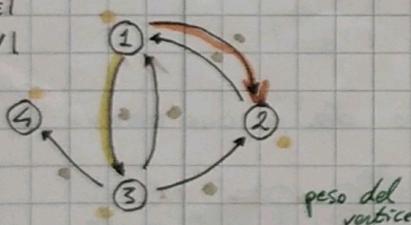
L'ARCO  $(u, v)$  è INCIDENTE ai vertici u e v



## (i) LISTA DI ADIACENZA

$$m = |E|$$

$$n = |V|$$



meglio per grafici SPARSI  
spazio in memoria  $n+m$

GRAFO SPARSO : (pochi archi) numero di archi  $\ll n^2 (\approx n)$

GRAFO DENSO : (molti archi) numero di archi  $\approx n^2$

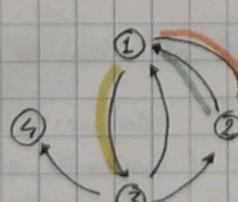
## (ii) MATRICE DI ADIACENZA

$$G = (V, E)$$

$$n = |V|$$

$$A_G = (a_{ij}) \rightarrow n \times n [n^2 \text{ elementi}]$$

$$a_{ij} = \begin{cases} 0 & \text{se } (i, j) \notin E \\ 1 & \text{se } (i, j) \in E \end{cases}$$



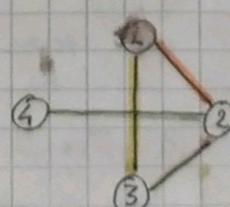
$$A = \begin{bmatrix} 1 & 2 & 3 & 4 \\ 1 & 0 & 1 & 0 \\ 2 & 2 & 0 & 0 \\ 3 & 1 & 2 & 0 \\ 4 & 0 & 0 & 0 \end{bmatrix}$$

se gli i pesi molto  
gari vali al posto di  
uno! (in caso di  
peso 0 usa  $\infty$  per  
indicare l'assenza di  
arco)

Se ho un grafo sparso  $\Rightarrow$  matrice SPARSA

Se ho un grafo DENSO  $\Rightarrow$  matrice Densa!

svantaggio: ricerca esistenza arco  
è COSTANTE



$$A = \begin{bmatrix} 0 & 1 & 1 & 0 \\ 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

$$a_{ij} = a_{ji} \quad f_{ij} = f_{ji}$$

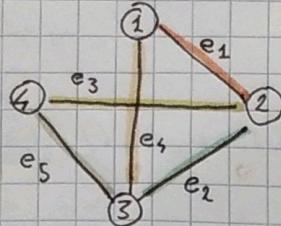
matrice SIMMETRICA!

### (iii) MATRICE DI INCIDENZA

$$n = |V|$$

$$m = |E|$$

$$M = n \times m$$



$e_1$  è  
INCIDENTE ai  
vertici 1 e 2

	$e_1$	$e_2$	$e_3$	$e_4$	$e_5$
1	1	0	0	1	0
2	1	1	1	0	0
3	0	1	0	1	1
4	0	0	1	0	1

$i \rightarrow j$   
è uscita +1 e -1 per  
indicare l'entrata  
e l'uscita

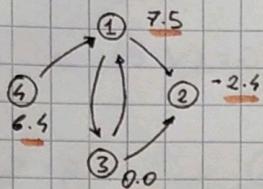
meglio per grafi sparsi!

### • GRAFI PESATI

#### (i) SUI VERTICI

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

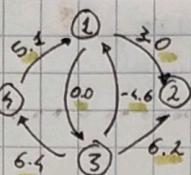
$$W: V \rightarrow \mathbb{R}$$



#### (ii) SUI GLI ARCHI

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

$$W: E \rightarrow \mathbb{R}$$

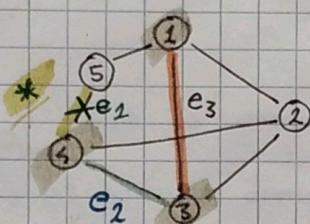


#### (iii) SU ENTRAMBI

### • SOTTOGRAFO

$$G = (V, E)$$

$$V = \{1, 2, 3, 4, 5\}$$



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

$$V' \subseteq V$$

$$\begin{cases} V' = \{1, 3, 4\} \\ E' = \{e_1, e_3\} \end{cases}$$

$\text{NON posso prendere } e_2!$

• SOTTOGRAFO INDOTTO :  $G[V'] = \{ E' = E \cap V' \times V' \}$   $\rightarrow$  devo prendere TUTTI gli archi!  
(anche  $e_2$  quindi)

### • CAMMINO TRA $u$ e $v$

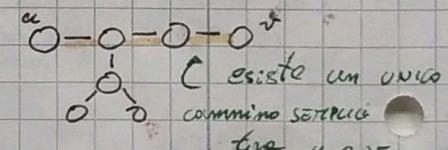
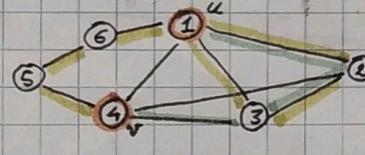
$$\langle x_0, x_1, \dots, x_q \rangle = \begin{cases} x_0 = u \\ x_q = v \\ (x_{i-1}, x_i) \in E \quad i=1 \dots q \end{cases} \rightarrow \text{esiste l'arco!}$$

•  $v$  è RAGGIUNGIBILE da  $u$  se ESISTE un CAMMINO da  $u$  che arriva a  $v$

(i) CAMMINO SEMPLICE (vertici distinti)

(ii) CAMMINO NON SEMPLICE (vertici ripetuti)

(iii) CICLO : cammino dove  $x_q = x_0$

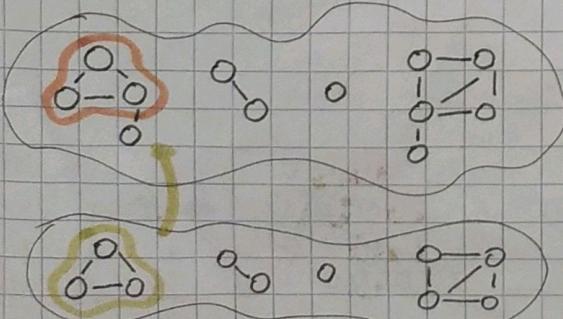


C'è esiste un UNICO  
cammino SEMPLICE  
tra  $u$  e  $v$

•  $G = (V, E)$  è CONNESSO se  $\forall u, v \in V$  :  $v$  è RAGGIUNGIBILE da  $u$

•  $G = (V, E)$  è ACICLICO se NON esistono cicli all'interno del graf

• COMPONENTE CONNESSA sottoinsieme di vertici  $V' \subseteq V$  :  $\begin{cases} G[V'] \text{ è CONNESSO} \\ V' \text{ non è contenuto strettamente in un sottoinsieme di } V \text{ che sia a sua volta connesso} \end{cases}$  [deve essere massimale]



Ogni singolo grafo da solo è una COMPONENTE CONNESSA

ma però NON ha un sottoinsieme più grande  
che sia connesso!

il sottografo indotto è connesso

i sottografi sono CONNESSI  
ESISTE un sottoinsieme più grande che lo contiene  
quindi NON è una componente connessa!



**LEMMA della STRETTA di ERNST** : La somma dei gradi è pari a 2 volte il numero di archi

$$G = (V, E)$$

NON orientato

$$n = |V|$$

$$m = |E|$$

$$\sum_{u \in V} \deg(u) = 2m$$

$$m = \frac{1}{2} \sum_{u \in V} \deg(u)$$

dimostrazione matrice adiacenza

$$V = \{1, 2, \dots, n\}$$

$$\deg(v_i) = \sum_{j=1}^n a_{ij} \quad \left( = \sum_{j=1}^n a_{ji} \text{ per simmetria} \right) \Rightarrow \sum_{i=1}^n \deg(v_i) = \sum_{i=1}^n \left( \sum_{j=1}^n a_{ij} \right) = \sum_{i=1}^n \sum_{j=1}^n a_{ij} = 2m$$

**PROPRIETÀ** : in un grafo NON orientato  $G$ , il numero di vertici di grado DISPARI è sempre PARI

dimostrazione :

$$G = (V, E) \text{ NON orientato} \quad \left\{ \begin{array}{l} P = \{u \in V \mid \deg(u) \text{ è pari}\} \\ S = \{u \in V \mid \deg(u) \text{ è dispari}\} \end{array} \right.$$

suddivido il grafo in due parti

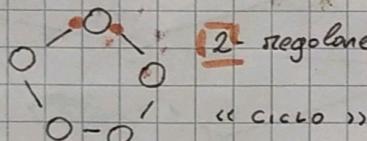
[applichiamo lemmi delle strette di meno]

$$2m = \sum_{u \in V} \deg(u) = \sum_{u \in P} \deg(u) + \sum_{u \in S} \deg(u) = \sum_{u \in P} 2h(u) + \sum_{u \in S} (2h(u) + 1) =$$

$$= 2 \sum_{u \in P} h(u) + 2 \left( \sum_{u \in S} h(u) \right) + |S| = \text{sommo 1 tante volte quanti sono i vertici di grado dispari } \sum_{u \in S} 1 = |S|$$

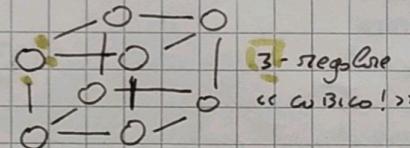
$$= 2 \sum_{u \in P} h(u) + |S| \quad \text{ma} \quad |S| = 2m - 2 \sum_{u \in P} h(u) = 2(m - \sum_{u \in P} h(u)) \Rightarrow |S| \text{ è pari!}$$

$G = (V, E)$  NON orientato  $\rightsquigarrow G$  è K-regolare se  $\forall u \in V : \deg(u) = k$   
K intero



$G$  2-regolare  $[\deg(u) = 2]$

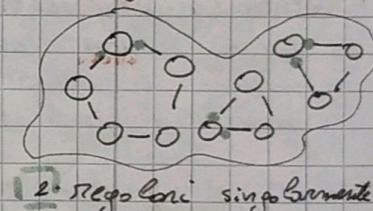
$$2m = \sum_{u \in V} \deg(u) = 2n \Rightarrow m = n$$



$G$  3-regolare  $[\deg(u) = 3]$

$$2m = \sum_{u \in V} \deg(u) = 3n \quad \begin{matrix} \text{numero} \\ \text{pari} \\ \text{di} \\ \text{vertici} \end{matrix}$$

$$2m = 3n = 2n + n \Rightarrow n = 2(m-n)$$



**PROPRIETÀ**  $G = (V, E)$  NON orientato SENZA vertici isolati

suppongo che  $|E| = |V| - 1$  ( $m = n-1$ )

Allora esistono ALMENO due vertici con grado uguale a 1!

«VETICI TERMINALI»

$$\text{ipotesi: } m = n-1 \Rightarrow 2n-2 = 2m = \sum_{u \in V} \deg(u) = \sum_{u \in V \setminus V_1} \deg(u) + \sum_{u \in V \cap V_1} \deg(u) =$$

$$* V_1 = \{u \in V \mid \deg(u) = 1\} \text{ } u \text{ è terminale} \quad = |V_1| + \sum_{u \in V \setminus V_1} \deg(u) =$$

voglio dimostrare che  $|V_1| > 1$

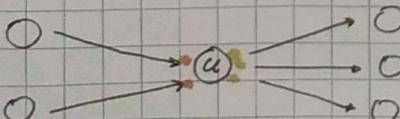
Tutti gradi = 1      Tutti gradi ≥ 2 qua      vi è vertice isolato ( $\deg = 0$ ) per ipotesi

$$* \geq |V_1| + 2|V \setminus V_1| = 1|V_1| + 2(n-1-1) = |V_1| + 2|V_1| - 2|V_1| \Rightarrow |V_1| \geq 2$$

$$2n-2 \geq |V_1| + 2n - 2|V_1| \Rightarrow |V_1| \geq 2$$

**GRAFI ORIENTATI**

$$\sum_{u \in V} \text{in-deg}(u) = \sum_{u \in V} \text{out-deg}(u) = m$$

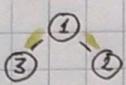


• in-deg(u) = 2 numero di archi in ingresso

• out-deg(u) = 3 numero di archi in uscita

## • ISOMORFISMO DEI GRAFI

$$G'$$
  
 $V' = \{1, 2, 3\}$



$$G''$$
  
 $V'' = \{a, b, c\}$

② - ③ - ④

in Teoria NON sono uguali MA se considero le RELAZIONI (ARCHI!) tra i vertici ci allora sono uguali

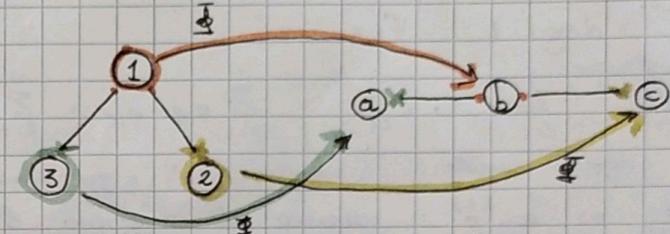
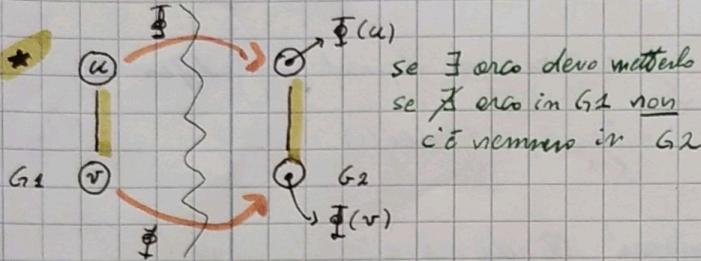
Dati due grafi:  $G_1 = (V_1, E_1)$  e  $G_2 = (V_2, E_2)$ , una funzione  $\Phi: V_1 \rightarrow V_2$  è un ISOMORFISMO se:

(i)  $\Phi$  è BIETIVA

(ii) preserva la RELAZIONE fra i vertici  $\Rightarrow$  se ho un arco in  $G_1$  devo avere anche in  $G_2$

$$\forall u, v \in V_1 : (u, v) \in E_1 \Leftrightarrow (\Phi(u), \Phi(v)) \in E_2$$

$G_1$  e  $G_2$  sono ISOTORRFI ( $G_1 \cong G_2$ ) se esiste un isomorfismo fra  $G_1$  e  $G_2$  non è detto che sia unico!



gli archi se esistono ci sono, se non ci sono non ci sono altra ec.

