NETWORK FLOWS

II° 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)

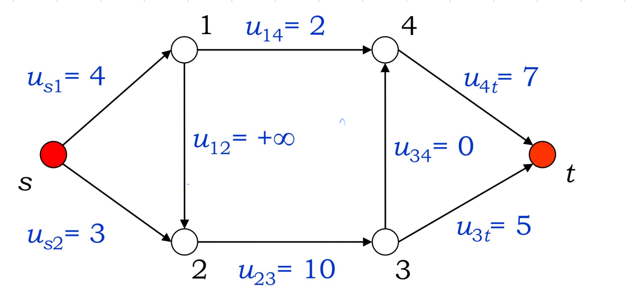
Book: INTEGER PROGRAMMING (L. WOLSEY, EDITION 1)

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

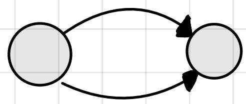
NETWORK DESIGN

DIRECTED GRAPH · $G(N, A)$ con due nodi speciali s (SOURCE) e t (SINK)

CAPACITY · $u_{ij} \in [0, +\infty)$ 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 $u_{ij} = \infty$
(2) G non ha ARCHI PARALLELI



(3) Posso aggiungere archi con $u_{ij}=0$ come mi pare

PATH PACKING PROBLEM

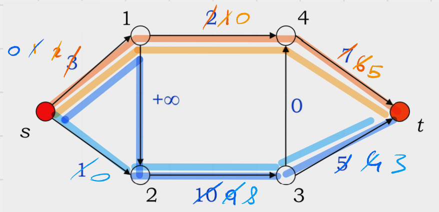
given... · $G(N,A)$, un capacity vector $u_{ij} \in \mathbb{Z}^{|A|}$

...find · $P = \{P_1, P_2, \ldots, P_k\}$ famiglia di SIMPLE DIPATH t/c

(1) ogni $(i,j) \in A$ fa parte di al più u_{ij} DIPATH

(2) k sia MASSIMIZZATO

ES

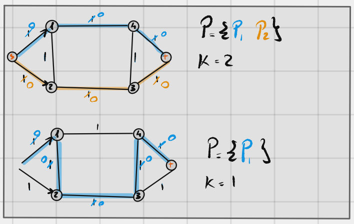


$\# P_1 = (s,1,4,t)$ $\# P_2 =$ "

$\# P_3 = (s,1,2,3,t)$ $\# P_4 = (s,2,3,t)$

• Possiamo certificare l'ottimalità della soluzione?

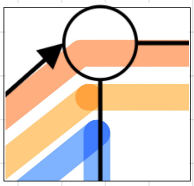
↳ No, il greedy non è detto funzioni.



ILP (Flow Formulation)

VARs. • X_{ij} occorrenze di (i,j) ed nei path in P

* (trivial) OBS • $\forall v \neq s,t \Rightarrow$ il # path entranti è lo stesso di quelli uscenti



NOTA: ignoriamo archi entranti in s

$$\max \sum_{j:(s,j)\in A} x_{sj} - \cancel{\sum_{j:(j,s)\in A} x_{js}}$$

s.t.

* BALANCE CONSTRAINT
$$\sum_{j:(i,j)\in A} x_{ij} - \sum_{j:(j,i)\in A} x_{ji} = 0 \quad \forall i \in N \setminus \{s,t\}$$

CAPACITY CONSTRAINT
$$0 \le x_{ij} \le u_{ij}, \ \forall (i,j) \in A$$

INTEGRALITY REQ
$$x_{ij} \text{ Integer}, \quad \forall (i,j) \in A$$

(s,t)-FLOW • Vettore X che soddisfa il BALANCE CONSTR.

FEASIBLE FLOW • Flow che soddisfa il CAPACITY CONSTR

NET FLOW IN v • È il termine: $f_x(v) = \sum_{j:(v,j)\in A} x_{vj} - \sum_{j:(j,v)\in A} x_{jv}$

FLOW VALUE IN G • $\delta_x(v)$

---

i Problemi sono equivalenti?

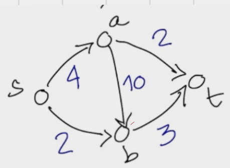
• Un Flow ci dice solo le occorrenze di ogni arco, ma non ci mostra i path.

(trivial) ⇒ • Dati i path $P = \{P_1 \_ P_K\}$ posso facilmente costruire la soluzione al problema FF

(to proof) ⇐ • Il contrario è vero? → dimostrandolo proviamo l'equivalenza tra i due problemi (Path packing e FF)

12/03

(⇒) es



$P_1 = \{s,a,t\}$
$P_2 = \{s,a,t\}$
$P_3 = \{s,a,b,t\}$
$P_4 = \{s,b,t\}$

$k=4$

Feasible solution to the path packing problem

⇒ Construct FF solution:

$x_{sa} = 3$ $\quad x_{at} = 2$ $\quad x_{ab} = 0$

$x_{sb} = 1$ $\quad x_{bt} = 2$

· Poi controllare che soddisfa BALANCE e CAPACITY CONSTR.

