

# NETWORK FLOWS

II<sup>º</sup> SEMESTRE  
2021



9/03

## INTRODUCTION

## NETWORK FLOWS



## APPROX. ALGOS

Tool: GUROBI 9.1  
Book: NETWORK FLOWS

## NETWORK OPTIMIZATION



## NP-HARD PROBLEMS

Tool: NETWORKX (PY. LIBRARY)  
Book: INTEGER PROGRAMMING  
(L. WOLSEY, EDITION 1)

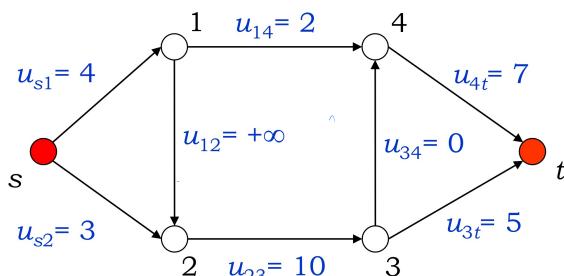
## MIDTERNS

- 1 - END MARCH
- 2 - END APRIL
- 3 - END MAY

## NETWORK DESIGN

## DIRECTED GRAPH

- $G(N, A)$  com due nodi speciali  $s$  (SOURCE) e  $t$  (SINK)
- $u_{ij} \in [0, +\infty)$  è la capacità associata a  $(i, j) \in A$



## ASSUNZIONI

- (1)  $G$  non ha un PATH DIRETTO (da  $s$  a  $t$ ) di soli archi con  $u_{ij} = 0$
- (2)  $G$  non ha ARCHI PARALLELI
- (3) Posso aggiungere archi con  $u_{ij} = 0$  come mi pare



## PATH PACKING PROBLEM

given...

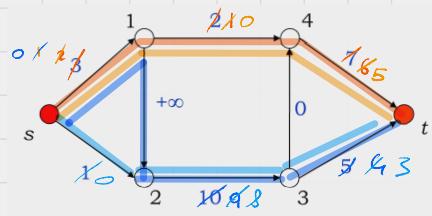
- $G(N, A)$ , un capacity vector  $u_{ij} \in \mathbb{Z}^{|A|}$

... find

- $P = \{P_1, P_2, \dots, P_k\}$  famiglia di SIMPLE DI PATH  $\subseteq$

(1) ogni  $(i, j) \in A$  fa parte di al più  $u_{ij}$  DI PATH(2)  $k$  sia MASSIMIZZATO

ES



$$\#P_1 = (s, 1, 4, t)$$

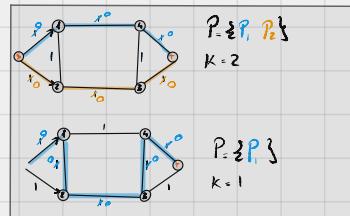
$$\#P_2 = (\dots)$$

$$\#P_3 = (s, 1, 2, 3, t)$$

$$\#P_4 = (s, 2, 3, t)$$

- Possiamo certificare l'ottimalità della soluzione?

↳ No, il greedy non è detto funziona.



LP

(Flow Formulation)

VARS.

\* (TRIVIAL) OBS

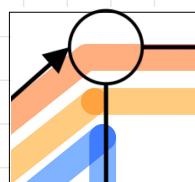
NOTA: IGNORIAMO ARCHI  
ENTRANTI IN S

\* BALANCE CONSTRAINT

CAPACITY CONSTRAINT

INTEGRITY REQ

$$\begin{aligned}
 & \max \quad \sum_{j:(s,j) \in A} x_{sj} - \sum_{j:(j,s) \in A} x_{js} \\
 & \text{s.t.} \quad \sum_{j:(i,j) \in A} x_{ij} - \sum_{j:(j,i) \in A} x_{ji} = 0 \quad \forall i \in N \setminus \{s,t\} \\
 & \qquad \qquad 0 \leq x_{ij} \leq u_{ij}, \forall (i,j) \in A \\
 & \qquad \qquad x_{ij} \text{ Integer}, \quad \forall (i,j) \in A
 \end{aligned}$$



(s,t)-Flow

FEASIBLE Flow

NET Flow IN v

Flow Value in G

• Vettore X che soddisfa i) BALANCE CONST.