$\Phi$  è un ISOTORRFISMO e  $G'$  e  $G''$  SONO ISOTORRFI

• CONDIZIONI NECESSARIE (non sufficienti)

$$G_1 = (V_1, E_1) \quad G_2 = (V_2, E_2)$$

- (i)  $|V_1| = |V_2|$  l'isomorfismo è una funzione biiettiva  
(ii)  $|E_1| = |E_2|$   $\Phi$  preserva la relazione fra vertici

(iii)  $\deg\text{-seq}(G_1) = \deg\text{-seq}(G_2)$

$$u \in V_1 \xrightarrow{\Phi} \Phi(u) \in V_2$$
  
 $\deg(u) = \deg(\Phi(u))$

(iv)  $cc(G_1) = cc(G_2)$   
C. numero d. componenti connesse

(v)  $w(G_1) = w(G_2)$  CLIQUE NUMBER!

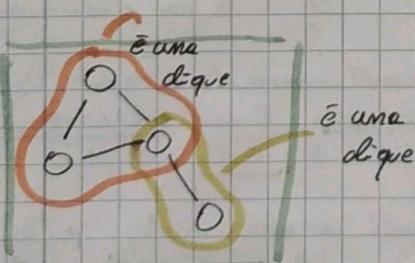
## • CLIQUE

$$G = (V, E)$$
 non orientato

clique

$$(C \subseteq V \text{ t.c. } G[C] \text{ è completo})$$

il grafo  
NON  
è una clique  
(nessuno indi)



è una clique  
MA NON È  
MASSIMA perché  
esiste una clique  
più grande che  
la contiene!

È una clique  
MASSIMA

è massima  
ma non massima!

$$\sim w(G) = 3$$

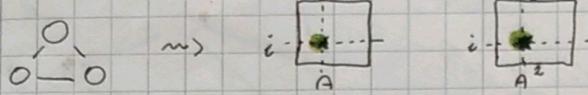
(a) CLIQUE MASSIMA: sottoinsieme di vertici all'interno di un grafo che forma una clique (non ne esiste una più grande)

(b) CLIQUE MASSIMALE: se NON è sottoinsieme di una clique più grande

• CLIQUE NUMBER: cardinalità della sua CLIQUE MASSIMA



$G$  non orientato contiene un TRAVERSOLI (d'ogni d. 3 vertici) se e solo se esistono due indici (vertici)  $i$  e  $j$  t.c. sia  $a_{ij}$  che  $a_{ji}$ <sup>(2)</sup> sono NON NULLI



bols!

$G$  è CONNESSO  $\Rightarrow |E| \geq |V| - 1$

$$G = (V, E)$$

$$n = |V|$$

$$m = |E|$$

dimostrazione per assurdo

Suppongo che  $G$  NON sia connesso  $\Rightarrow$  ci sono almeno 2 componenti connesse



$$\begin{aligned} n &= |V| \geq |V_1| + |V_2| \geq \left(\frac{n-1}{2} + 1\right) + \left(\frac{n-1}{2} + 1\right) = n+1 > n \\ |V_1| &\geq \frac{n-1}{2} + 1 \quad \text{c'è anche } x \\ |V_2| &\geq \frac{n-1}{2} + 1 \quad \text{c'è anche } y \end{aligned}$$

ASSURDO

### GRAFI ACICLICI [foreste]

$$G = (V, E) \text{ non orientato} \rightsquigarrow$$

dimostrazione su  $n = |V|$

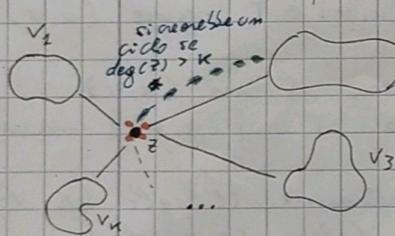
CASO BASE: è banale che  $(n=3)$

PASSO INDUTTIVO  $n \geq 4$

prendo  $G = (V, E)$  aciclico con  $n = |V|$

prendo  $z \in V$  e lo rimuovo  $\Rightarrow G' = G[V \setminus \{z\}]$

sia  $k$  il numero d. componenti connesse d.  $G'$



$$\forall i=1 \dots n : |E_i| \leq |V_i| - 1$$

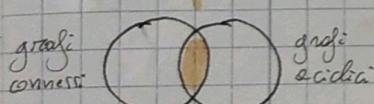
$$\begin{aligned} |E| &= \sum_{i=1}^k |E_i| + \deg(z) \leq \sum_{i=1}^k (|V_i| - 1) + \deg(z) = \\ &= \sum_{i=1}^k |V_i| - k + \deg(z) = |V| - 1 + \deg(z) - k \end{aligned}$$

$$\leq |V| - 1$$

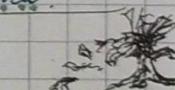
$$\deg(z) - k \leq 0 ?$$

$\deg(z) \leq k$  per forza, altrimenti si creerebbe un ciclo!

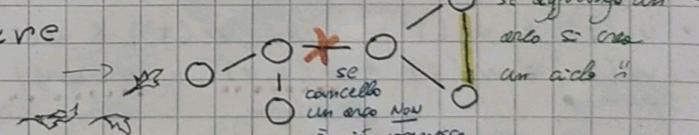
ALBERO!  $G$  ACICLICO e CONNESSO  $\Rightarrow |E| = |V| - 1$



sono strutture  
fragili!



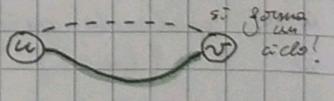
se aggiunge un  
arco si crea  
un ciclo



### PROPRIETÀ

Sia  $G = (V, E)$  non orientato allora le seguenti affermazioni sono EQUIVALENTE (sono dei se e solo se)

- (a)  $G$  è un ALBERO
- (b) due vertici qualsiasi di  $G$  sono connessi da un UNICO CAMMINO semplice
- (c)  $G$  è CONNESSO, ma se qualunque arco è rimosso da  $E$ , il grafo risultante è DISCONNESSO
- (d)  $G$  è CONNESSO e  $|E| = |V| - 1$
- (e)  $G$  è ACICLICO e  $|E| = |V| - 1$
- (f)  $G$  è ACICLICO, ma aggiungendo un qualsiasi arco ad  $E$ , il grafo risultante è circolo

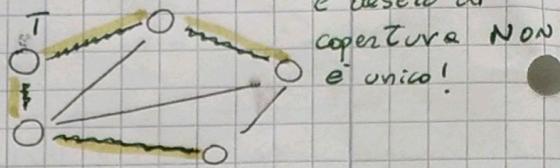


## • ALBERI DI COPERTURA MINIMA « MST : minimum spanning tree »

$G = (V, E)$  non orientati e connessi

### • Alberi di copertura [ST]

$T \subseteq E$  t.c.  $(V, T)$  è un albero (tocco tutti i vertici della ~~del~~ del grafo di pertinenza)



### • MST : albero di copertura ~~minima~~ con PESO MINIMO

$G = (V, E, w)$  non orientato e connesso

↓

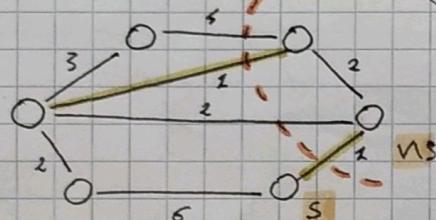
$$w: E \rightarrow \mathbb{R} \quad w(T) = \sum_{(u,v) \in T} w(u,v)$$

$T \subseteq E$  è MST se il peso dell'albero di copertura è

il minimo fra tutti gli alberi

MST :  $\min w(T) \wedge T \text{ è ST}$

### • Fatto circolare degli MST



• TAGLIO : partizionamento del grafo in due parti  $(S, V \setminus S)$   $S \subseteq V$

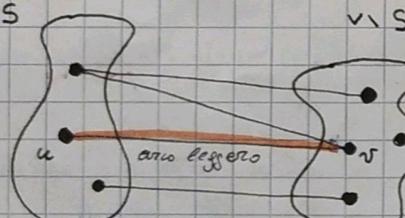
• ARCO LEGGERO attraversa il taglio e deve avere peso minore fra tutti quelli che lo attraversano!

• L'arco leggero che attraversa il taglio appartiene per forza ad ad un MST

$$(u, v) \in E : (u \in S \wedge v \in V \setminus S) \text{ or } (u \in V \setminus S \wedge v \in S) \Rightarrow w(u, v) \leq w(x, y) \Rightarrow (u, v) \text{ attraversa il taglio}$$

### • TECNICA dei CUCI - E - TAGLIATI

sia  $T$  un MST di  $G \Rightarrow (u, v) \in T$  oppure  $(u, v) \notin T$



(i) inizio da  $T$  che non contiene l'arco  $(u, v)$

(ii) aggiungo l'arco all'albero  $T \cup \{(u, v)\}$  creando

(iii) so che  $\exists (x, y) \in T$  che attraversa il taglio  $(S, V \setminus S)$   
se  $w(u, v) < w(x, y)$  tolgo  $(x, y)$  da  $T$

•  $T \cup \{(u, v)\} \setminus \{(x, y)\} = T$   $\Rightarrow$  è un albero di copertura  
(ST) ed è pure  
nuovo! (MST)

$$w(T) \leq w(T') \wedge w(T') \leq w(T) \Rightarrow w(T) = w(T')$$

$$w(T') = w(T) + w(u, v) - w(x, y) \leq w(T)$$

arco leggero attraversa il taglio

### • ARCO SIGNORO

$A \subseteq$  contenuto in qualche MST, un arco  $(u, v) \in A$   $\Rightarrow (u, v) \notin A$  si dice signoroso per  $A$   
se  $A \cup \{(u, v)\}$  è contenuto in qualche MST.

### • GENERIC-MST ( $G, w$ )

$$A \leftarrow \emptyset$$

while  $|A| < |V| - 1$

- Trova un arco signoroso per  $A$   $(u, v)$
- $A \leftarrow A \cup \{(u, v)\}$

return  $A$

Un taglio **RISPETTA**  $A$  se nessuno degli archi di  $A$  attraversa il taglio!

## • TEOREMA FONDAMENTALE MST

sia  $G = (V, E, w)$  un grafo NON orientato, pesato sugli archi e connesso, e siano:

- (a)  $A \subseteq E$  contenuto in qualche MST
- (b)  $(S, V \setminus S)$  taglio che ~~RISPETTA A~~
- (c)  $(u, v) \in E$  ARCO LEGGERO che attraversa il taglio  $(S, V \setminus S)$

allora l'arco  $(u, v)$  è **SICURO** per  $A$ ! ( $\exists$  mst che contiene  $A \cup \{(u, v)\}$ )

dimostrazione TESI:  $\exists$  mst che contiene  $A \cup \{(u, v)\}$

~~prova con risurrezione di Archimede~~

sia  $T \subseteq E$  un mst che contiene  $A$  [ipotesi (a)]

CASO 1.

$(u, v) \in T$  allora sicuramente  $A \cup \{(u, v)\} \subseteq T$ , anche se ho un unico arco leggero

CASO 2.

$\downarrow (u, v) \notin T$  aggiungo  $(u, v) \in T \Rightarrow T' = T \cup \{(u, v)\}$

quando ho  
più archi leggeri

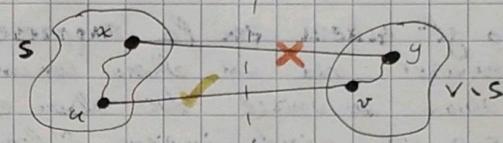
quindi costruisco  $T'' = T \cup \{(u, v)\} \setminus \{(x, y)\}$

altra andata  $T''$  è un albero ovviamente

$T''$  è un ST (dato che  $T$  era un mst).

un mst  $w(T'') = w(T)$

- trova un arco che  
passa per  $(u, v)$



$$\rightsquigarrow w(T'') = w(T) + w(u, v) - w(x, y)$$

$$w(T'') \leq w(T) \rightsquigarrow w(T'') = w(T) + w(u, v) - w(x, y) < w(T)$$

$$w(T'') \geq w(T) \rightsquigarrow \text{vera perché } T \text{ è un mst e il suo peso sarà minore o uguale di qualsiasi ST}$$

$\rightsquigarrow A \cup \{(u, v)\} \subseteq T''$ ?  $(x, y) \in A$  perde il taglio rispetto a  $A$  e  $(u, v) \in T''$   
per costruzione dunque  $(u, v)$  è ~~pesante~~ è sicuro

\*  $w(u, v) = w(x, y)$  se l'arco leggero che attraversa  $(S, V \setminus S)$   $(u, v)$  è UNICO

$\rightsquigarrow$  SE ESISTE un arco leggero che attraversa il taglio  $(S, V \setminus S)$ , allora TUTTI gli altri contengono l'arco leggero  $(u, v)$ !

• COROLARIO  $G = (V, E, w)$  NON orientato e connesso e  $w: E \rightarrow \mathbb{R}$

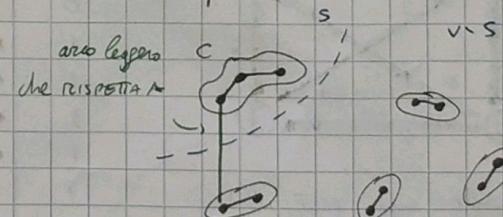
(i)  $A \subseteq E$  contenuto in qualche mst

(ii)  $C$  è una componente connessa di  $G_A = (V, A)$

(iii)  $(u, v)$  ARCO LEGGERO che collega  $C$  ad un'altra componente connessa di  $G_A$

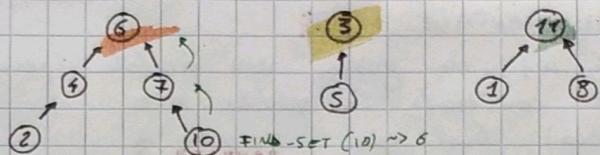
allora  $(u, v)$  è **sicuro** per  $A$

$\rightsquigarrow$  ricordo nel Teorema fondamentale



## STRUTTURE DATI PER INSERIMENTI DISGIUNTI \*

$$\{2, 4, 6, 7, 10\} \quad \{3, 5\} \quad \{1, 8, 11\}$$



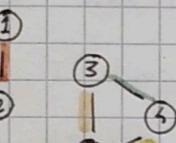
(i) MAKE-SET( $x$ )  $\rightsquigarrow \{x\}$

(ii) UNION( $x, y$ )  $\rightsquigarrow S_x \cup S_y$  i rappresentanti sono ex.g. 4

(iii) FIND-SET( $x$ )  $\rightsquigarrow$  cerca il rappresentante dell'insieme che contiene  $x$

Dato  $G$ , restituire le sue componenti connesse

• CONNECTED-COMPONENTS ( $G$ ) insieme dei vertici del grafo  $G$   
for each  $v \in V[G]$   $\rightsquigarrow$  MAKE-SET( $v$ ) /\* crea un insieme per ogni vertice \*/



for each  $(u, v) \in E[G]$   
if FIND-SET( $u$ )  $\neq$  FIND-SET( $v$ ) THEN  
UNION( $u, v$ )

step	insiemi
0	{1} {2} {3} {4} {5}
1	{1, 2} {3} {4} {5}
2	{1, 2} {3, 4} {5}
3	{1, 2} {3, 4, 5}
4	- nulla perché 3 e 5 stanno nello stesso insieme!