Dimostriamo (⇐)

DECOMPOSITION THEOREM

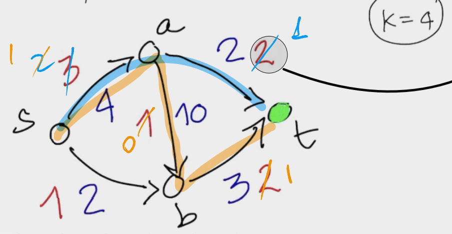
Decomposition theorem

Given a graph $G=(N,A)$, vector of capacities $u \in \mathbb{Z}_+^{|A|}$ 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 u_{ij} times by the paths in $\mathcal{P}$, if and only if there exists an INTEGRAL FEASIBLE (s,t) flow of value k

↳ Proof

es.

Suppose you have a feasible solution to FF. Is it always possible to build a family of k (s,t)-paths?



(1) Scegliamo un arco con $x_{ij} \geq 1$

Dato che $x_{st} \geq 1$, dal Balance constraint sappiamo che c'è almeno un arco entrante

(2) Prendiamo un arco (i,a) con $x_{ia} \geq 1$

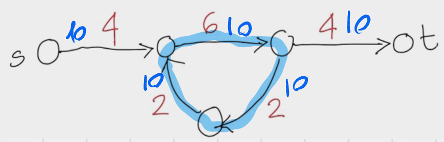
In questo caso prendiamo (s,a) e abbiamo il primo path

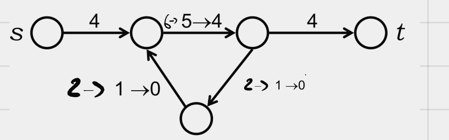
$P_1 = \{s,a,t\}$

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

es.

Ritorna CYCLIC FEAS. FLOW ⇒ RETURN ACYCLIC FEAS. FLOW





• È possibile, dato x, FEAS FLOW, possiamo trasformarlo in una aciclica?

CYCLIC TO ACYCLIC

- Trasformiamo X feasible flow in X' feas e acyclic flow
  (1) Identifichiamo il ciclo
  (2) sottraiamo 1 a ogni X_{ij} sul ciclo finchè almeno un X_{ij} non diventa $\emptyset$

ALGO

- (1) TRASFORMA x feas flow in X' aciclico
  (2) Partendo da t:
  - selezioniamo un arco con $X_{it} > 0$
  - continuiamo finchè s non è nel path
  (3) Ripeti (2) finchè puoi

$\int$

CUT of GRAPH

- Dato $G=(N,A)$, un TAGLIO è un insieme di archi:

$$\delta(R) = \{(v,w) \mid (v,w) \in A,\ v \in R,\ w \in \bar{R}\},\ R \subseteq N$$

NOTA

- IN un grafo diretto
  - (i,j) t/c $i \in R$ e $j \in \bar{R}$ è in $\delta(R)$
  - " $i \in \bar{R}$ e $j \in R$ è in $\delta(\bar{R})$

CAPACITY of CUT

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

theorem WEAK DUALITY

- Per ogni (s,t)-cut $\delta(R)$ e ogni feas. flow x allora

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

↳ PROOF
- Preso R, insieme che definisce un (s,t)-CUT
- Considera TUTTI i BALANCE CONSTRAINT associati a $v \in R$, $v \neq s$

$$\boxed{\underbrace{\sum_{j\in\delta^+(i)} x_{ij} - \sum_{j\in\delta^-(i)} x_{ji}}_{LHS} = \underbrace{0}_{RHS}}$$

- Dimostreremo il teorema dividendo gli archi in insiemi

6

ARCHI INTERNI

(1) $\forall (v,w) \mid v,w \in R$ e $v \neq s \Rightarrow x_{vw}$ NON appare in LHS del BAL. CON

(2) $\forall (v,w) \mid v,w \notin R \Rightarrow x_{vw}$ NON appare in LHS

ARCHI $R \to \overline{R}$ e $\overline{R} \to R$

(3) Per ogni $(v,w) \mid v \in R, w \notin R \Rightarrow x_{vw}$ appare in LHS con coefficiente +1

(4) Per ogni $(v,w) \mid v \notin R, w \in R \Rightarrow x_{vw}$ appare in LHS con coefficiente -1

ARCHI DA S e IN S

(5) Per ogni $(s,v) \mid v \in R \Rightarrow x_{sv}$ appare con coefficiente -1

(6) Per ogni $(v,s) \mid v \in R \Rightarrow x_{sv}$ appare con coefficiente -1

- RAGGRUPPANDO le vars che soddisfano (3),(4) otteniamo:

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

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

$$-f_x(s)$$

⇒ In conclusione quindi:

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

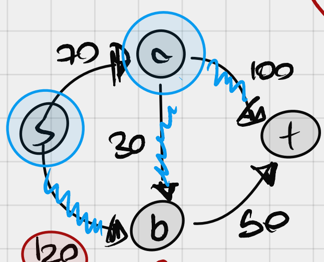
Corollary

ovviamente a interessa il cut $\delta(R)$ con minima capacità

- $\forall (s,t)$-CUT $\delta(R)$ e $\forall (s,t)$-FLOW x FEASIBLE

$$\boxed{f_x(s) \leq u(\delta(R))}$$

esempio



MAX POSSIBLE FLOW

120 COINCIDONO 120

- vediamo i tagli e le loro capacità

$R = \{s\} \Rightarrow u(\delta(R)) = 190$

$R = \{s,a\} \Rightarrow u(\delta(R)) = 250$

$R = \{s,a,b\} \Rightarrow$ " $= 150$

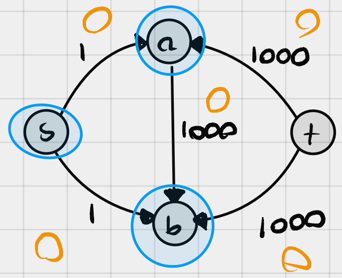
$R = \{s,b\} \Rightarrow$ " $= 120$

LOWER CAPACITY

OPT. CERTIFICATE

• Con questo teorema possiamo trarre un CERTIFICATO DI OTTIMALITÀ, ovvero, grazie alla WEAK DUALITY: una soluzione ottima del MIN-CAPACITY-CUT coincide con una ottima del MAX-FLOW

esempio



MAX (s,t)-FLOW
• sicuro 0 è un flow valido

MIN Capacity (s,t)-CUT
• $R=\{s\} \rightarrow u(\delta(R))=2$
• $R=\{s,a\} \rightarrow$ " $=101$
• $R=\{s,b\} \rightarrow$ " $=1$
$R=\{s,a,b\} \rightarrow$ " $=0$

↳ NO ARCHI DA R A $\bar{R}$

↳ Proof

• Dal WEAK DUALITY TH. sappiamo che

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

• Dalla definizione

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

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

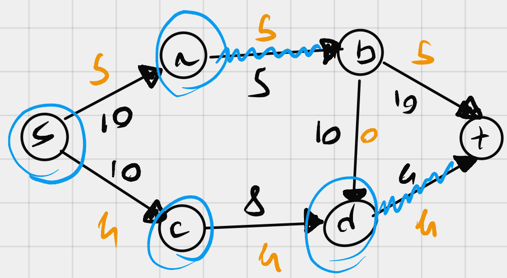
STRONG DUALITY TH.
[FORD e FULKERSON 1956 KOTZIG]

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

$$\boxed{\begin{aligned}&\max\{f_x(s) : x \text{ is a feasible } (s,t)\text{-flow}\} = \\ &= \min\{u(\delta(R)) : \delta(R) \text{ is an } (s,t)\text{-cut}\}\end{aligned}}$$

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

esempio



• Troviamo un FLOW AMMISS.
$x=9$

• Troviamo il min CAP. CUT

$R=\{s,a,c,b\}$

$\delta(R)=\{(a,b),(d,t)\}$

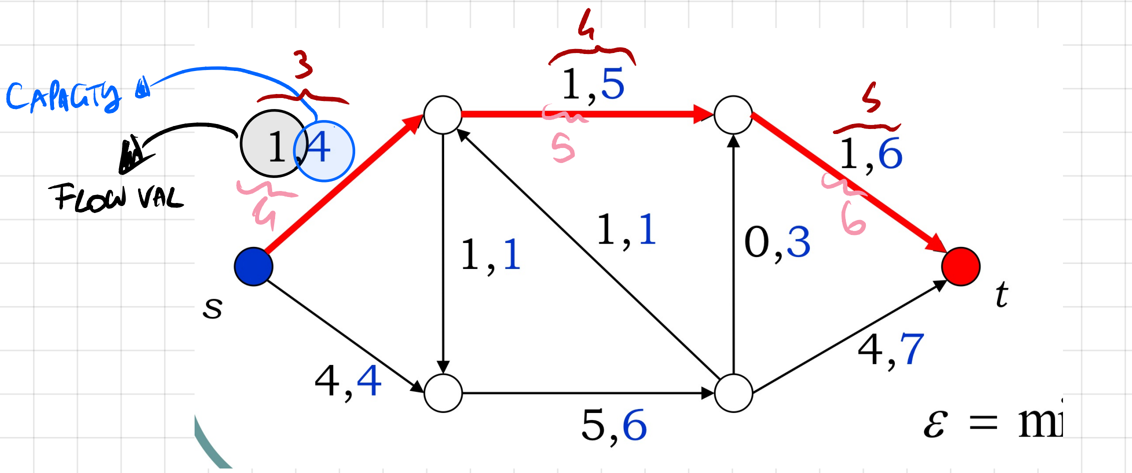
$u(\delta(R))=4+5=9$

$\Rightarrow 9=9 \Rightarrow$ il flow è ottimo!