• Flow che soddisfa i) CAPACITY CONST

$$\cdot \exists \text{ i termini: } f_x(v) = \sum_{j:(v,j) \in A} x_{vj} - \sum_{j:(j,v) \in A} x_{jv}$$

•  $\delta_x(v)$

• Un Flow ci dice solo le occorrenze di ogni arco, ma non ci mostra PATH.

• Dati i path  $P = \{P_1, \dots, P_k\}$  posso facilmente costruire la soluzione al problema FF

• I) contrario è vero? → dimostrando proviamo l'equivalenza tra i due problemi (PATH PACKING e FF)

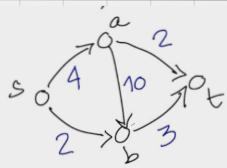
(to proof) ←

i Problemi sono equivalenti?

(trivial) ⇒

12/03

( $\Rightarrow$ ) es



$$\begin{aligned} P_1 &= \{s, a, t\} \\ P_2 &= \{s, a, t\} \\ P_3 &= \{s, a, b, t\} \\ P_4 &= \{s, b, t\} \end{aligned} \quad k = 4$$

Feasible solution to the path packing problem

$\Rightarrow$  Contract FF solution:

$$\begin{aligned} x_{sa} &= 3 & x_{at} &= 2 & x_{ab} &= 0 \\ x_{sb} &= 1 & x_{bt} &= 2 \end{aligned}$$

• Puoi controllare che soddisfa BALANCE e CAPACITY CONST.

## Demonstrando ( $\Leftarrow$ ) DECOMPOSITION THEOREM

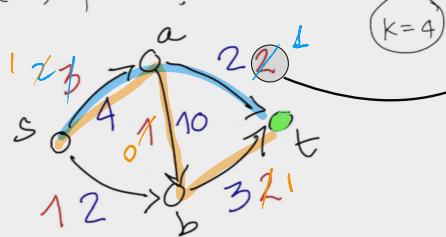
### Decomposition theorem

Given a graph  $G = (N, A)$ , vector of capacities  $\kappa \in \mathbb{Z}_+^{|A|}$   
 there exists a family  $\mathcal{P} = \{P_1, P_2, \dots, P_k\}$  of  $k$   $(s, t)$ -paths  
 such that each arc  $(i, j) \in A$  is used at most  $\kappa_{ij}$   
 times by the paths in  $\mathcal{P}$ , if and only if  
 there exists an INTEGRAL FEASIBLE  $(s, t)$  flow of  
 value  $K$

$\hookrightarrow$  Proof

es.

Suppose you have a feasible solution to FF. Is it always possible to build a family of  $K$   $(st)$ -paths?



(1) scegliamo un arco con  $x_{ij} \geq 1$   
 Dato che  $x_{ij} \geq 1$ , da Balance constraint supponiamo che è almeno un arco contratto

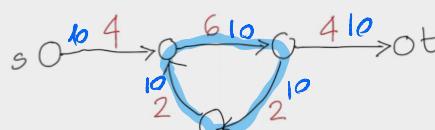
(2) Prendiamo un arco  $(i,j)$  con  $x_{ij} \geq 1$   
 In questo caso prendiamo  $(s,a)$  e abbiamo il primo PATH

$$P_1 = \{s, a, t\}$$

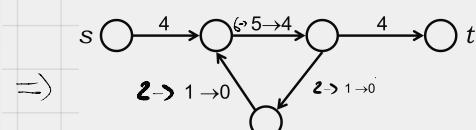
$$P_2 = \{s, a, b, t\}$$

es.

Ritorna CYCLIC FEAS. FLOW



return ACYCLIC FEAS. FLOW



- E Possibile, dato  $X$ , FEAS Flow, Possiamo trasformarlo in una aciclica?

## Cyclic To Acyclic

- Transformiamo  $X$  feasible flow in  $X'$  feas. e acyclic flow
    - (1) Identifichiamo i) archi
    - (2) sostituiamo 1 o ogni  $X_{ij}$  se solo conde finire almeno un  $X_{ij}$  non diverso  $\emptyset$
  - (1) TRASFORMA  $X$  feas flow in  $X'$  acyclic
    - (2) Partendo da  $t$ :
      - Se troviamo un vcc con  $X_{it} > 0$
      - continuando finire s mancanti path
    - (3) Ripeti (2) finché puoi
- $\mathcal{S}$