\* USO GL ALBERI PER LA RAPPRESENTAZIONE!

$\rightsquigarrow$  MAKESET crea un albero con un nodo  $x$

$\rightsquigarrow$  UNION ~~è~~ unisce due alberi e è BILANCIA!

$\rightsquigarrow$  FIND-SET quindi impiega  $O(\log n)$  [albero BILANCIA!]

• MST-VEROSIMIL ( $G, w$ )  $|V|=n$   $|E|=m$

1.  $A \leftarrow \emptyset$   
2. for each  $v \in V[G]$  }  $\Theta(n)$   
3. MAKE-SET( $v$ )

è un algoritmo GREEDY

4. ORDINA gli archi di  $E$  per PESO ~~non~~ decrescente  $\Theta(m \log m)$

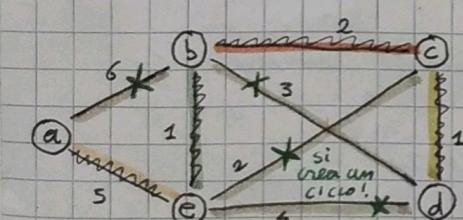
5. for each  $(u, v) \in E$   $\rightsquigarrow$  PRESO SECONDO L'ORDINE STABILITO AL PASSO 4 }  $\Theta(\log m)m$   
6. if FIND-SET( $u$ )  $\neq$  FIND-SET( $v$ ) then  
7. UNION( $u, v$ )  
8.  $A \cup \{(u, v)\}$

FIND-SET ↑  
ogni arc  
endo union

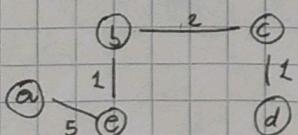
9. return  $A$

COMPLESSITÀ:  $\Theta(n + m \log m + m \log m) = \Theta(m \log m)$

CORRETTEZZA: vero per il Teorema fondamentale degli MST, sto estraendo un arco leggero (estraggo per peso ~~non~~ decrescente)



ho formato un MST !!



n-step	A	INSIEMI	ARCO ESTRASSO
0	{}	{a} {b} {c} {d} {e}	(c, d)
1	{(c, d)}	{a} {b} {c, d} {e}	(b, e)
2	{(c, d), (b, e)}	{a} {b, c} {d} {e}	(b, c)
3	{(c, d), (b, e), (b, c)}	{a} {b, c, d, e}	(e, c)
4	"	"	(b, d)
5	"	"	(e, d)
6	"	"	(a, e)
7	{(c, d), (b, e), (b, c), (a, e)}	{a, b, c, d, e}	end!
8	"	"	

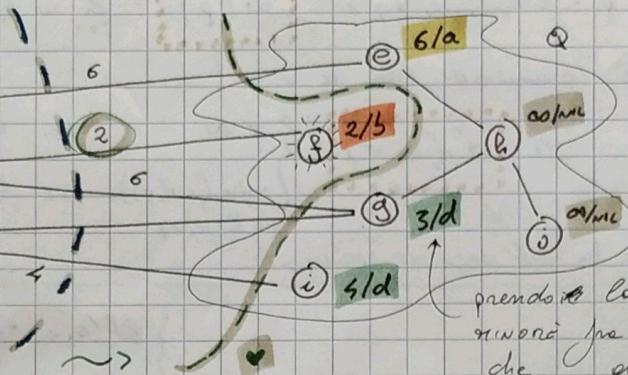
• PRIM :  $\pi_u$  = radice osi che verrà estratto [NON ha un predecessore]

$f_u \in V[G]$  { 1. key[u]  $\rightarrow$  chiave ha il peso minimo fra tutti gli archi che attraversano il TAGLIO e sono incidenti a  $\pi_u$  ]  
 2.  $\pi_u$  « PREDECESSORE » }

$Q \in V[G]$  : vertici da estrarre  
 gestite come  $V \setminus Q$  : insieme dei vertici già estratti  
 cosa d'priorità  
 [heap binario]  $V \setminus Q$   
 min-heap!

insieme degli archi che costituiscono il mio MST

$$A = \{(u, \pi[u]) \in E \mid u \in V \setminus \{n\} \setminus Q\}$$



cerca l'arco leggero che attraversa il taglio, quindi poi sposta il taglio aggiungendo gli altri vertici estratti

prendo la chiave di peso minore fra tutti gli archi che attraversano il taglio!

• PRIM-MST ( $G, w, n$ )

$$1. Q \leftarrow V[G]$$

2. for each  $u \in Q$  do  $\left\{ \begin{array}{l} \text{key}[u] = +\infty \\ \pi[u] = \text{NIL} \end{array} \right.$

3.  $\text{key}[n] \leftarrow 0$  /\* per costruzione, sarà il 1° vertice estratto \*/

4. while  $Q \neq \emptyset$  /\* finché ho vertici da estrarre \*/  $\leftarrow n$ -volte  $\boxed{O(n \log n)}$

5.  $u \leftarrow \text{extract } \min(Q)$   $\boxed{O(\log n)}$

6. for each  $v \in \text{Adj}[u]$   $\leftarrow 2m$

7. if  $v \in Q \wedge w(u,v) < \text{key}[v]$  then  $\boxed{O(m \log n)}$

8.  $\pi[v] \leftarrow u$

9.  $\text{key}[v] \leftarrow w(u,v)$   $\log n$

10.  $\pi[u] \leftarrow v$   $\log n$

11.  $\pi[v] \leftarrow u$   $\log n$

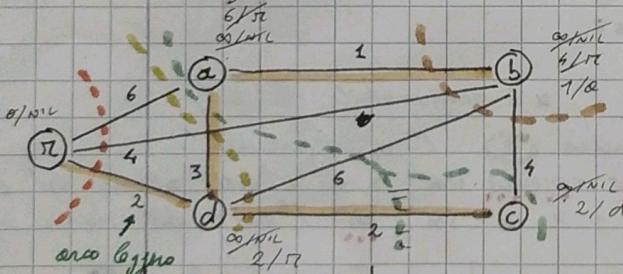
12. return  $A = \{(u, \pi[u]) \in E \mid u \in V \setminus \{n\}\}$

\*  $\deg(u_1) + \dots + \deg(u_n) = \sum_{i=1}^n \deg(u_i) = 2m$   
 per il lemma della stratta d'arco

COMPLESSITÀ:  $T_{\text{prim}}(n,m) = O(n + n \log n + m \log n) = \boxed{O(m \log n)}$

$\hookrightarrow G$  è connesso  $\Rightarrow m \geq n-1$  quindi  $m \leq n-1$

CORRETTEZZA: ora per il terzino fondamentale degli osi (enunciato e dimostrarlo)



è un algoritmo greedy

tra i vertici da estrarre sceglie quello con chiave minore!

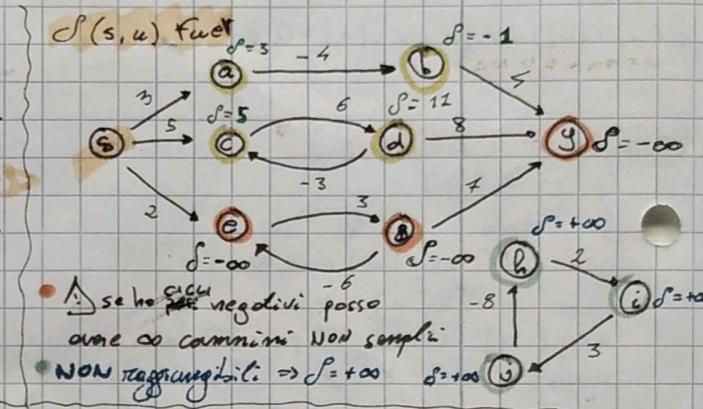
n-step	$V \setminus Q$	$\text{key}   \pi$					
0	{}	0 NIL	00 NIL	00 NIL	00 NIL	00 NIL	00 NIL
1	{n}	0 NIL	6 n	6 n	6 n	6 n	6 n
2	{n, d}	0 NIL	3 d	3 d	3 d	3 d	3 d
3	{n, d, c}	0 NIL	3 d	3 d	3 d	3 d	3 d
4	{n, d, c, a}	0 NIL	3 d	3 d	1 a	1 a	1 a
5	{n, d, c, a, b}	0 NIL	3 d	3 d	1 a	2 d	2 d

( $\rightarrow$  guardo i predecessori per ricostruire l'albero!)

- CANNINI MINIMI
- CANNINO  $u, v \in V$
- $P = \langle x_0, x_1, \dots, x_q \rangle$  dove  $x_0 = u$ ,  $x_q = v$   $\rightsquigarrow$   $P$  è minimo se  $w(P) = \delta(u, v)$
- e  $i = 1 \dots q$  :  $(x_{i-1}, x_i) \in E$
- PESO DEL CANNINO :  $P = \langle x_0, \dots, x_q \rangle$  :  $w(P) = \sum_{i=1}^q w(x_{i-1}, x_i)$
- $C(u, v) = \{ P \mid P \text{ è cammino tra } u \text{ e } v \}$
- «DISTANZA» tra  $u$  e  $v$   $\rightsquigarrow \delta(u, v) = \begin{cases} 0 & \text{se } C(u, v) \neq \emptyset (\exists \text{ cammino tra } u \text{ e } v) \\ \min(w(P)) & \text{se } C(u, v) \neq \emptyset (\exists \text{ almeno un cammino}) \\ -\infty & \text{se ho solo negativi raggiungibili} \end{cases}$

### • VARIANTI PROBLEMA

destinazione sortiente	SINGOLA	MULTIPLA
SINGOLA	In: $G = (V, E, w)$ $u, v \in V$ Out: $\delta(u, v)$	In: $G = (V, E, w)$ $s \in V$ sortiente Out: $\forall u \in V : \delta(s, u)$
MULTIPLA	In: $G = (V, E, w)$ $d \in V$ destinazione Out: $\forall u \in V : \delta(u, d)$	In: $G = (V, E, w)$ $d \in V$ destinazione Out: $\forall u, v \in V : \delta(u, v)$ si può ricordare a scorrere SX e trovare (invertendo il verso degli archi)



### • STRUTTURA dati

- (ii)  $d[u] =$  "stima" di cammino minimo  $f_u \in V$ , alla fine  $d[u] = \delta(s, u)$   $s \in V$
- (iii)  $\pi[u] =$  predecessore

### • GRATO DEI PREDECESSORI

$$G_\pi = (V_\pi, E_\pi) \rightsquigarrow \left\{ \begin{array}{l} V_\pi = \{u \in V \mid \pi[u] \neq \text{NIL}\} \cup \{s\} \\ E_\pi = \{(u, \pi[u]) \in E \mid u \in V_\pi \setminus \{s\}\} \end{array} \right.$$

dipende dall'esecuzione dell'algoritmo

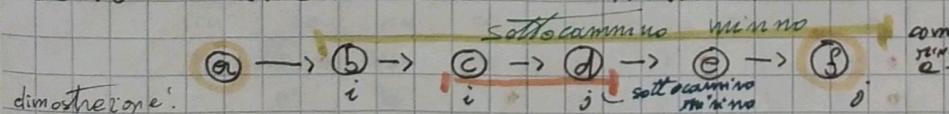
### • ALBERO DEI CANNINI MINIMI $G = (V, E)$ $w: E \rightarrow \mathbb{R}$ $s \in V$

$$G'(V', E') = \left\{ \begin{array}{l} (i) V' = \{u \in V \mid \delta(s, u) < +\infty\} /* u è raggiungibile da s */ \\ (ii) G' forma un ALBERO con radice s /* albero radicato */ \\ (iii) \forall u \in V' \text{ c'è UNICO CAMMINO SEMPRE tra } s \text{ e } u \text{ nell'albero } G' \text{ è } \text{non è detto che sia unico} \end{array} \right.$$

### • PROPRIETÀ IMPORTANTI

#### • SOTTOCAMMINI di cammini minimi

• Sono anch'essi minimi



dimostrazione:

$u, v \in V$   $P = \langle x_0, x_1, \dots, x_q \rangle$  cammino minimo  $x_0 = u$ ,  $x_q = v$

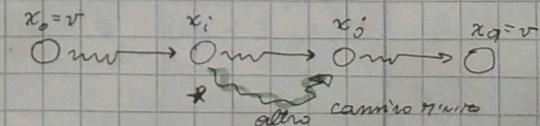
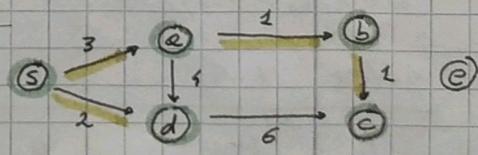
SOTTOCAMMINO  $\langle x_i, x_{i+1}, \dots, x_j \rangle = P_{ij}$

$x_i$  dove  $0 \leq i \leq j \leq q$

$\rightsquigarrow$  più è minimo

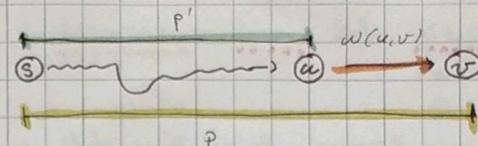
$x_j$

\* se per assurdo suppongo che il sottocammino  $\langle x_i, \dots, x_j \rangle$  non sia minimo allora dovrebbe esisterne uno più piccolo anche tra  $u$  e  $v$  (passando per il cammino minimo tra  $i$  e  $j$ )!



$$G = (V, E, w) \quad w: E \rightarrow \mathbb{R}$$

P cammino minimo tra  $s$  e  $v$   
P' è un sottocammino di  $P$ , quindi è minimo



$$\delta(s, v) = \delta(s, u) + w(u, v)$$

sottocammino peso sull'arco

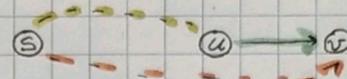
dimostrazione:  $\delta(s, v) = w(p) = w(p') + w(u, v) = \delta(s, u) + w(u, v)$

### DISEGUALANZA TRIANGOLARE

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

$$w: E \rightarrow \mathbb{R}$$

$$\delta(s, v) \leq \delta(s, u) + w(u, v)$$



dimostrazione

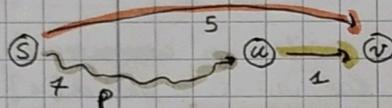
(a)  $\delta(s, u) = +\infty$  :  $u$  non è raggiungibile da  $s$

S non ha arco  $\Rightarrow \delta(s, v) \leq +\infty = \delta(s, u) + w(u, v)$   
 potrebbe essere!