↳ Proof

• SE IL FLOW VAL È uguale alla CAPACITY, l'arco È SATURED

• se non è saturated possiamo incrementare il flow?



• Aggiorniamo la soluzione mettendo FLOW = FLOW + ε dove sta $\varepsilon = \min_{(i,j)\in P} \{u_{ij} - x_{ij}\} = 3$, quindi (1,4) che diventa 4,4, Poiché il loro $\varepsilon = 3$

• Sì, NON ROMPIAMO il CAPACITY CONSTRAINT, ma il BALANCE?

• Cambiamo il flow nell'(s-t)-PATH di una quantità ε mantiene, per costruzione, il Flow feasible

---

MAX FLOW (ALGO)

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

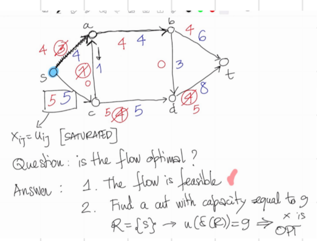
While $P \neq \emptyset$

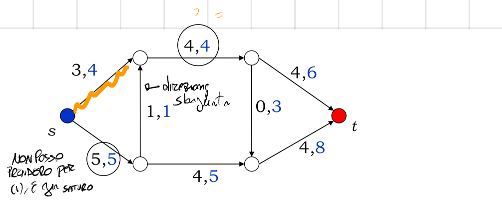
(1) cerca in G un (s-t)-PATH P t.c. $x_{ij} < u_{ij} \quad \forall (i,j) \in P$

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

• TERMINATION? OPT. SOL? COMPLEXITY?

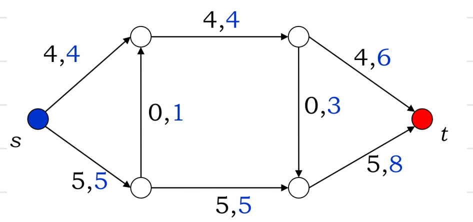
OPT. SOL.





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.

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



• Come posso arrivare a un optimal flow come questo nella figura accanto?

STRONG DUALITY TH.
[FORD e FULKERSON 1956
KOTZIG]

↳ Proof:

FORWARD/REVERS ARCs

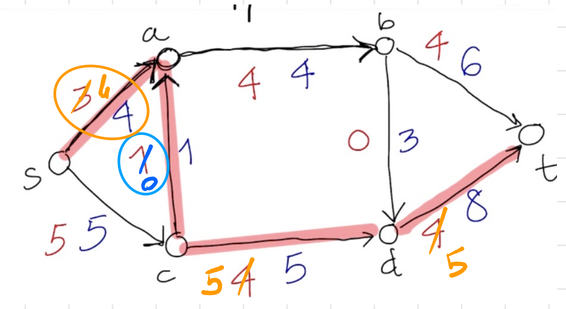
- Considera ogni (s-t)-PATH P, un arco può essere un
  FORWARD ARC (diretto da s a t)
  REVERSE ARC (altrimenti)

AUGMENTING PATH

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

$$\varepsilon_1 = \min_{\substack{(i,j)\in P \\ \text{t/c } (i,j) \text{ is forward}}} \{u_{ij} - x_{ij}\} = 1$$

$$\varepsilon_2 = \min_{\substack{(i,j)\in P \\ \text{t/c } (i,j) \text{ is reverse}}} \{x_{ij}\} = 1$$



- Se incontriamo un FORWARD sul PATH incrementiamo il flow di ε_1, se invece incontriamo un REVERSE decrementiamo di ε_2

⇒ Tale tecnica ci porta da una FEAS. SOL a un altra FEAS.

L'ALGO TROVA UNA SOLUZIONE OTTIMA?

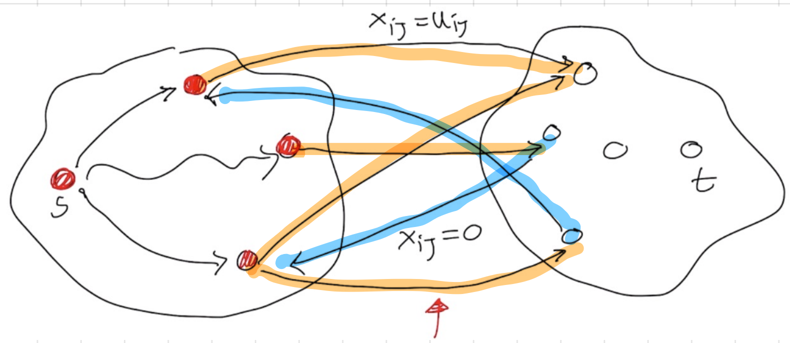
R|C

- Weak duality TH per ottenere un certificato di ottimalità dobbiamo avere un flow e un CUT t/c $\boxed{f_x(s) = u(\delta(R))}$