## CUT of GRAPH

- Data  $G = (N, A)$ , un TAGLIO è un insieme di archi:
 
$$\mathcal{S}(R) = \{(v, w) \mid (v, w) \in A, v \in R, w \in \bar{R}\}, R \subseteq N$$

NOTA

- IN un grafo diretto
  - $(i, j) \in \mathcal{S}(R) \cap \mathcal{S}(\bar{R})$  è in  $\mathcal{S}(R)$
  - "  $i \in \bar{R} \cap \mathcal{S}(\bar{R})$  è in  $\mathcal{S}(\bar{R})$

## CAPACITY of CUT

$$u(\mathcal{S}(R)) = \sum_{(i, j) \in \mathcal{S}(R)} u_{ij}$$

~~theorem  
WEAK DUALITY~~

- Per ogni  $(s, t)$ -cut  $\mathcal{S}(R)$  c'è ogni feas. flow  $X$  che ha

$$X(\mathcal{S}(R)) \leq X(\mathcal{S}(\bar{R})) = f_X(s)$$

↳ PROOF

- Preso  $R$ , insieme che definisce un  $(s, t)$ -cut
- Considera tutti i BALANCE CONSTRAINT associati a  $v \in R, v \neq s$

$$\sum_{j \in \mathcal{S}(i)} X_{is} - \sum_{j \in \mathcal{S}(ii)} X_{si} = 0$$

LHS                    RHS

- Dimostreremo i) teorema dividendo gli archi in 6 insiemi

ARAI INTERVI

- (1)  $\nexists (v, w) \mid v, w \in R \text{ e } v \neq s \Rightarrow x_{vw} \text{ non appare im LHS del BAL. CON}$
- (2)  $\nexists (v, w) \mid v, w \in R \Rightarrow x_{vw} \text{ non appare im LHS}$

ARAI  $R \rightarrow \bar{R}$   
e  $\bar{R} \rightarrow R$

ARAI DA S e  
IN S

- (3) Per ogni  $(v, w) \mid v \in R, w \notin R \Rightarrow x_{vw} \text{ appare im LHS con coefficiente } +1$
- (4) Per ogni  $(v, w) \mid v \in R, w \in R \Rightarrow x_{vw} \text{ appare im LHS con coefficiente } -1$
- (5) Per ogni  $(s, v) \mid v \in R \Rightarrow x_{sv} \text{ appare con coefficiente } -1$
- (6) Per ogni  $(v, s) \mid v \in R \Rightarrow x_{sv} \text{ appare con coefficiente } -1$

- RAGGIUNGENDO le varie che soddisfano (3), (4) otteniamo:

$$x(\delta(R)) - x(\bar{\delta(R)})$$

- RAGGIUNGENDO " " " (5), (6)  
invece

$$- f_x(s)$$

$\Rightarrow$  In conclusione quindi:

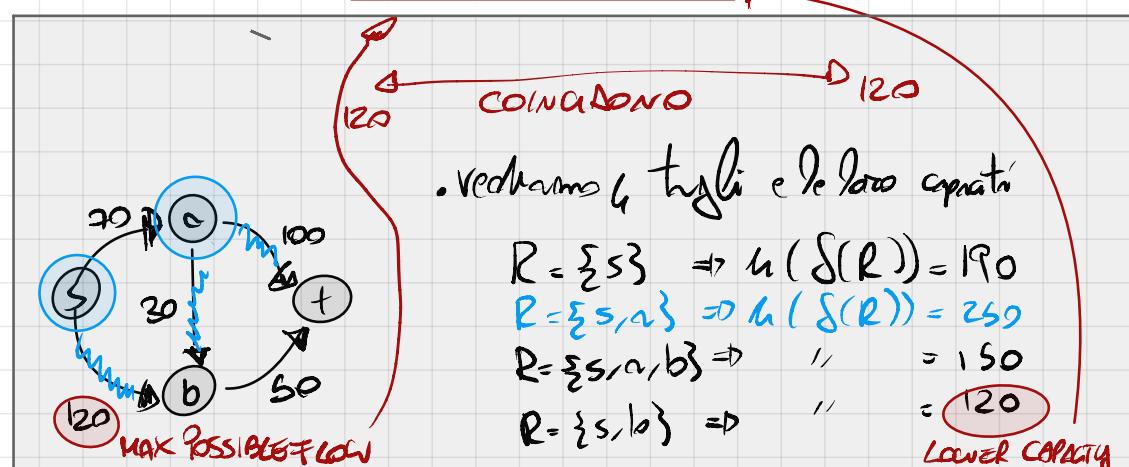
$$\text{LHS} = x(\delta(R)) - x(\bar{\delta(R)}) - f_x(s)$$