(b)  $\delta(s, u) = -\infty$  : esiste un ciclo negativo tra  $s$  e  $u$  raggiungibile dalla sorgente

S ciclo negativo!  $\Rightarrow \delta(s, v) = -\infty = \delta(s, u) + w(u, v)$

(c)  $\delta(s, u) \in \mathbb{R}$  : non esiste un ciclo negativo



$p$  è un cammino minimo tra  $s$  e  $u$ , non è detto che raggiungendo  $w(u, v)$  esso sia ancora minimo!

\* per definizione è il peso del cammino più piccolo tra  $s$  e  $v$   
 quindi è  $\leq$  del peso di qualsiasi cammino tra  $s$  e  $v$

$\Rightarrow \delta(s, v) \leq \delta(s, u) + w(u, v) \leq w(p) + w(u, v) = \delta(s, u) + w(u, v)$   
 dato che  $p$  è minimo!

### CAMMINI MINIMI SORGENTI SINGOLI

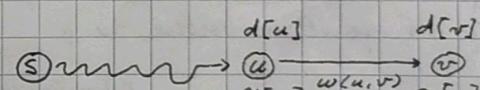
#### STRUTTURE DATI

posso manipolare i campi solo con init-ss e relax!

#### INIT - SINGLE - SOURCE ( $G, s$ )

1. for each  $u \in V[G]$
2.  $d[u] = +\infty$
3.  $\pi[u] = \text{NIL}$
4.  $d[s] = 0$

$$\left\{ \begin{array}{l} d[u] : \text{stima di } \delta(s, u) \\ \pi[u] : \text{predecessore} \end{array} \right.$$



- RELAX ( $u, v, w(u, v)$ )
  1. if  $d[v] > d[u] + w(u, v)$  then
  2.  $d[v] = d[u] + w(u, v)$
  3.  $\pi[v] = u$



UNA PROPRIETÀ : dopo aver eseguito una relax( $u, v, w(u, v)$ ) si ha sempre  $d[v] \leq d[u] + w(u, v)$

dim: se entra nell'if allora  $d[v] = d[u] + w(u, v)$ , altrimenti  $d[v] < d[u] + w(u, v)$

PROPRIETÀ [sotto] GRADO dei predecessori : alla fine di un algoritmo che usa la relax INIT-SS si ha  $d[u] = \delta(s, u)$   $\forall u \in V$  stime tutte corrette!

$\pi$  è un ALBERO di cammini minimi

### • PROPRIETÀ del LIMITE INFERIORE

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

$$W : E \rightarrow \mathbb{R}$$

se si inizializzano le strutture dati con una INIT-SINGLE-SOURCE per qualunque sequenza di RELAX si ha sempre

$$\delta(s, u) \leq d[u] \quad \forall u \in V$$

inoltre, se dopo la RELAX si ha  $\delta(s, u) = d[u]$  allora il valore  $d[u]$  NON può più cambiare!

dimostrazione: - - - - -  $\left\{ \begin{array}{l} u \neq s \quad d[u] = +\infty \quad \delta(s, s) = -\infty \\ u = s \quad d[s] = 0 \quad \delta(s, s) = 0 \end{array} \right.$   $\text{on } \delta[s]$

- la proprietà è vera subito dopo la INIT-SS  $\left\{ \begin{array}{l} u = s \quad d[s] = 0 \quad \delta(s, s) = 0 \\ \text{per ASSURDO assumo che } \exists v \in V \text{ t.c. dopo una RELAX } d[v] < \delta(s, v) \\ \text{quindi } v \text{ è il } \text{1}^{\text{o}} \text{ vertice per cui la proprietà è violata! ma allora deve} \\ \text{esserci stata una RELAX}(u, v, w(u, v)) \text{ che ha violato le cose} \end{array} \right.$   $\downarrow d[v]$

$$d[u] + w(u, v) = d[v] \quad \left[ \begin{array}{l} \text{per ipotesi} \\ \text{diseguaglianza triangolare} \end{array} \right] \quad \delta(s, u) + w(u, v) \Rightarrow d[u] < \delta(s, v)$$

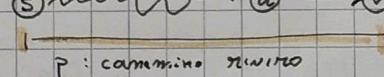
ma allora deve essere vero primo (per u)  $\leftarrow$  RELAX non può cambiare  
della RELAX ma è impossibile  $\leftarrow$  il valore di u

- nel momento in cui  $d[u] = \delta(s, v)$  si ha che NESSUNA RELAX potrà cambiare il valore di  $d[v]$  perché  $\delta(s, v)$  è il LIMITE INFERIORE, inoltre NON può nemmeno aumentare perché o altrimenti o non fa nulla!

• PROPRIETÀ dell'ASSERZIONE di CARRINO: se tra s e v NON esiste un cammino ( $\Rightarrow v$  non è raggiungibile)  $\delta(s, v) = +\infty$  dopo la INIT-SS si ha  $d[u] = \delta(s, v)$

• PROPRIETÀ della CONVERGENZA: lancio la INIT-SS e dopo una sequenza di RELAX ho  $d[u] = \delta(s, u)$  (per ipotesi)

$s \xrightarrow{\text{RELAX}} u \xrightarrow{\text{RELAX}} v$  allora dopo la 1<sup>a</sup> RELAX ( $u, v, w(u, v)$ ) si ha



$$d[v] = \delta(s, v)$$

dimostrazione

[dopo la RELAX]

[per ipotesi]

$$\delta(s, v) \leq d[v] \leq d[u] + w(u, v) = \delta(s, u) + w(u, v) = \delta(s, v)$$

[dato che p è un cammino scorso]

[proprietà del  
LIMITE INFERIORE!]

$$\Rightarrow d[v] = \delta(s, v)$$

• DIJKSTRA ( $G, w, s$ )

1. INIT-SINGLE-SOURCE ( $G, s$ ) v l  
 2.  $Q \leftarrow V[G]$   
 3.  $/* S \leftarrow \emptyset */$  vertici già estratti  
 4. WHILE  $Q \neq \emptyset$  do  
 5.      $u = \text{EXTRACT-MIN}(Q)$   
 6.      $/* S \leftarrow S \cup \{u\} */$   
 7.     for each  $v \in \text{Adj}[u]$   
 8.         RELAX ( $u, v$ ,  $d_v$ )  
 9. return  $\langle d, G_{\pi} \rangle$

• COMPLESSITÀ: dipende dalla densità del grafo

(ii) Q ARRAY LINEAR  $\Rightarrow O(n+n^2+m) = O(n^2)$

(iii) Q HEAP BINARIO  $\Rightarrow O(n + n \log n + m \log n) = O(m \log n)$  greedy

► Q: code di punte, VERTICI DA ESTRARRE!

► d : campo chiave di Q

•  $S \cup Q = V$  ( $S = V \setminus Q$ ) VERTICI GIA' ESTRASSI!

Adj[u] DIPENDE dall' OUT-DEGREE() di OGNI  
 ARCO!  $m = \sum_{u \in V} \text{out-deg}(u)$

MICCO: VH -  $\Delta$   $\omega_{EV}$  801-009 (a)

GRAFO	ARRAY	HEAP
NORMALE $\gg$	$O(n^2)$	$O(m \log n)$
SPARSO $m \approx n$	$O(n^2)$	$O(n \log n)$
DENSO $m \approx n^2$	$O(n^2)$	$O(n^2 \log n)$

## TEOREMA DISJUNTURA (correttezza!)

sia  $G = (V, E)$  con  $W: E \rightarrow \mathbb{R}$  e  $s \in V$ : sorgente. assumo che i pesi siano tutti positivi.  
 quindi  $w(u, v) \geq 0 \quad \forall (u, v) \in E$   
 allora, al termine [in realtà si verifica al momento dell'espirazione] si avrà:

- (a)  $d[u] = \delta(s, u)$   $\forall u \in V$  /\* tutte le streghe sono connette \*/  
 (b) G<sub>H</sub> è un ALBERO di cammini minimi. /\* è vero se è vero la a \*/

### dimostrazione (a)

OSSERVAZIONE: al momento dell'estrazione di  $u$  da  $Q$  si ha  $d[u] = d(s, u)$

per assurso assumo che  $\exists u \in V : d[u] \neq d(s, u)$  al momento dell'estrazione e assumo che sia il 1° vertice per cui questo accade!

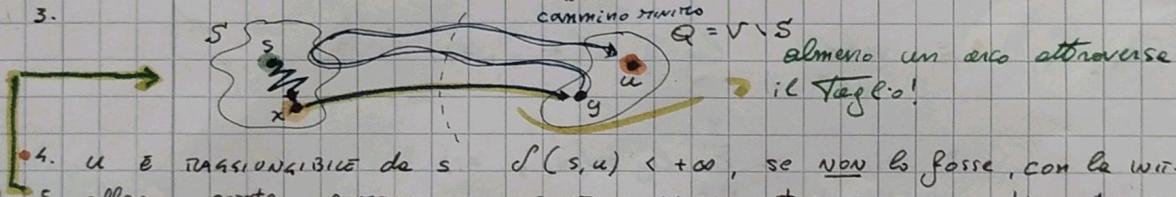
ALLOTA

- 1.  $u \neq s$  perché  $d[s] = 0 \neq d(s, s)$  nella INIT-SINGLE-SOURCE!

[PER DEFINIZIONE, NO CICL NEGATIVI]

2. all'estrazione di un posso affermare che sia (SES)

3



- almeno un arco attraverso il Taglio!

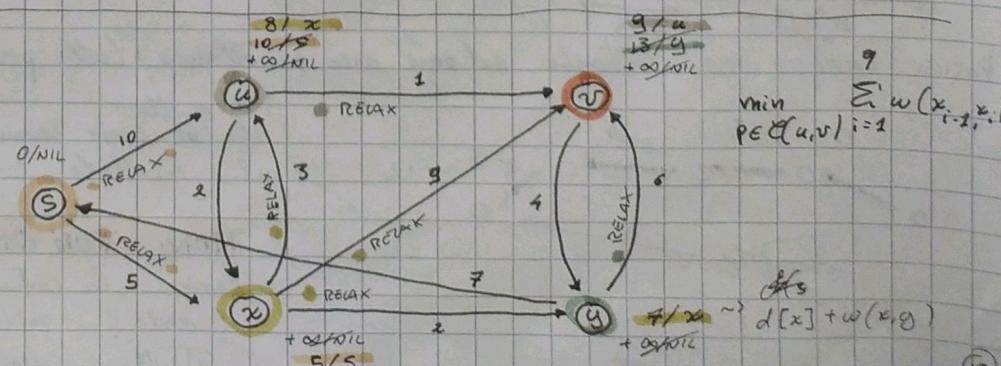
  - 4.  $u$  è raggiungibile da  $s$   $d(s, u) < +\infty$ , se non lo fosse, con la Wt-SS,  $d[u] = +\infty = f(s, u)$
  - 5. allora esiste altrove un cammino minimo tra  $s$  e  $u$ , chiamato  $p$
  - 6.  $d[x] = f(s, x)$  per ipotesi ( $u$  è il primo vertice che viola le condizioni)
  - 7.  $d[y] = f(s, y)$  proprietà della convergenza (c'è solo un arco che collega  $x$  e  $y$ )  
di queste ha fatto una relazione su tutti gli archi uscenti
  - 8.  $d[u] \leq d[y]$  sta per essere estratto  $u$ , quindi di Dijkstra estrae il vertice con campo di più piccolo quindi per farlo sole  $f(s, u)$  è il peso di tutto il cammino minimo, mentre  $f(s, y)$  è un sotto cammino. vale solo perché i pesi sono tutti  $> 0$
  - 9.  $f(s, y) \leq f(s, u)$  proprietà del limite inferiore
  - 10.  $f(s, u) \leq d[u]$

QUINDI  $\mathcal{S}(s, u) \leq d[u] \leq d[y] = \mathcal{S}(s, y) \leq \mathcal{S}(s, u) \Rightarrow d[u] = \mathcal{S}(s, u)$  assunto che  $\mathcal{S} \neq$

esempio!