- SUPPONIAMO MAX FLOW x

I NODI ROSSI SONO QUELLI RAGGIUNGIBILI TRAMITE AUGMENTING PATH
I BIANCHI NO
GLI ARCHI IN MEZZO SONO FORWARD o REVERSE

$$\begin{cases} x_{ij} = u_{ij}, & \text{per i forward} \\ x_{ji} = \emptyset, & \text{per i reverse} \end{cases}$$



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

FORD e FULKERSON ALGO

(1) INIZIALIZZA il flow $x = 0$

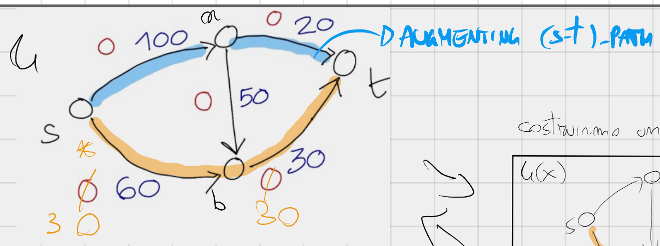
DO

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

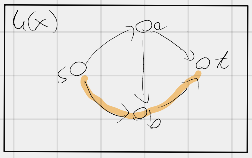
(3) Incrementa il flow sul path di $x \min\{\varepsilon_1, \varepsilon_2\}$

WHILE $P \neq 0$

esempio



costruiamo un AUXILIARY GRAPH G(x)

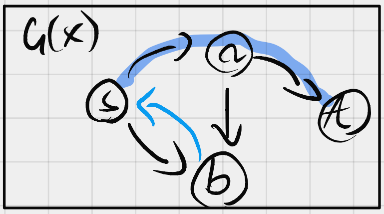


- Cerchiamo un oriented (s-t)-path
- Usiamo questo sul grafo di partenza e incrementiamo il flow *

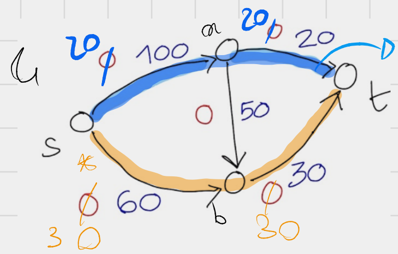
- Dopo aver trovato l'ORIENTED PATH E INCREMENTATO IL FLOW CALCOLIAMO $\delta_x(s) = u(\delta(R))$

→ Disegnamo il grafo con solo gli archi NON-SATURATI

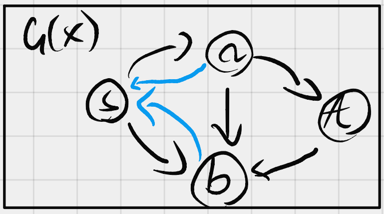
→ Possiamo aggiungere archi reverse dove il flow è $x_{ij} > 0$



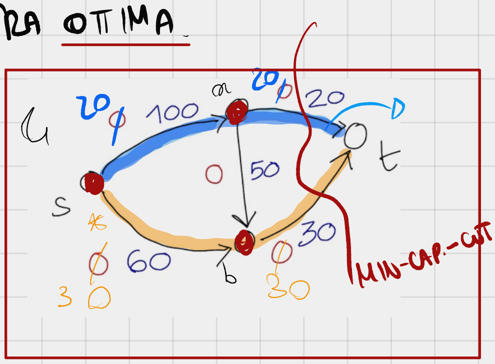
- TROVIAMO UN ORIENTED (s,t)-PATH
- AGGIUNGIAMOLO AL GRAFO SCALANDO i FLOW



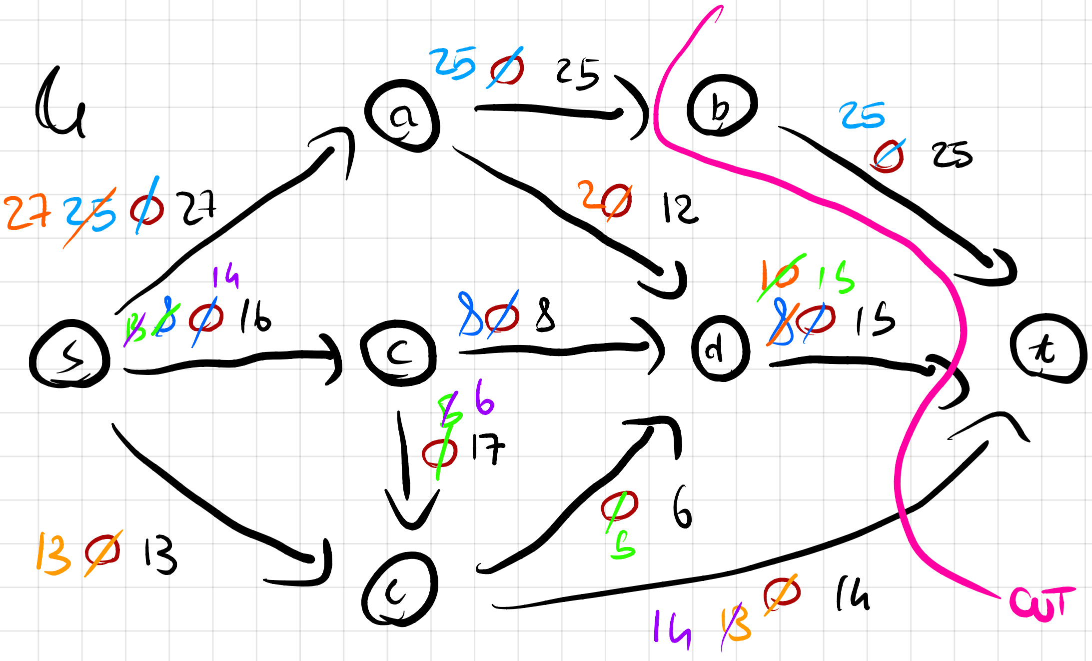
- Continuiamo
- non c'è più un AUGMENTING PATH.