## Corollary

- $\nexists (s, t)$ -CUT  $S(R) \in \nexists (s, t)$ -Flow  $\times$  FEASIBLE

$$f_x(s) \leq u(\delta(R))$$



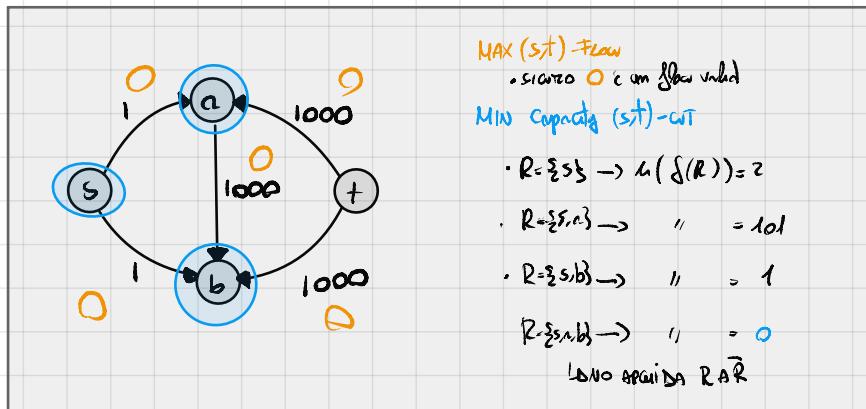
ovviamente a interessare i CUT  $\delta(R)$  con minima capacità

Esempio

## OPT. CERTIFICATE

- Con questo teorema possiamo tirarne un **CERTIFICATE D'OTTIMALITÀ**, ovvero, grazie a un **WEAK DUALITY**: una soluzione ottima del **MIN-CAPACITY-CUT** coincide con una ottima del **MAX-FLOW**

### Esempio



### ↳ Proof

- Dal Weak Duality Th. Supponiamo che

$$x(\delta(R)) - x(\delta(\bar{R})) = f_x(s)$$

- Dalla definizione

$$u(\delta(R)) \geq x(\delta(R))$$

$$\Rightarrow u(\delta(R)) \geq x(\delta(R)) - x(\delta(\bar{R})) \geq f_x(s)$$

## STRONG DUALITY TH.

[FORD e FULKERSON  
1956  
KOTzig]