• ORDINE DI ESTRAZIONE

S  
x  
g  
u



esercizio e caso

$$\bullet G = (V, E)$$

$\bullet f(u, v) \in \mathbb{R} : \pi[u, v] \in [0, 1]$  "affidabilità" canale di comunicazione tra  $u$  e  $v$

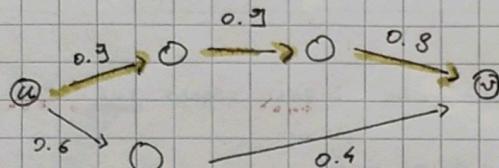
$\downarrow$   
 $\pi(u, v) = \text{probabilità che il canale tra } u \text{ e } v \text{ trasmetta correttamente un messaggio [indipendenti]}$

costruire un algoritmo che trovi un cammino tra  $u$  e  $v$  con la massima AFFIDABILITÀ

$$C(u, v) = \{p \mid p \text{ è cammino tra } u \text{ e } v\}$$

$$p = \langle x_0, x_1, \dots, x_q \rangle \text{ con } x_0 = u \text{ e } x_q = v$$

$$\alpha(p) = \prod_{i=0}^{q-1} \pi(x_{i-1}, x_i) \quad \text{AFFIDABILITÀ SISTEMA}$$



$$\max_{p \in C(u, v)} \alpha(p) = \prod_{i=1}^{q-1} \pi(x_{i-1}, x_i)$$

vorrei trasformarlo in un problema di CAMMINI MINIMI

$$\min_{p \in C(u, v)} \sum_{i=1}^{q-1} \pi(x_{i-1}, x_i)$$

COSÌ

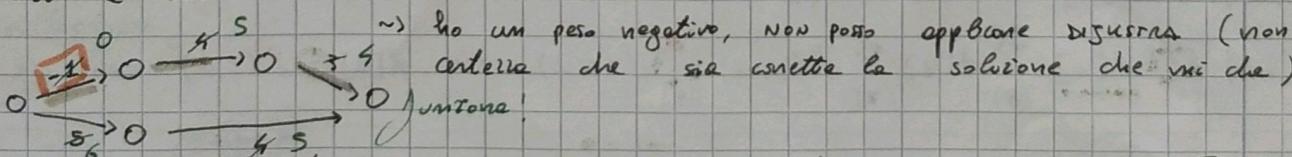
- $\max f(x) \equiv \max G(f(x))$  con  $G$  fz. crescente, solo se i gli sono  $> 0$  !!!
- SPESO si USA LA FZ. LOGARITMO!
- $\max f(x) \equiv \max \log(f(x))$  min > log trasforma: prodotti in somme
- $\max f(x) \equiv \min G(f(x))$  con  $G$  decrescente
- SPESO SI UTILIZZA  $G(x) = -x$
- $\max f(x) \equiv \min -f(x)$

TRASFORMARE MAX IN MIN CON MAX PRODOTTI E MIN SOMME

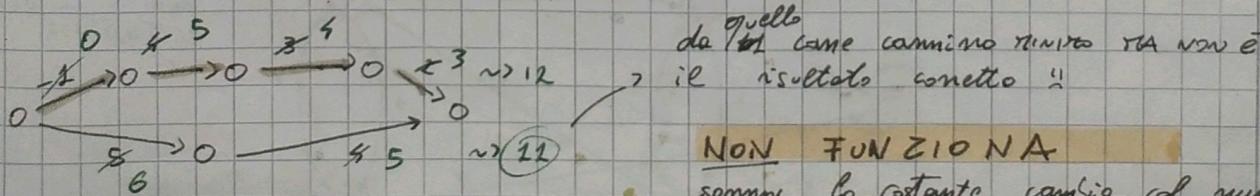
$$\begin{aligned} \max_{p \in C(u, v)} \prod_{i=1}^{q-1} \pi(x_{i-1}, x_i) &\equiv \max_{p \in C(u, v)} \log \prod_{i=1}^{q-1} \pi(x_{i-1}, x_i) \\ &\equiv \min_{p \in C(u, v)} - \sum_{i=1}^{q-1} \log \pi(x_{i-1}, x_i) \\ &\equiv \min_{p \in C(u, v)} \sum_{i=1}^{q-1} \log \frac{1}{\pi(x_{i-1}, x_i)} \end{aligned}$$

quindi  $f'(u, v)$  è sostituito  $\pi(u, v) \rightarrow \log\left(\frac{1}{\pi(u, v)}\right)$  tutti i pesi sono positivi, uso dijstra e ho la soluzione equivalente al problema di percorso!

esercizio e caso



POTREI SOTTRAERE UNA COSTANTE AD OGNI ARCO PER OTTENERE TUTTI PESI POSITIVI, MA (NON FUNZIONA SEMPRE)



NON FUNZIONA

somma la costante cambia col numero di archi! potenzialmente cambia la struttura dei cammini

## BELLMAN-FORD (G, w, s)

```

1. INIT-SOURCE (G, S) ~> n^(n-1)
2. for i=1 to |V[G]| - 1 ~> m volte
   for each (u,v) G E[G] ~> relax u
3.   RELAX (u, v, w(u,v)) ] m volte O(1)
4. /* verifica se nel grafo ci sono cicli NEGATIVI */
5. for each (u,v) E[G] ~> m volte
6.   if d[v] > d[u] + w(u,v) then
7.     RETURN < FALSE, d, π>
8. RETURN < TRUE, d, π>

```

funzione ANCHE con pesi NEGATIVI (obv. no cicli)

► RIASSA TUTTI gli ARCHI  $n-1$  volte BROTE FORCE!

$$T(n, m) = \Theta(n + (n-1)m + m) = \Theta(n \cdot m)$$

	GRAFO SPARSO ( $m \approx n$ )	GRAFO DENSO ( $m \approx n^2$ )
DISTANZA (anag)	$n^2$	$n^2$
DISTANZA (cheap)	$n \log n$	$n^2 \log n$
BELLMAN-FORD	$n^2$	$n^3$

confronto  
fra DISTANZA  
e BELLMAN-FORD

## BELLMAN-FORD TEOREMA (CORRETTOzza)

Sia  $G = (V, E)$  con  $W: E \rightarrow \mathbb{R}$  con vertice sorgente  $s \in V$  e si esegua BF

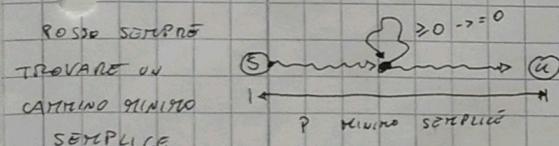
- (i) se  $G$  NON contiene cicli NEGATIVI raggiungibili da  $s$ , alla fine di BF:
- (a)  $d[u] = f(s, u)$   $\forall u \in V$
  - (b)  $G_\pi$  è un ALBERO DI CAMMINI MINIMI
  - (c) l'algoritmo restituisce TRUE

(ii) se in  $G$  ESISTE un CICLO NEGATIVO raggiungibile da  $s$ , allora BF restituisce FALSE

dimostrazione!

(a)  $\forall u \in V \Rightarrow d[u] = f(s, u) \Leftrightarrow f(s, u) = \begin{cases} +\infty & \text{verrà avvi (vero dopo la INIT-SS)} \\ \in \mathbb{R} & * \\ -\infty & \text{impossibile perché per ipotesi NON ci sono cicli negativi raggiungibili da } s \end{cases}$

\*  $f(s, u) \in \mathbb{R}$ ,  $u$  è raggiungibile da  $s \Rightarrow \exists$  ACCEDO un cammino tra  $s$  e  $u$  e quindi un cammino MINIMO SEMPLICE (NO CICLI NEGATIVI)



• Sia  $p$  un cammino minimo semplice tra  $s$  e  $u$   
•  $p = \langle x_0, x_1, \dots, x_q \rangle$   $x_0 = s$ ,  $x_q = u$   
 $x_0 = s \rightarrow x_1 \rightarrow x_2 \rightarrow \dots \rightarrow x_q$   
 $d[x_0] = 0 = f(s, s)$   $d[x_1] = f(s, x_1)$   $d[x_2] = f(s, x_2)$   $d[x_q] = f(s, x_{q-1})$   
INIT-SS RELAX proprietà convergenza!  
BF rilassa tutti gli archi

•  $p$  ha AL PIÙ  $n-1$  archi in quanto è CAMMINO SEMPLICE!

(c) alla fine restituisce TRUE  $f(u, v) \leq d[v] \leq d[u] + w(u, v)$  ma  $f(s, v) \leq f(s, u) + w(u, v)$  [P.TRIANGOLARE]

(ii) ipotesi:  $\exists$  ciclo NEGATIVO raggiungibile da  $s$ ; TOSI: l'algoritmo restituisce FALSE

assumo per ASSURDO che BF restituisca TRUE  $\Rightarrow f(u, v) \leq d[v] \leq d[u] + w(u, v)$

$\exists c = \langle x_0, \dots, x_q \rangle$  raggiungibile da  $s \Rightarrow \nexists x_0 = x_q$  ciclo!

$$\sum_{i=1}^q w(x_{i-1}, x_i) \leq 0 *$$

$$\forall i=1 \dots q : d[x_i] \leq d[x_{i-1}] + w(x_{i-1}, x_i)$$

$$\sum_{i=1}^q d[x_i] \leq \sum_{i=1}^q d[x_{i-1}] + \sum_{i=1}^q w(x_{i-1}, x_i)$$

$$\begin{aligned} & d[x_1] + d[x_2] + \dots + d[x_q] \\ & d[x_0] + d[x_1] + d[x_2] + \dots + d[x_{q-1}] \\ & \downarrow \\ & d[x_0] = d[x_q] \text{ ciclo!} \end{aligned}$$

ASSURDO perché ho assunto che era  $< 0 *$

Esercizio random!

$G = (V, E)$   $w: E \rightarrow \mathbb{R}$  g.o. assumo  $w(u, v) > 0$   $\forall s \in V$  sorgente

$f(u, v)$   $\exists$  ciclo in  $G$  raggiungibile da un vertice  $s \in V$  t.c.  $\prod_{i=1}^q w(x_{i-1}, x_i) \leq 1$

posso trasformare la condizione in  $\log \prod_{i=1}^q w(x_{i-1}, x_i) < \log 1$

inoltre  $\log$  di probabilità posso scrivere come somma  
e il  $\log 1$  è sempre zero

quindi posso usare BELLMAN-FORD trovando  $c = (x_0, x_1, \dots, x_q)$   
ovviamente trasformando i pesi

Esercizio ARBITRAZIO

determinare  $c = (x_0, x_1, \dots, x_q)$   $x_0 = x_q$  t.c.  $\prod_{i=1}^q w(x_{i-1}, x_i) \geq 1$

$$\log \prod_{i=1}^q w(x_{i-1}, x_i) \geq \log 1$$

$$\sum_{i=1}^q \log w(x_{i-1}, x_i) \geq 0$$

$$\sum_{i=1}^q -\log w(x_{i-1}, x_i) \leq 0$$

$$\sum_{i=1}^q \left( \frac{1}{w(x_{i-1}, x_i)} \right) \leq 0$$

ho ricondotto a un problema d. cicli negativi, non ho un vertice sorgente  
quind' applico BELLMAN-FORD ripetutamente su ogni vertice

**⚠ CAMMINI MINIMI ERA TUTTE LE COPPIE ⚠**

molto poco efficiente

ITERATED-DIGESTRA ( $G, w$ ) /* o BF */	IT-D (1)	$n^3$
for each $s \in V[G]$ do	IT-D (2)	$n^2 \log n$
DIGESTRA ( $G, w, s$ ) /* o BF */	IT-BF	$n^3$

grado SPARSO minimo grado DENSE max

### FLOYD-WARSHALL

$$G = (V, E) \quad w : E \rightarrow \mathbb{R}$$

$$V = \{1, 2, \dots, n\} \quad \text{vertici numerati}$$

OK ARCHI NEGATIVI

NO CICLI NEGATIVI

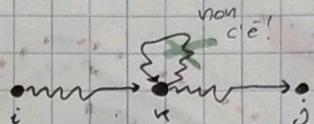
$$\boxed{W} \Rightarrow \boxed{\text{FLOYD-WARSHALL}} \Rightarrow \begin{cases} D & : \text{matrice delle DISTANZE} \\ \pi & \end{cases}$$

$$W = (w_{ij}) \in \mathbb{R}^{n \times n} = w_{ij} = \begin{cases} 0 & \text{se } i = j \\ w(i,j) & \text{se } i \neq j \wedge (i,j) \in E \\ +\infty & \text{se } i \neq j \wedge (i,j) \notin E \end{cases}$$

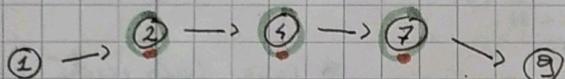
$\boxed{W}$  matrice che rappresenta il grafo

- dati  $i, j \in V$   $D_{ij} = \{p \mid p \text{ è un CAMMINO SEMPLICE tra } i \text{ e } j\}$
- $v \in V = \{1, \dots, n\}$

voglio risolvere il problema  $\delta(i, j) = \min_{p \in D_{ij}} w(p)$

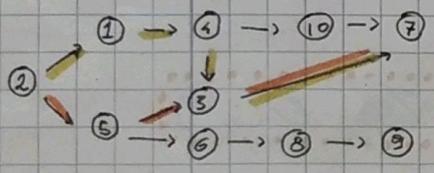


### vertice intermedio



- $i, j \in V \rightarrow D_{ij} = \{p \mid p \in D_{ij}, i \text{ vertici INTERMEDI sono MINORI o uguali a } v\}$
- $v \in V$

ma allora  $d_{ij}^{(k)} = \min_{p \in D_{ij}^{(k)}} w(p) \Rightarrow \delta(i, j) = d_{ij}^{(n)}$



$$\begin{aligned} D_{2,7}^{(1)} &= \emptyset \\ D_{2,7}^{(2)} &= D_{2,7}^{(3)} = \emptyset \\ D_{2,7}^{(4)} &= \{<2, 1, 4, 3, 7>\} \\ D_{2,7}^{(5)} &= \{<2, 1, 4, 3, 7>, <2, 5, 3, 7>\} \end{aligned}$$

tutti i vertici intermedi sono  $\leq k$  ( $k=4$ )

ovviamente  $D_{ij} \subseteq D_{ij}^{(k)}$

$$D_{ij} = D_{ij}^{(n)}$$

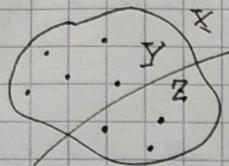
non è il numero di vertici intermedi  
MA è il contenuto dell'etichetta dei vertici intermedi che deve essere  $\leq k$  !!!

$$\bullet d_{ij}^{(k)} = \min_{p \in D_{ij}^{(k)}} w(p) \Rightarrow \boxed{d_{ij}^{(n)} = \min_{p \in D_{ij}^{(n)}} w(p)}$$

$$\left\{ \begin{array}{l} \text{matrice } D^{(n)} = (d_{ij}^{(n)}) \quad i=0 \dots n \\ D^{(0)} = W \\ D^{(1)} \subseteq D^{(2)} \subseteq \dots \subseteq D^{(n)} \end{array} \right.$$

I tre lutti i cammini che partono da  $i$  e arrivano a  $j$  i cui vertici intermedi sono tutti minori o uguali a  $k$ , qual è il cammino che ha la lunghezza minima?