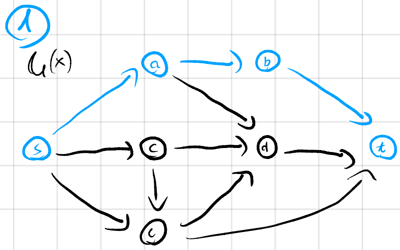
⇒ L'ULTIMA SOLUZIONE ERA OTTIMA.

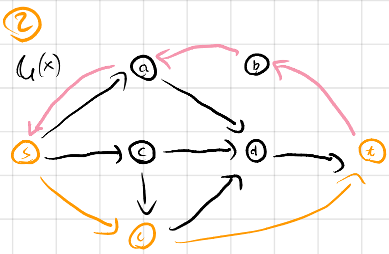


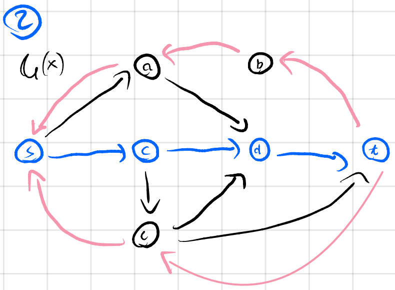
esempio

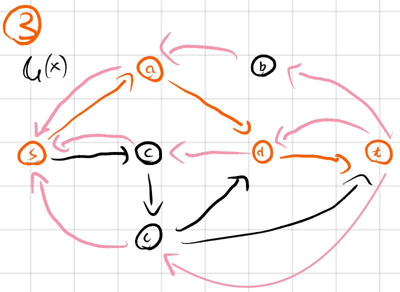


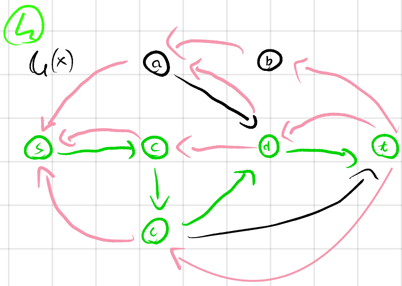
$$\Rightarrow \delta_x(s) = 25 + 38 + 8 + 2 + 5 + 1 = 79$$

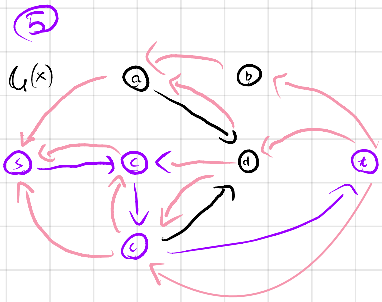


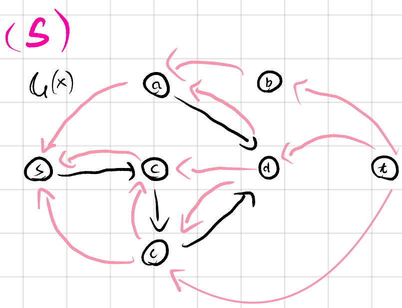












$R = \{s, a, c, d, e\}$

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

SOMMARIO

Max flow - min Cut

· Se un grafo $G=(N,A)$ ammette un (s,t) maximum flow allora
$\max\{f_x(s) \mid x$ è una FEAS. SOL.$\} = \min\{u(\delta(R)) \mid \delta(R)$ è un (s,t)-CUT$\}$

Ford and Fulkerson (ALGO)

(è feasible per il Th sopra)
(termina per quello sotto)

· Inizialization (un qualsiasi flow ammissibile)
$x=0$
DO
  Cerca in G un (s,t)-AUGMENTING PATH
  Incrementa il flow su P di $x \min\{\varepsilon_1, \varepsilon_2\}$
WHILE $P \neq \emptyset$

K

upper bound per K: la capacità è m.U

$O(Km) = O(mU_K)$

TH · Un feasible flow x è ottimo $\Leftrightarrow$ non ci sono augmenting path in G