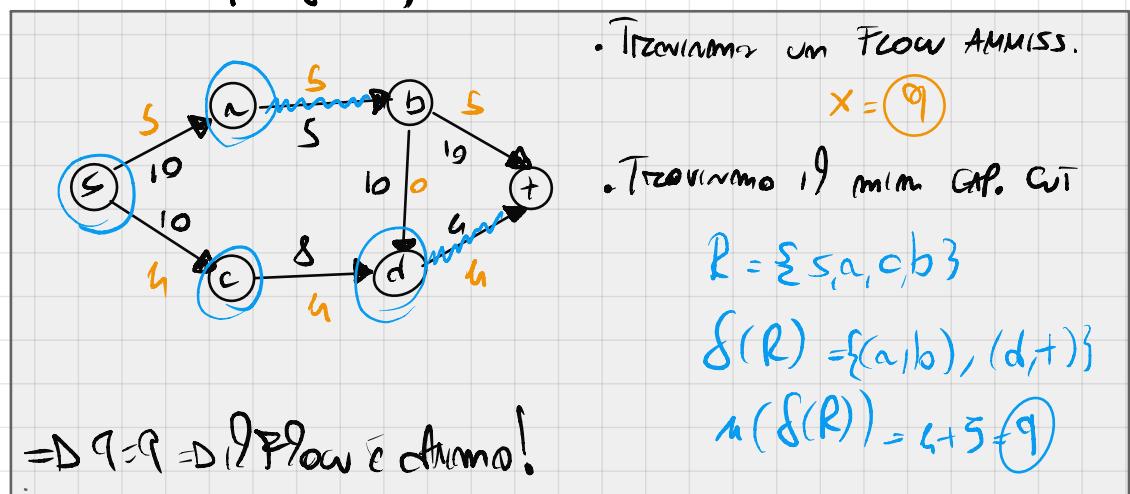
### Esempio

- Se  $G = (N, A)$  ammette  $(s, t)$  maximum flow allora

$$\max \{f_x(s) : x \text{ is a feasible } (s, t) \text{-flow}\} =$$

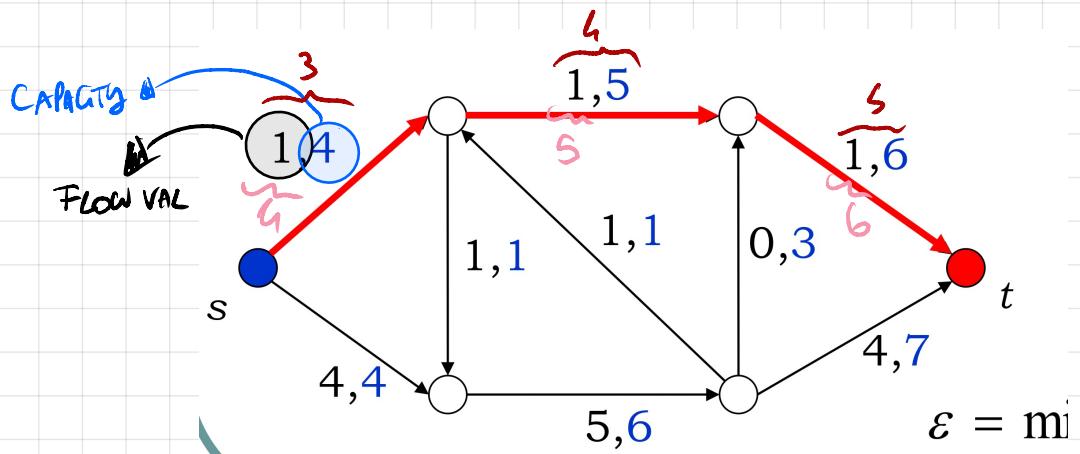
$$= \min \{u(\delta(R)) : \delta(R) \text{ is an } (s, t) \text{-cut}\}$$

(ci dice che il max flow ha sempre valore UGUALE al min capacity cut)



## ↳ Proof

- SE IL FLOW VAL È UGUALE ALLA CAPACITY, L'ARCO È SATURATED
- SE NON È SATURATED POSSIAMO INCREMENTARE IL FLOW?



- Aggiorniamo la soluzione mettendo  $F_{\text{flow}} = F_{\text{curr}} + \varepsilon$   
dove sta  $\varepsilon = \min_{(i,j) \in P} \{u_{ij} - x_{ij}\} = 3$ , quindi  $(1,4)$  che diventa  $6,6$ . Poi se il loro  $\varepsilon = 3$
- Sì, NON ROMPIAMO IL CAPACITY CONSTRAINT, MA IL BAGNO!
- Cambiando il flow nel  $(s-t)$ -path di una quantità  $\varepsilon$  manteniamo per costituzione, il  $F_{\text{flow}}$  feasible

## MAX FLOW (ALGO)

(1) Impossibilizza soluzione  $x_{ij} = 0 \forall (i,j) \in A$

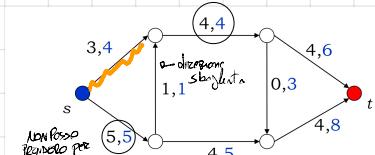
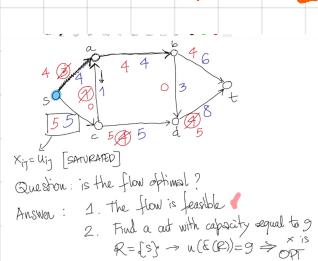
With  $P \neq \emptyset$

(1) cerca in  $G$  un  $(s-t)$ -path  $P$  /c  $x_{ij} < u_{ij} \forall (i,j) \in P$

(2) incrementa il Flow  $x$  su  $P$  della quantità  $\varepsilon = \min_{(i,j) \in P} \{u_{ij} - x_{ij}\}$

• TERMINATION? OPT. SOL? COMPLEXITY?

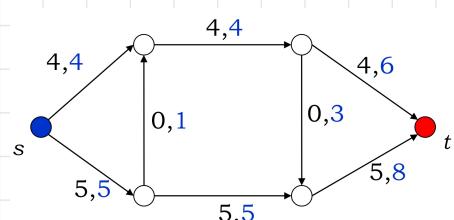
### OPT. SOL.