• dato un generico  $D^{(n)}$  come faccio a costruire  $D^{(n+1)}$ ?



voglio trovare  $\min_X$   
partizione  $X$  in due parti  $Y$  e  $Z$   
allora  $\min_X = \min \{\min_Y, \min_Z\}$

principio di programmazione dinamica

$k-1$  perché  
i cammini sono  
semplici e non ha  
vertici  
ripetuti



•  $\hat{D}_{ij}^{(k)}$ : cammini che PASSANO per  $k$

•  $\hat{D}_{ij}^{(k-1)}$ : cammini che NON passano per  $k$

$$D_{ij}^{(n+1)} = \hat{D}_{ij}^{(n)} \cup \hat{D}_{ij}^{(n-1)}$$

$$\hat{D}_{ij}^{(n)} = D_{ik}^{(n-1)} \cup D_{kj}^{(n-1)}$$

$$d_{ij}^{(n+1)} = \min_{p \in D_{ij}^{(n+1)}} w(p) = \min \left\{ \min_{p \in \hat{D}_{ij}^{(n)}} w(p), \min_{p \in \hat{D}_{ij}^{(n-1)}} w(p) \right\} =$$

$$= \min \left\{ \min_{p \in D_{ik}^{(n-1)}} w(p) + \min_{p \in D_{kj}^{(n-1)}} w(p), \min_{p \in D_{ij}^{(n-1)}} w(p) \right\} =$$

$$= \min \left\{ \min_{p \in D_{ik}^{(n-1)}} d_{ik}^{(n-1)} + d_{kj}^{(n-1)}, d_{ij}^{(n-1)} \right\}$$

\* posso farlo perché sono cammini semplici  
quindi i sottocammini appartengono all'insieme  
con indice  $n-1$

$$\text{ORA ALLORA } d_{ij}^{(n+1)} = \begin{cases} w_{ij} & \text{se } n=0 \\ \min \{ d_{ik}^{(n-1)} + d_{kj}^{(n-1)}, d_{ij}^{(n-1)} \} & \text{altrimenti} \end{cases}$$

### FLOYD-WARSHALL (W)

```

1. v = rows(W)
2. D(0) = W
3. for k=1 to n /* calcola D(1) D(2) ... D(n) */
4.   for i=1 to n
5.     for j=1 to n
6.       dij(n) = min { dik(n-1) + dkj(n-1), dij(n-1) }
7. return D(n)

```

$$T(n) = \Theta(n^3)$$

• Sovrascrive la matrice togliendo gli indici superiori ed è corretto per la proprietà (ii)  $\rightarrow$

quindi  $S(n) = \Theta(n^2)$

(ii) se  $\exists$  cicli negativi  $\Rightarrow f_{u=0 \dots n} : d_{ii}^{(k)} = 0$  [la diagonale principale è nulla]

dimostrazione

- $v=0$  vero per costruzione  $D^{(0)} = w$
- $v > 1$   $d_{ii}^{(k+1)} = 0$

assumo che  $d_{ii}^{(k+1)} = 0$  e dimostro che vale anche per  $d_{ii}^{(k)}$

$$d_{ii}^{(k)} = \min \{ d_{ik}^{(k-1)} + d_{ki}^{(k-1)}, d_{ii}^{(k-1)} \}$$

[per definizione]

è zero per ipotesi induttiva

deve essere  $\geq 0$  perché la somma è il peso di un ciclo i cui vertici intermedii sono tutti  $< v-1$  e se avessi un valore  $< 0$  significherebbe che ho un ciclo negativo ma so che non esistono!

$\rightarrow$  allora il minimo è zero ( $d_{ii}^{(k-1)} = 0$ )

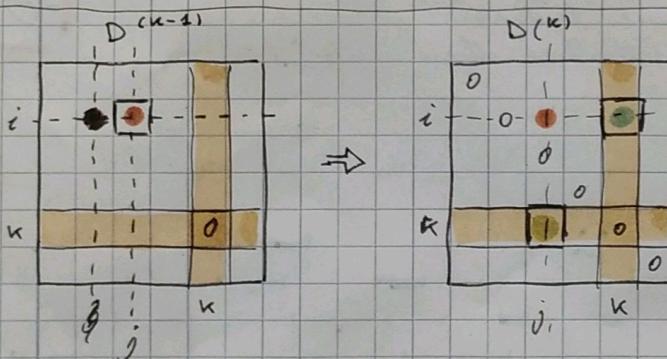
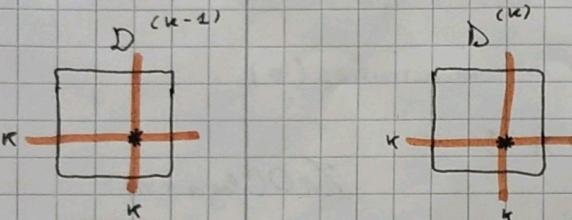
se voglio che FLOYD mi avisi di cicli negativi guardo le diagonali, se ho un valore  $< 0$  altrimenti ci sono cicli negativi senza dubbio

(iii)  $i, j \in V$  e  $v \in V \Rightarrow \begin{cases} d_{iv}^{(k)} = d_{iv}^{(k-1)} \\ d_{vj}^{(k)} = d_{vj}^{(k-1)} \end{cases}$  riga e colonna al passo  $v$  sono uguali al passo  $v-1$

è zero

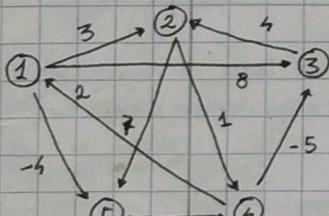
DIAGONALE

dimostrazione  $d_{iv}^{(k)} = \min \{ d_{iv}^{(k-1)}, d_{iv}^{(k-1)} + d_{vv}^{(k-1)} \} = d_{iv}^{(k-1)}$  PRINCIPATO



per calcolare l'elemento in posizione  $i, j$  della matrice  $v$   
lo trovo come il più piccolo elemento  
fra l'elemento in posizione  $i, j$   
e la somma tra i valori in  
 $i, v$  e  $v, j$

esempio



	1	2	3	4	5	6
1	0	3	8	00	-4	
2	00	0	00	1	7	
3	00	4	0	00	00	
4	2	00	-5	0	00	
5	00	00	00	6	0	

$w = D^0 = 3$

	1	2	3	4	5	6
1	0	3	8	00	-4	
2	00	0	00	1	7	
3	00	4	0	00	00	
4	2	5	-5	0	-2	
5	00	00	00	6	0	

$D^1 = \begin{bmatrix} 0 & 3 & 8 & 00 & -4 \\ 00 & 0 & 00 & 1 & 7 \\ 00 & 4 & 0 & 00 & 00 \\ 2 & 5 & -5 & 0 & -2 \\ 00 & 00 & 00 & 6 & 0 \end{bmatrix}$

confronto con  $\infty$   
la riga resto uguale  
sempre! (o colonna)

$$\begin{aligned} \min \{ \infty, \infty + 8 \} &= \infty \\ \min \{ 2, \infty + \infty \} &= \infty \\ \min \{ 7, \infty - 4 \} &= 7 \end{aligned}$$

$$\begin{aligned} \min \{ \infty, 2 + 3 \} &= 5 \\ \min \{ -5, 2 + 8 \} &= -5 \\ \min \{ \infty, 2 - 4 \} &= -2 \end{aligned}$$

$$\begin{aligned} \min \{ \infty, s + 1 \} &= 5 \\ \min \{ \infty, 4 + 7 \} &= 11 \\ \min \{ -2, 5 + 7 \} &= -2 \end{aligned}$$

$$\begin{aligned} 1 & 2 & 3 & 4 & 5 \\ 0 & 3 & 8 & 00 & -4 \\ 00 & 0 & 00 & 1 & 7 \\ 00 & 4 & 0 & 00 & 00 \\ 2 & 5 & -5 & 0 & -2 \\ 00 & 00 & 00 & 6 & 0 \end{aligned}$$

$$\begin{aligned} 1 & 2 & 3 & 4 & 5 \\ 0 & 3 & 8 & 00 & -4 \\ 00 & 0 & 00 & 1 & 7 \\ 00 & 4 & 0 & 5 & 11 \\ 2 & 5 & -5 & 0 & -2 \\ 00 & 00 & 00 & 6 & 0 \end{aligned}$$

$$\begin{aligned} 1 & 2 & 3 & 4 & 5 \\ 0 & 3 & 8 & 00 & -4 \\ 00 & 0 & 00 & 1 & 7 \\ 00 & 4 & 0 & 5 & 11 \\ 2 & 5 & -5 & 0 & -2 \\ 00 & 00 & 00 & 6 & 0 \end{aligned}$$

$$\begin{aligned} 1 & 2 & 3 & 4 & 5 \\ 0 & 3 & 8 & 00 & -4 \\ 00 & 0 & 00 & 1 & 7 \\ 00 & 4 & 0 & 5 & 11 \\ 2 & 5 & -5 & 0 & -2 \\ 00 & 00 & 00 & 6 & 0 \end{aligned}$$

$$\begin{aligned} 1 & 2 & 3 & 4 & 5 \\ 0 & 3 & 8 & 00 & -4 \\ 00 & 0 & 00 & 1 & 7 \\ 00 & 4 & 0 & 5 & 11 \\ 2 & 5 & -5 & 0 & -2 \\ 00 & 00 & 00 & 6 & 0 \end{aligned}$$

continua per

$D^2$   $D^3$   $D^4$

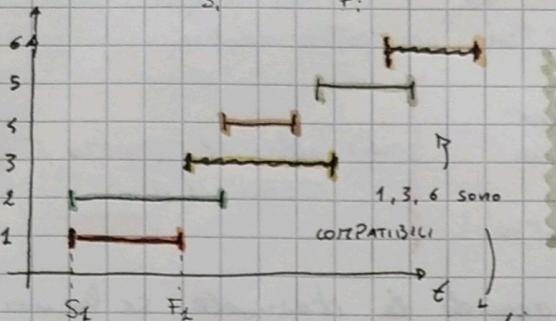
contiene tutte le  
DISTANZE MINIME!

## • ALGORITMI GREEDY

algoritmi iterativi che ad ogni step devono prendere una **DECISIONE**, prendono la più conveniente in quel momento! (problemi di ottimizzazione)

## • PROBLEMA SELEZIONE ATTIVITÀ

- $S = \{1, 2, \dots, n\}$  attività numerate con accedono a **UNA RISORSA** in COMUNE
- $F_i = i \dots n$  :  $\underbrace{s_i}_{S_i}$ ,  $\underbrace{f_i}_{F_i} \rightarrow$  tempo inizio e fine attività  $i$ -esima



ATTIVITÀ  $i = [S_i, F_i]$  e  $j = [S_j, F_j]$  sono

**COMPATIBILI** se  $[S_i, F_i] \cap [S_j, F_j] = \emptyset$  (non si sovrappongono)

$\rightarrow$  se ( $F_i < S_j$  OR  $F_j < S_i$ )

NON è detto che siano le uniche attività compatibili, ma io voglio il MAX numero d' attività compatibili!

d. esce

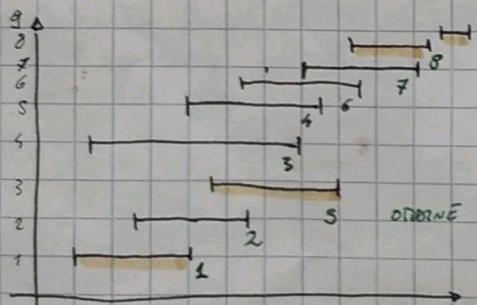
**IDEA!** ordinare le  $n$  attività in base al **tempo d'FINE**  $F_1 \leq F_2 \leq \dots \leq F_n$

## • GREEDY - ACTIVITIES-SELECTION ( $S, F$ ) /\* array dei tempi d'inizio e fine \*/

1.  $n = \text{Length}[S]$
2. ORDINA le  $n$  attività per tempo d'fine NON DECRESCENTE  $\rightarrow$  SORT-ACTIVITIES ( $F$ )
3.  $A = \emptyset$  /\* A attività compatibili, la 1<sup>a</sup> c'è sicuro \*/
4.  $j = 1$  /\* rappresenta l'ultima attività inserita in A \*/
5. for  $i = 2$  to  $n$
6. if  $S_i \geq F_j$  then **compatibilità**
7.  $A = A \cup \{i\}$  /\* aggiunge l'attività \*/
8.  $j = i$  /\* aggiorna all'ultima attività inserita \*/
9. return A

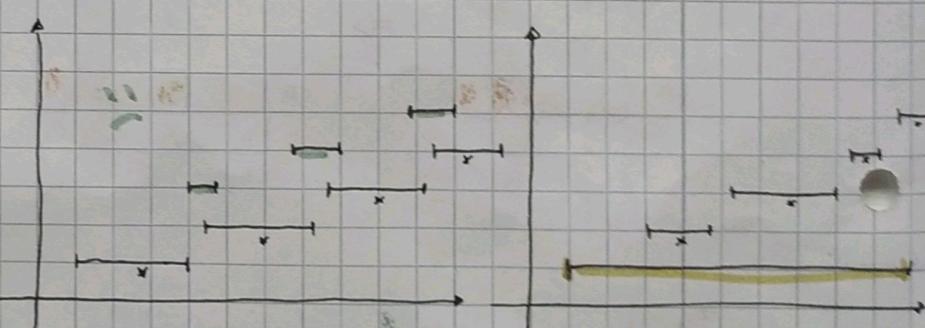
$$T_{\text{ini}} = O(n \log n) =$$

$$= O(n \log n)$$



estrai 6 attività con

«ORDINAMENTO BASATO SUL TEMPO D'FINE»



ne sceglie 3 invece delle 4

«ORDINAMENTO BASATO SULLA DURATA»

ne trova 4 invece di 5

«ORDINAMENTO BASATO SU TEMPO DI INIZIO»

INIZIO >

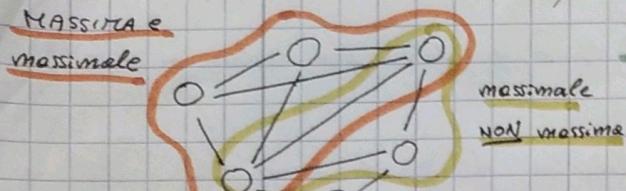
## • SCHEMA DI ALGORITMI GREEDY (non tutti, ma molti si)

1. ORDINAMENTO (secondo un certo criterio)
2.  $A \leftarrow \emptyset$
3. for each element  $x$  preso secondo l'ordine stabilito in (1)
  - if  $A \cup \{x\}$  è OK then
  - $A \leftarrow A \cup \{x\}$
6. return A

## • PROBLEMA MAX CLIQUE

$G = (V, E)$  NON ORIENTATO

- $C \subseteq V$  t.c. qualunque vertice è collegato a tutti gli altri  $G[C]$  è COMPLETO
- CLIQUE MASSITÀLE È clique D t.c.  $C \subseteq D$  ma non è contenuta in una CLIQUE MASSITÀ
- CLIQUE MASSITÀ  $|C|$  è MASSITÀ fra tutte le clique CLIQUE PIÙ GRANDE!



$$* w(G) = \max \{ |C| : C \text{ clique} \}$$

## • GREEDY-CLIQUE ( $G$ )

1. ORDINA i vertici di  $G$  secondo il NUMERO DI TRIANGOLI  $[O(n \log n)]$

2.  $C \leftarrow \emptyset$

3. PER OGNI vertice  $u$  di  $G$  estratto secondo l'ordine stabilito in (2)  $n$ -volte

4. if IS-A-CLIQUE ( $C, u$ ) then  $[O(n)]$

5.  $C \leftarrow C \cup \{u\}$

6. return  $C$

• IS-A-CLIQUE ( $C, u$ ) /\* vedo se  $\exists$   $(u, v)$  \*/

1. for each  $v \in C$   $\leftarrow O(n)$

2. if  $(u, v) \in E$  then

3. return false

4. return true

alle fine  $C$  è una CLIQUE MASSITÀ?