EDMONDS AND KARP

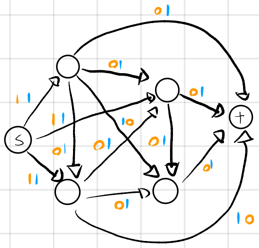
· Se a ogni step di F&F scegliamo lo SHORTEST AUG. PATH, la complessità è $O(n\,m^2)$
Perché lo visitiamo con un BFS

· COME RAGGIUNGIAMO UNA COMPLESSITÀ POLINOMIALE?
↳ L'unico modo di farlo è sfruttare le giuste istanze del problema:
  (1) Perché partire da $x=0$?
  (2) selezionare l'augmenting path che porta al massimo incremento del flow

23/03 ESERCIZI PARZIALE

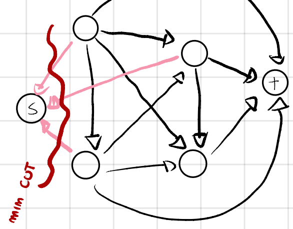
(1) · Sia $G=(N,A)$ un grafo diretto
· Trova il minimo numero di archi la cui rimozione disconnetta s da t

⇒ · Risolviamo trovando il max flow con F&F in modo da avere anche il min CUT
· Possiamo iniziare settando le CAPACITY a 1



TROVIAMO UN FLOW FEASIBLE:
↳ $f_x(s)=3$
↳ x è feasible

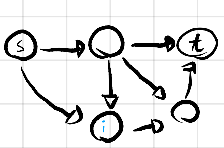
⇒



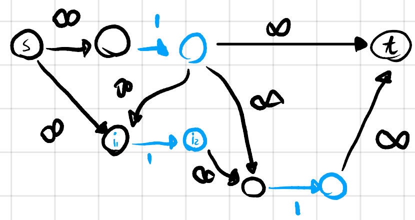
· Alla prima iterazione non troviamo alcun AUGM. PATH, quindi X è massimo e corrisponde al min CUT

(2) · Sia $G=(N,A)$ un grafo diretto e s,t due nodi di N
· Trova il minimo numero di nodi la cui rimozione disconnetta s da t

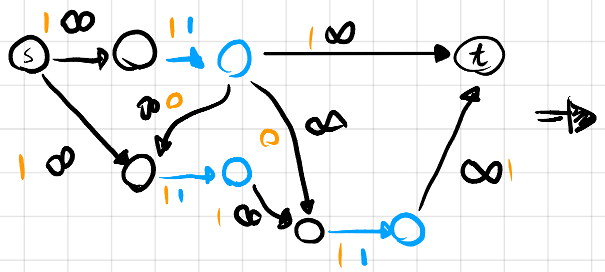
⇒ · Node splitting raddoppio tutti i nodi tranne s,t e inizializzo le capacity a ∞ sui vecchi archi e 1 sui nuovi



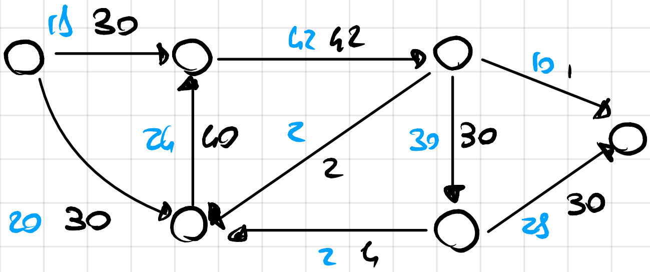
⇒



• Vediamo col flow

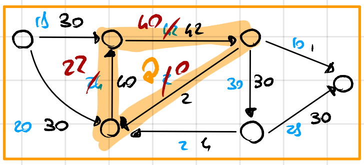


(3)



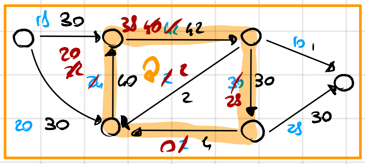
(1) • Il flow è feasible?
(2) • Ci sono flow equivalenti che non inducono un ciclo?
(3) • Trova il massimo numero di (s,t)-PATH che possono essere PACKED

1 ↳ Capacity e Balance constrs soddisfatti
2 ↳ Ecco il ciclo

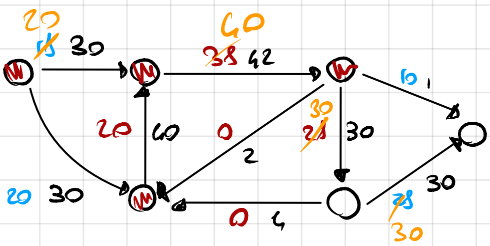


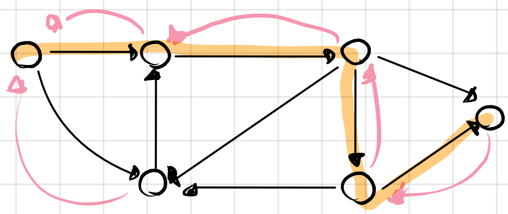
Applichiamo la tecnica discussa nella prova del TH.1
→ RIDUCIAMO IL FLOW sul ciclo fino a portare a ∅ il flow dell'arco che lo aveva più piccolo