There is no  $(s, t)$  path  $P$  such that  $x_{ij} < u_{ij}$  for all arc  $(i, j) \in P$ , but the flow is not optimal.

• Come possa dunque esistere un optimal flow come questo nella figura accanto?

① Non c'è un  $(s-t)$ -path che rispetti la definizione



STRONG DUALITY TH.  
[FORD e FULKERSON  
1956  
KOTzig]

↳ Proof:

## FORWARD/REVERSE ARCS

- Considerare ogni  $(s-t)$ -PATH  $P$ , un arco può essere un FORWARD ARC (diretto da  $s$  a  $t$ ) o REVERSE ARC (altrimenti)

## AUGMENTING PATH

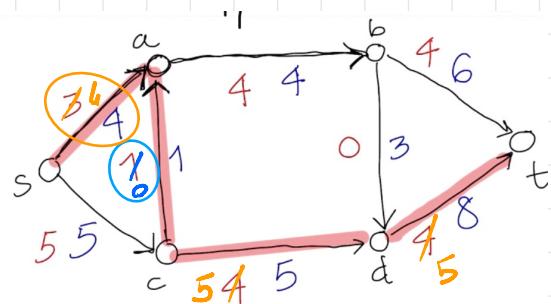
- Un  $(s-t)$ -PATH  $P$  è ogni forward arc ha  $x_{ij} < u_{ij}$  e ogni reverse arc ha  $x_{ji} > 0$

$$\mathcal{E}_1 = \min_{(i,j) \in P} \{u_{ij} - x_{ij}\} = 1$$

$\tau_c(i,j)$  is forward

$$\mathcal{E}_2 = \min_{(i,j) \in P} \{x_{ji}\} = 1$$

$\tau_c(i,j)$  is reverse



- Se incontriamo un FORWARD SUC PATH incrementiamo il flow di  $\mathcal{E}_1$ , se invece incontriamo un REVERSE decrementiamo di  $\mathcal{E}_2$

→ Tale tecnica ci porta da un FEAS. SOL a un altro FEAS.

L'ALGO TROVA UNA SOLUZIONE OPTIMA?

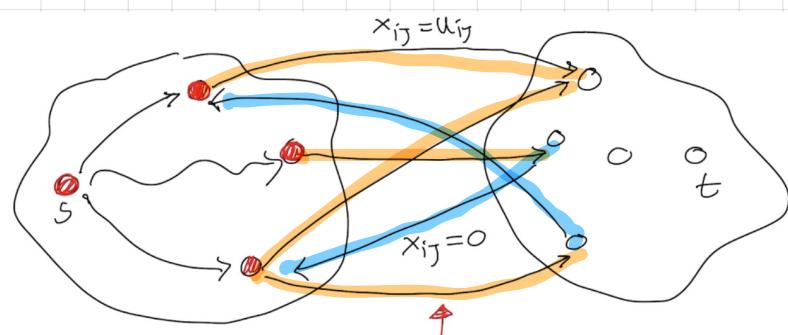
RIC

- Work duality TM per ottenere un certificato di ottimalità dobbiamo avere un Flow e un CUT  $\mathcal{C}$   $\delta_x(s) = u(\delta(R))$

SUPPONIAMO MAX FLOW  $x$

I nodi Rossi sono quegli raggiungibili tramite augmenting PATH  
I Bianchi no  
Gli Arci in NERO sono forward o reverse

$$\begin{cases} x_{ij} = u_{ij}, \text{ Per i forward} \\ x_{ji} = 0, \text{ Per i reverse} \end{cases}$$



$$\delta_x(s) = \underbrace{x(\delta(R))}_{u(\delta(R))} - \underbrace{x(\delta(\bar{R}))}_{=0} = u(\delta(R))$$