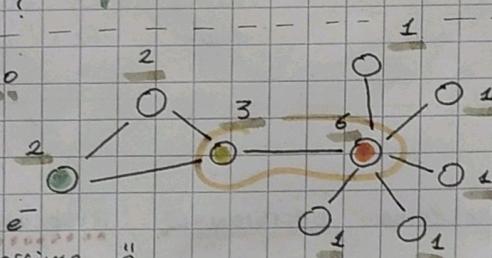
(i) ORDINAMENTO SECONDO IL GIRADITO

nope

restituisce  $\bullet - \bullet$  che è  
una CLIQUE MASSITÀ MA NON MASSIMALE

⚠ rende SEMPRE UNA CLIQUE MASSITÀ, ma NON È detto che sia anche MASSITÀ ⚠

⚠ è impossibile che un algoritmo greedy si esca e trovare la CLIQUE MASSITÀ in modo efficiente! ⚠



grad. vertici

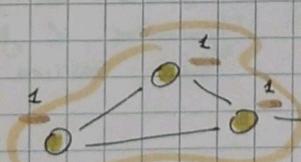
ORDINE ESTRAZIONE

1.  $\bullet$

2.  $\bullet - \bullet$  formano una clique  
NON lo può inserire

(ii) ORDINAMENTO SECONDO IL NUMERO DI TRIANGOLI

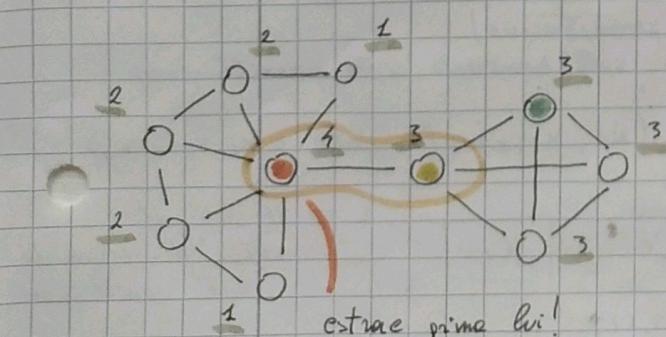
3 triangoli incidenti:



in questo caso funziona!

estrae il vertice con numero di TRIANGOLI MASSIMO  
e lo mette in  $C$ , estrae il successivo  
e lo inserisce  
poi già non può inserirlo (non si collega agli  
altri due)

alle fine rende  $C = \{\bullet, \bullet\}$



• PROBLEMI NP-COMPLETI (intractabili)

• PROBLEMA: relazione binaria tra due insiemi

$$\mathcal{P} = \mathcal{I} \times \mathcal{S}$$

istanze

solutions

$$i \in \mathcal{I} \Rightarrow d$$

$$d \in \mathcal{S}$$

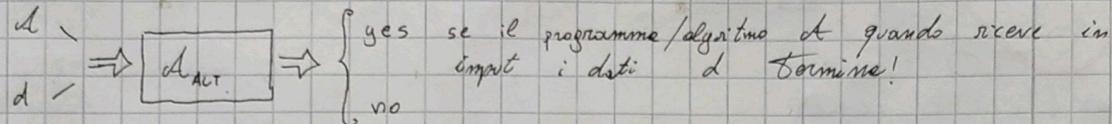
$$(i, d) \in \mathcal{P}$$

se

(ii) INDECIDIBILI

è impossibile scrivere un algoritmo che riesca a risolvere il problema, perché non terminerebbe mai!

ESEMPIO: ATM (dice se un algoritmo termina dato un algoritmo e i dati, può andare in loop!)  
ALT (problema delle fermate)



(iii) DECIDIBILE: convergono in TEMPO FINITO

O TRATTABILI: «efficiente» il problema deve essere risolto in TEMPO POLINOMIALE  $O(n^k)$

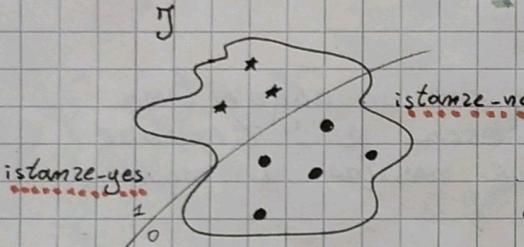
O INTRACTABILI: NON possono essere risolti in tempo polinomiale, ESPOENZIALI  $O(2^n)$

DECISIONALE \* RISPOSTA BINARIA  $S = \{0, 1\}$

CATEGORIE DI PROBLEMI

DI OTTIMIZZAZIONE

[massimizzazione/minimizzazione, sono i più frequenti]



$$i \in \mathcal{I} \Rightarrow d \Rightarrow \begin{cases} 0 \\ 1 \end{cases}$$

ESEMPIO: PROBLEMA grafo HAMILTONIANO  
vedere se esiste un ciclo in un grafo che connette tutti i nodi

hanno la stessa complessità  
dato un problema di ottimizzazione posso sempre avere un decisionale

\* P/NP/NPC

P = { $\mathcal{P} | \mathcal{P}$  è un problema DECISIONALE RISOLVIBILE polinomialmente}

P ∈ NP

NP = { $\mathcal{P} | \mathcal{P}$  è un problema DECISIONALE VERIFICABILE polinomialmente}

P ⊂ co-NP

presunte soluzioni del problema

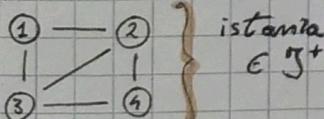
certificato  
 $i \in \mathcal{I}^+$

$$\text{ALGORITMO VERIFICA} \Rightarrow \begin{cases} \text{yes} \\ \text{no} \end{cases}$$

non mi vuol dire che il certificato NON è sufficiente a garantire il fatto che l'istanza in ingresso sia positiva

• GRAFO HAMILTONIANO

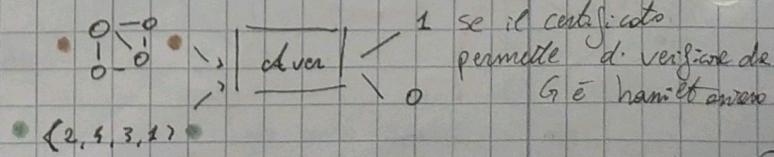
ENP!



CERTIFICATO: sequenza ordinata degli  $n$  vertici (permutazione)  
 $\langle 2, 3, 1, 2 \rangle$  identifica il cammino sull'eventuale ciclo hamiltoniano

L'ALGORITMO deve verificare che  $\forall i = 1, \dots, n-1 : (x_i, x_{i+1}) \in E$  dato  $\langle x_1, x_2, \dots, x_n \rangle$   
e  $(x_n, x_1) \in E$

DEVE avere complessità POLINOMIALE!



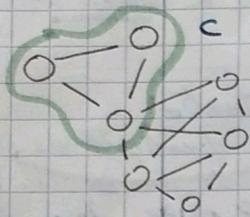
## PROBLEMA CLIQUE

E N.P!

PROBLEMA: esiste una clique in  $G$  d'  $\kappa$  vertici?

ISTANZE:  $G, \kappa$

CERTIFICATO:  $C \subseteq V$  presunta clique d'  $\kappa$  vertici



$\kappa = 3$

1. verificare che  $|C| \geq \kappa$

for  $i = 1$  to  $|C|$

for  $j = i+1$  to  $|C|$

$\Rightarrow$  oppure basta far  $i=1$  e  $j = \kappa$   
IS-A CLIQUE

2. verifica che esista l'arco tra i e j  
quando (obv.)  $i \neq j$ !

## COMPLEMENTO DI UN PROBLEMA (DECISIONALE)

P:  $G \Rightarrow \boxed{d_{RIS}}$   $\Rightarrow$  yes se  $G$  è HAMILTONIANO  
nope altrimenti

$\bar{P}: G \Rightarrow \boxed{d_{RIS}}$   $\Rightarrow$  yes se  $G$  NON è  
HAMILTONIANO  
nope

ISTANZE: tutte le possibili PERMUTAZIONI di vertici e verifica che NON ci sono cicli  
HAMILTONIANI

$n! \Rightarrow$  complessità  $\Omega(n!)$   $\Rightarrow$   $\bar{P} \notin \text{NP}$

molto più complicato dimostrare che qualcosa NON esiste!

CO-NP = {  $P$  |  $P$  sia un problema DECISIONALE tale che  $\bar{P}$  è VERIFICABILE in tempo POLINOMIALE }  $\Rightarrow \bar{P} \in \text{NP}$

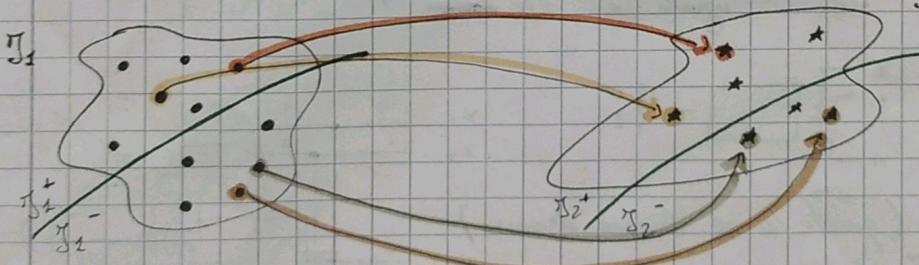
## REDUCIBILITÀ POLINOMIALE

: relazione BINARIA fra PROBLEMI DECISIONALI

$P_1 \leq_p P_2$  se esiste un algoritmo POLINOMIALE  $d_{12}$  che prende in ingresso un'istanza di  $P_1$  e la TRASFORMA in un'ISTANZA EQUIVALENTE di  $P_2$

$$\begin{aligned} P_1 &\in \mathcal{J}_1 \times \{0,1\} \\ P_2 &\in \mathcal{J}_2 \times \{0,1\} \end{aligned} \quad i \in \mathcal{J}_1 \Rightarrow \boxed{d_{12}} \Rightarrow j \in \mathcal{J}_2$$

se esiste un algoritmo per risolvere  $P_2$  alla  $\mathcal{J}_2$  posso risolvere  $P_1$



mantiene la divisione fra istanze + e -

(i)  $\leq_p$  è RIFLESSIVA

(ii)  $\leq_p$  è TRANSITIVA

dimostrazione

$\exists d_{12}$ : algoritmo polinomiale che mappa le istanze di  $P_1$  in  $P_2$

$\exists d_{23}$ : algoritmo polinomiale che mappa le istanze di  $P_2$  in  $P_3$

$$i \in \mathcal{J}_1 \rightarrow \boxed{d_{12}} \rightarrow j \in \mathcal{J}_2$$

$$j \in \mathcal{J}_2 \rightarrow \boxed{d_{23}} \rightarrow k \in \mathcal{J}_3$$

costruisco CONCATENANDO i due ALGORITMI  $d_{13}$   $i \in \mathcal{J}_1 \rightarrow \boxed{d_{12}} \xrightarrow{j \in \mathcal{J}_2} \boxed{d_{23}} \rightarrow k \in \mathcal{J}_3$

(iii)  $\leq_p$  è SIMMETRICA

se  $P_1 \leq_p P_2 \Rightarrow P_2 \leq_p P_1$

vale SOLO per sottoinsiemi di problemi per def. di NP-C

CICLO-NEGATIVO  $\leq_p$  GP  $\in$  NP

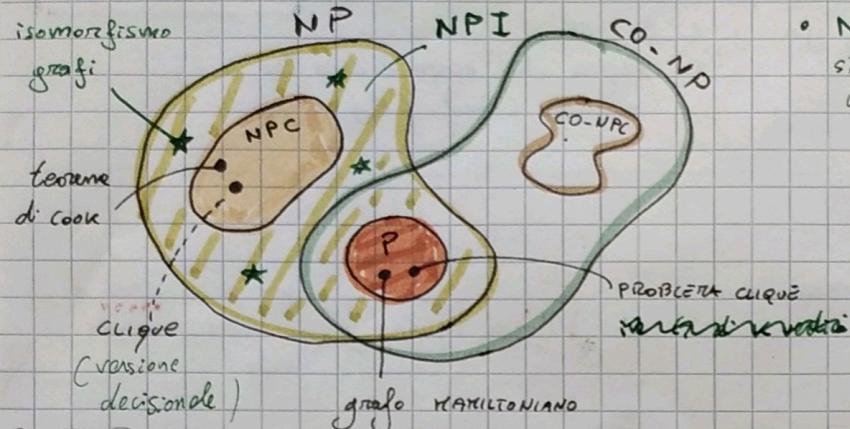
CICLO-NEGATIVO  $\leq_p$  3-SAT-NPC

NP-C

Ma il contrario no!

NOPE

$\bullet$   $NPC = \{ P \mid P \text{ è un problema DECISIONALE t.c. } P \in NP \text{ e } \nexists P' \in NP : P' \leq_p P \}$   
 è una cosa per il caso pessimo!  
 $\forall$  tutti i problemi in NPC sono riducibili l'uno all'altro  
 $P_1 \in NPC \wedge P_2 \in NPC \Rightarrow P_1 \leq_p P_2 \wedge P_2 \leq_p P_1$  vale la simmetria!  
 intrattabili!!!



$\bullet$  **NPI**: NP - indeterminato  
 stiamo in NP ma non si è dimostrato che stiamo in NPC e non si è trovato un algoritmo polinomiale  
 $\star$  per vedere se  $P \in NPC$  devo che risolvendo se  
 (i)  $P \in NP$   
 (ii)  $\nexists P' \in NP : P' \leq_p P$

### TEOREMA FONDAMENTALE NP-COMPLETITÀ

$\bullet$  se  $P \cap NPC \neq \emptyset \Rightarrow P = NP$   
 dimostrazione  
 - ipotesi:  $\exists P \in P \cap NPC$  allora  $P \in P \wedge P \in NPC$

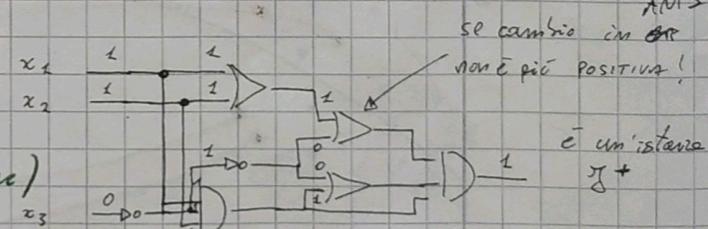
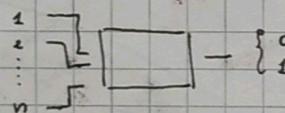
se trovo anche solo un problema NPC risolvibile in tempo polinomiale allora  $P = NP$ , altrimenti gli NPC sarebbero risolvibili in tempo polinomiale

(i)  $P \subseteq NP$ : vera, problemi risolvibili polinomialmente sono anche verificabili  
 (ii)  $NP \subseteq P$ : sia  $Q \in NP$   
 allora per definizione di NPC  $Q \leq_p P$  (istante di  $Q$  in mappa le istanze di  $Q$  in istante di  $P$  equivalenti (tempo polinomiale))  
 e per ipotesi  $P \in P$  dato che  $P \in NPC$   
 ma quindi necessariamente  $[Q \in P] \rightarrow P \in P$  quindi è risolvibile concatenando i due algoritmi  
 quindi  $P = NP$  polinomialmente

### TEOREMA DI COOK

$\bullet$  CIRCUIT-SAT: dato un circuito booleano con  $n$  ingressi e 1 uscita è possibile determinare la configurazione dei valori d'ingresso tale che l'uscita sia uguale a  $1/0$ ?

not and or  
 $\neg a = D -$



$\bullet$  CIRCUIT-SAT  $\in NPC$  (questo è il teorema di cook)

(ii) sicuramente appartiene ad NP

CERTIFICATO: configurazione valori d'ingresso  
ISTANZA: do un'istanza positiva

$\bullet$   $\nexists P' \in NP : P' \leq_p \text{CIRCUIT-SAT} \wedge \text{non è molto difficile!}$

$\bullet$   $P \in NP$  e  $\exists P' \in NPC$  t.c.  $P' \leq_p P \Rightarrow P \in NPC$   
 dim.

$\nexists Q \in NP : Q \leq_p P' \leq_p P \Rightarrow Q \leq_p P$   
 [per ipotesi] [transitività]

per dimostrare che  
 $P \in NPC$   
 faccio così!

vedo se il mio problema  $\in NP$   
 e se è un problema NPC riducibile al reo (Trasversalità)

SAT: soddisfattibilità di formula booleana

E NPC

→

SAT ∈ NP

ISTANZE: formule

CERTIFICATO: configurazione valori

→ SAT ∈ NPC

CIRCUIT-SAT ≤<sub>p</sub> SAT

$$\Phi = ((x_1 \Rightarrow x_2) \vee \neg((\neg x_1 \Leftarrow x_3) \vee x_5)) \wedge \neg x_2$$

$x_1 = 0 \quad x_2 = 0 \quad x_3 = 1 \quad x_5 = 1$

### 3-SAT-NPC

forme normale congiuntiva

$$x_1 = 1 \quad x_2 = 0 \quad x_3 = 0 \quad x_5 = 2$$

esempio:

$$\Phi = (x_1 \vee \neg x_2 \vee \neg x_4) \wedge (x_2 \vee x_3 \vee x_5) \wedge (\neg x_2 \vee \neg x_3 \vee \neg x_4)$$

• clausole

→ ogni clausola contiene ESATTAMENTE 3 letterali! (3-SAT)

$$3\text{-SAT-NPC} \leq_p \Phi = C_1 \wedge C_2 \wedge \dots \wedge C_k \quad \text{dove } C_i = (C_1^{(i)} \vee C_2^{(i)} \vee C_3^{(i)})$$

► 3-SAT-NPC ∈ NP e SAT ≤<sub>p</sub> 3-SAT-NPC ⇒ 3-SAT-NPC ∈ NPC

⚠ 2-NPC ∈ P ma k-SAT-NPC ∈ NPC  $\forall k \geq 3$  ⚡

• CLIQUE ∈ NPC \* versione decisionale di MAX-CLIQUE

dato  $G = (V, E)$  e  $k \in \mathbb{N}$ , esiste una clique in  $G$  di  $k$  vertici?

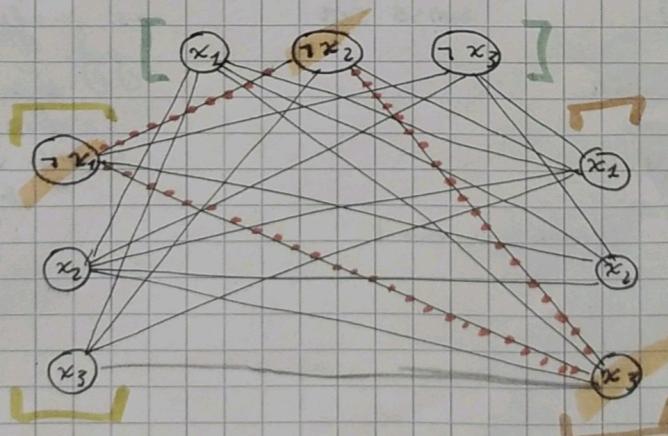
• (i) CLIQUE ∈ NP? (è verificabile polinomialmente) → pag. 25!

\* se trovasse un algoritmo polinomiale che risolve il problema delle clique essendo lui NPC, rientrare nell'ipotesi del Teorema P=NP

• (ii) 3-SAT-NPC ≤<sub>p</sub> CLIQUE

→ GRAFO: un vertice per ogni letterale ( $9$  vertici)  
Arco tra letterali di clausole diverse se i due non negazione dell'altro.

$$\Phi = (x_1 \vee \neg x_2 \vee \neg x_3) \wedge (\neg x_1 \vee x_2 \vee x_3) \wedge (x_1 \vee x_2 \vee x_3)$$



metto  $x_3 = 1$   
 $x_2 = 0$  (negazione)  
 $x_1 = 0$   
e trovo la clique

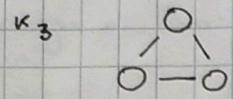
$\Phi$  è soddisfattibile  $\Leftrightarrow$  nel grafo ESISTE una CLIQUE MASSIMA di 3 vertici (3 è il numero di clausole)

$$w(G) = 3$$

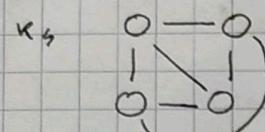
## • PROBLEMI NPC come risolverli

(i) APPROXIMAZIONE: soluzione subottimale in tempo polinomiale e a volte mi va già bene ad esempio fissando una tolleranza E entro la quale mi accontento  
MA ci sono NPC che NON si possono approssimare " (es.: clique)

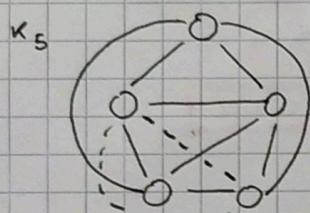
(ii) CASI PARTICOLARI:



GRAFO PLANARE: non si intersecano gli archi quando lo disegno



• SE e solo se la clique massima è di 4 vertici!



ma allora il problema della clique su grafi planari è polinomiale (cioè la costante d'averà al massimo una clique di 5 vertici)

mi interseco "

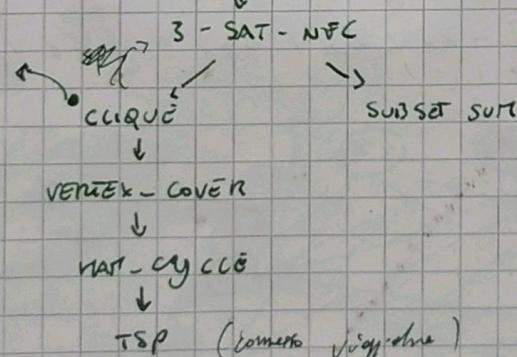
(iii) EUCLIDESCHÉ: algoritmo dove NON è dimostrata la correttezza MA funziona!

è valido sperimentalmente su molte istanze.

CIRCUIT SAT

↓ ↗ ↘ ↗  
SAT

↓



$G = (V, E)$  con una clique di  $n$  vertici

SA

corrisponde 3-SAT-DEC

clique è NP

• certificato: prescrive clique di  $n$  vertici

3 - SAT - NPC  $\leq_p$  CLIQUE

• si adatta

grado vertice + letterale

ero tre debole

linee

soddisf. se esiste una

clique di 3 vertici

• dato  $G = (V, E)$  non orientato, connesso e pesato sugli archi  $w: E \rightarrow \mathbb{R}$   
dimostrare che, se i pesi sono tutti distinti, allora esiste un unico MST!

D.M. per ASSURDO

assumo che in  $G$  esistano due resti distinti  $T_1$  e  $T_2$   
assumo che  $\exists (u, v) \in T_1$ , arco d'peso minimo

aggiungo l'arco  $(u, v)$  in  $T_2 \Rightarrow T_2 = T_1 \cup \{(u, v)\}$  ora sicuramente un ciclo!  
quindi deve esistere un altro arco  $(x, y) \in T_2$  che non sta in  $T_1$  (altrimenti  
 $T_1$  sarebbe ciclico)

essendo  $(u, v)$  d'peso minimo si ha che  $w(u, v) < w(x, y)$   
per ipotesi i pesi sugli archi sono tutti distinti  $\Rightarrow w(u, v) < w(x, y)$

dato che  $(u, v)$  ha un peso minore, sostituisco  $(x, y) \rightsquigarrow T_2 = T_1 \cup \{(u, v)\} \setminus \{(x, y)\}$   
ottenendo un  $\Rightarrow$  MST.

questo contraddice l'ipotesi di partenza che afferma che  $T_2$  fosse un resto  
ma allora deve esistere per fare un unico MST se i pesi degli archi sono  
tutti distinti!

---

sia  $G = (V, E)$  v.o. connesso  $w: E \rightarrow \mathbb{R}$   
 $(u, v)$  arco d'peso minimo  
allora  $\exists$  MST  $T$  t.c.  $(u, v) \notin T$

dim (no terremo resti)  $\Rightarrow$  Teoria Teorema cari

$G$  è connesso allora esiste almeno un MST

1. se  $(u, v) \notin T$  è banale dimo

2. se  $(u, v) \in T$

aggiungo  $(u, v)$  a  $T \Rightarrow T \cup \{(u, v)\}$

questo forma un ciclo

quindi rimovo  $(x, y)$

ottenendo un nuovo insieme di archi  $T' = T \cup \{(u, v)\} \setminus \{(x, y)\}$

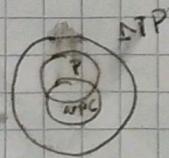
ora, dato che  $(u, v)$  è d'peso minimo  $w(u, v) \leq w(x, y)$

si ha che  $w(T') \leq w(T)$  quindi  $T'$  è MST del resto  $(u, v)$

---

VERSIÓN COMPLETA CORRIENDO! (cc)

- se  $P_1 \leq_p P_2$  e  $P_2 \in P$   $\Rightarrow P_1 \in P$
- $NP \subseteq P$  non è stata ancora dimostrata, se forse vera allora tutti i problemi sarebbero risolvibili in tempo polinomiale!
- $P \cap NP = P$  so che  $P \subseteq NP$ , quindi per le proprietà sugli insiemi è vero!
- $P \cup NPC = NP$  al momento è falsa, dato che si è dimostrato che esiste almeno un problema (isomorfismo tra grafi) che non sta né in  $P$  né in  $NPC$ !
- $P \cap NPC = \emptyset$  al momento è vero perché non è stato trovato nessun problema che appartiene a entrambe le classi
- CLIQUE  $\leq_p 3\text{-SAT-NPC}$   
so che  $3\text{-SAT-NPC} \leq_p \text{clique}$  (dimostrato a lezione)  
per definizione di  $NPC$ , tutti i problemi in  $NP$  sono riducibili a problemi  $NPC$   
dato che dice clique che  $3\text{-SAT-NPC} \in NP$  è vero che clique  $\leq_p 3\text{-SAT-NPC}$
- ISOMORFISMO-GRAFI  $\leq_p 3\text{-SAT-NPC}$   
stesso ragionamento  
 $\text{ISOMORFISMO-GRAFI} \in NP \quad \left. \begin{array}{l} \text{altra} \\ \text{OK} \end{array} \right\}$   
 $3\text{-SAT-NPC} \in NPC$
- $3\text{-SAT-NPC} \leq_p \text{ISOM.GRAFI?}$  **Nope** non è dimostrato che un problema  $NPC$  possa essere ridotto ad un  $NP$ , ma l'altro non ne è dimostrato nemmeno che  $\text{ISOM.GRAFI} \in NPC$
- CICLO-NEGATIVO  $\leq_p 3\text{-SAT-NPC}?$   
il problema del ~~ciclo~~ ciclo negativo è risolvibile in tempo polinomiale (BF)  
quindi  $\text{CICLO-NEGATIVO} \in P$  quindi  $\in NP$  che è riducibile a un qualcosa' problema in  $NPC$  (in questo caso  $3\text{-SAT-NPC}$ )
- $3\text{-SAT-NPC} \leq_p \text{CICLO-NEGATIVO}$  **Nope**
- $3\text{-SAT-NPC} \leq_p \text{GRAFI-HAMILTONIANO}$   
 $\in NPC$  è  $NP$   
impossibile!  
esiste almeno un problema che non sta né in  $P$  né in  $NPC$   
ad esempio c'è ISOMORFISMO TRA GRAFI  $\in NPC = NP \setminus P \cup NPC$



Ragionamento inversibile per il Teorema fondamentale della NP completezza  
perché affatto ~~esiste~~

$$\text{se } P \cap NPC \neq \emptyset \Rightarrow P = NP$$

quindi tutti i problemi in  $NP$  sarebbero risolvibili in tempo polinomiale, quindi  $NP \subseteq P$  ma non è stato dimostrato.

dim.  
tesi:  $P \neq NP$   $\Rightarrow P \subseteq NP \wedge NP \not\subseteq P$   
ipotesi:  $\exists Q \in P \wedge Q \in NPC \Rightarrow Q \in P \wedge Q \in NP$

(i)  $P \subseteq NP$  vero, tutti i p. risolvibili in t. polinomiale sono anche verificabili

(ii)  $NP \subseteq P$ : sia  $Q \in NP$  mappa le istanze di  $Q$  in istanze di  $Q$  equivalenti (in t. pol.) dato che  $P \in NPC$

elle per def. d'  $NPC$   $Q \leq_p Q \wedge Q \in P$  R.S. per

e per ipotesi  $Q \in P$  R.S. per

altre  $Q \in P$  (come mai i due opposti?)