# FORD e Fulkerson ALGO

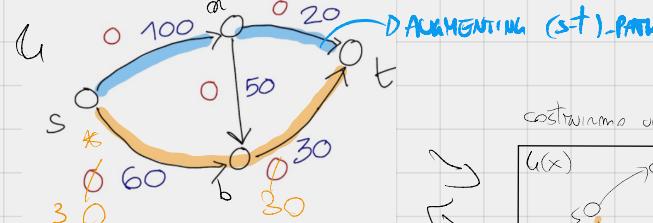
(1) INITIALIZZA il flow  $x=0$

WHILE  $P_i \neq 0$

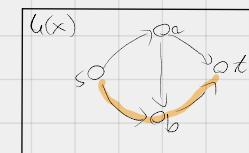
(2) Cerca  $(s-t)$ -AUGMENTING PATH

(3) Incrementa il flow sul path di  $x = \min\{e_i\}$

ESEMPIO



costruiamo un AUXILIARY GRAPH  $G(x)$

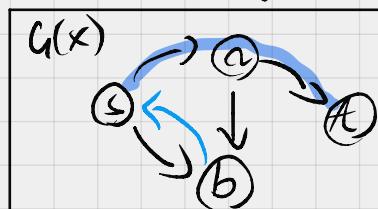


- Costruiamo un oriented (s-t)-path
- Usiamo questo sul grafo di percorrere e incrementare il flow \*

Dopo aver trovato l'orienting PATH  
E INCREMENTANDO IL FLOW CALCOLAMO  
 $\delta_x(s) = u(\delta(t))$

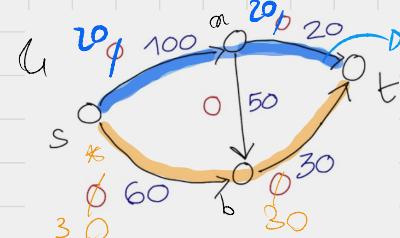
→ Disegniamo il grafo con solo gli archi NON-SATURATI

→ Possiamo raggiungere nodi **reverse** dove il flow è  $x_{is} > 0$



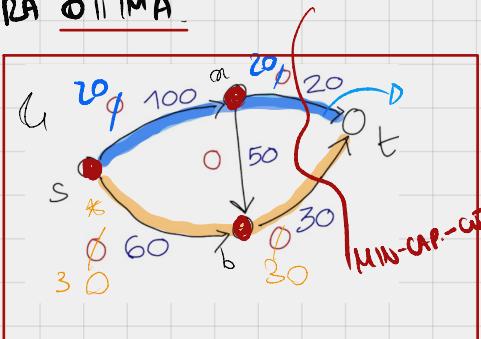
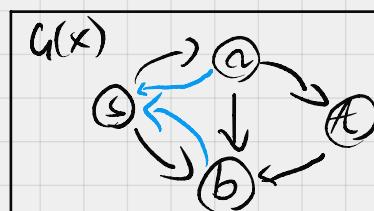
• TROVIAMO UN ORIENTED (s-t)-PATH

• AGGIUNGIAMO AL GRAFO SCAGLIANDO I FLOW

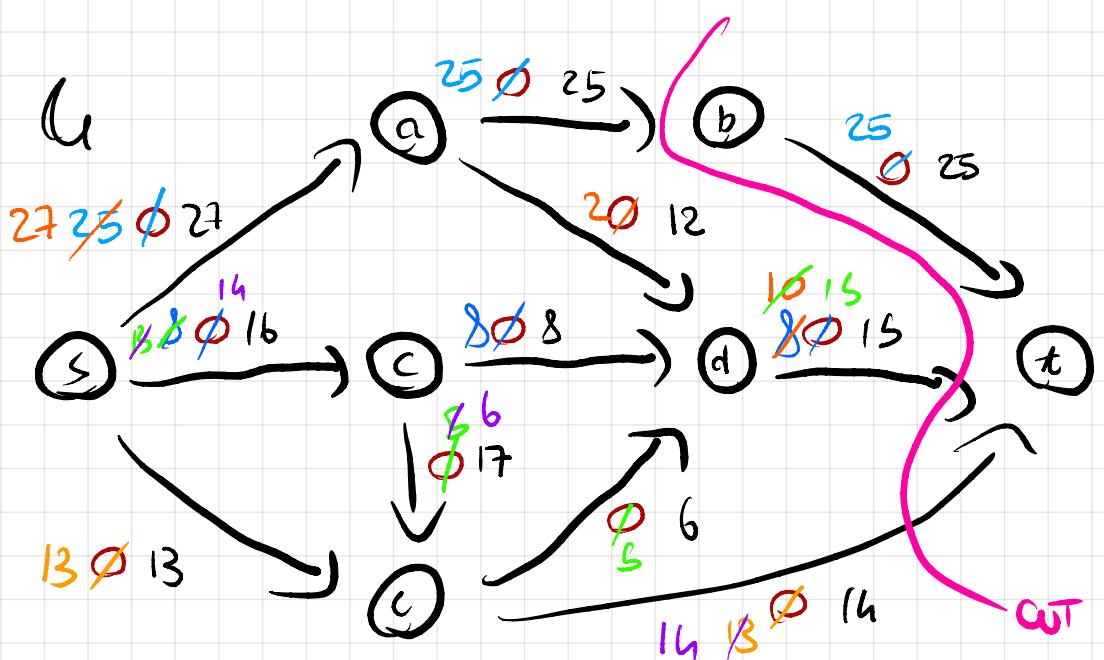


- Continuiamo
- non c'è più un AUGMENTING PATH

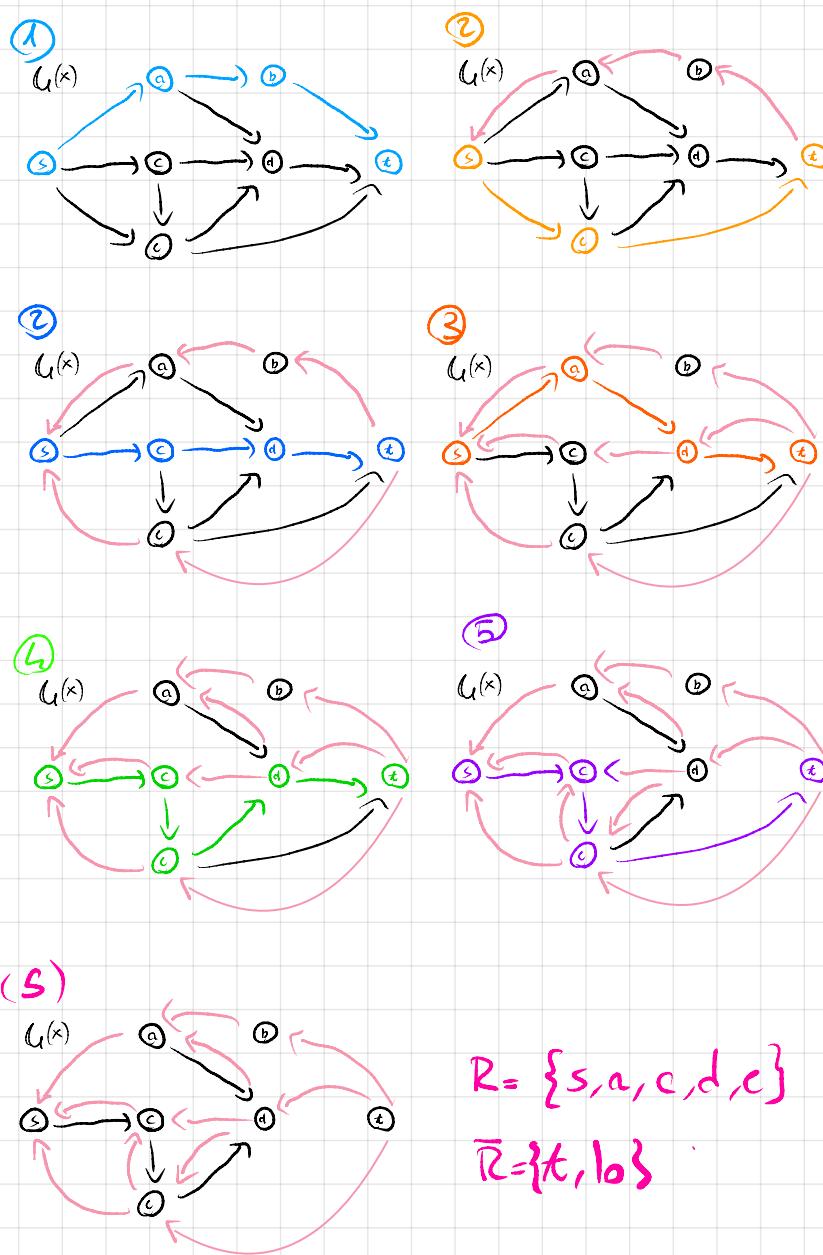
⇒ L'ultima soluzione è OTTIMA



esempio



$$\Rightarrow \delta_x(s) = 25 + 38 + 8 + 2 + 5 + 1 = 79$$



$$R = \{s, a, c, d, e\}$$

$$\bar{R} = \{t, 10\}$$