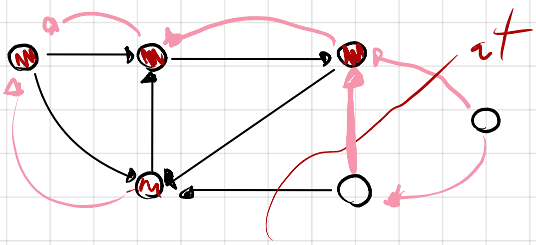
↳ finiamo in un altro ciclo, allora REITERIAMO:



3 ↳ Applichiamo max (s,t)-FLOW

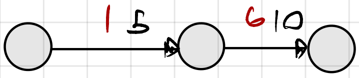






FLOWS WITH LOWER BOUNDS

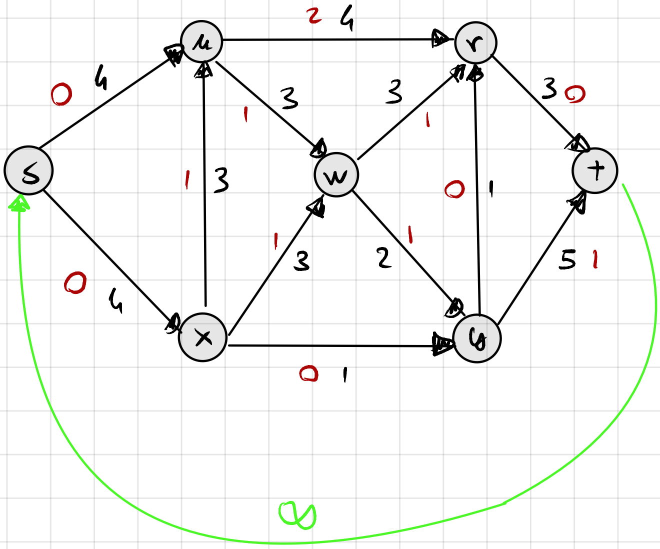
- Se impostiamo di requisiti di minimo sul Flow, esiste sempre una FEAS. SOL.?



// NON ESISTE PERCHÉ IL MINIMO FLOW RICHIESTO DA (b,c) È 6, E IL MASSIMO SU (a,b) È 5

FEASIBILITY CHECK

#MIN-REQs : l_{ij}



CIRCULATION PROBLEM

- BALANCE CONSTRAINTS

$s: \ x_{su} + x_{sx} - x_{ts} = 0$

$u: \ x_{uv} + x_{uw} - x_{su} - x_{xu} = 0$

$v: \ x_{vt} - x_{uv} - x_{wv} - x_{yv} = 0$

$w: \ x_{wv} + x_{wy} - x_{uw} - x_{xw} = 0$

$x: \ x_{xu} + x_{xw} + x_{xy} - x_{sx} = 0$

$y: \ x_{yv} + x_{yt} - x_{wy} - x_{xy} = 0$

$t: \ x_{ts} - x_{vt} - x_{yt} = 0$

- BOUNDS DI x_{ij} → $\boxed{l_{ij} \leq x_{ij} \leq u_{ij}}$

- Sostituiamo tutti gli x_{ij} → $\boxed{x_{ij} = x'_{ij} + l_{ij}}$

$$\Rightarrow l_{ij} \leq x_{ij} \leq u_{ij} \underset{x_{ij} = x'_{ij} + l_{ij}}{\Rightarrow} l_{ij} \leq x'_{ij} + l_{ij} \leq u_{ij} \Rightarrow \boxed{\overbrace{\cancel{l_{ij} - l_{ij}}}^{\phi} \leq x'_{ij} \leq u_{ij} - l_{ij}}$$

- i BALANCE CONSTRs diventano

$s: \ x_{su} + x_{sx} - x_{ts} = 0$

$u: \ x_{uv} + x_{uw} - x_{su} - x_{xu} = \cancel{0} \ -2$

$v: \ x_{vt} - x_{uv} - x_{wv} - x_{yv} = \cancel{0} \ 3$

$w: \ x_{wv} + x_{wy} - x_{uw} - x_{xw} = 0$

$x: \ x_{xu} + x_{xw} + x_{xy} - x_{sx} = 0$

$y: \ x_{yv} + x_{yt} - x_{wy} - x_{xy} = 0$

$t: \ x_{ts} - x_{vt} - x_{yt} = 0$

→ $u: \ x'_{uv} + l_{uv} + x'_{uw} + l_{uw} - x'_{su} - l_{su} - x'_{xu} - l_{xu} = 0$

$\Rightarrow x'_{uv} + 2 + x'_{uw} \cancel{+1} - x'_{su} \cancel{-0} - x'_{xu} \cancel{-1} = 0$

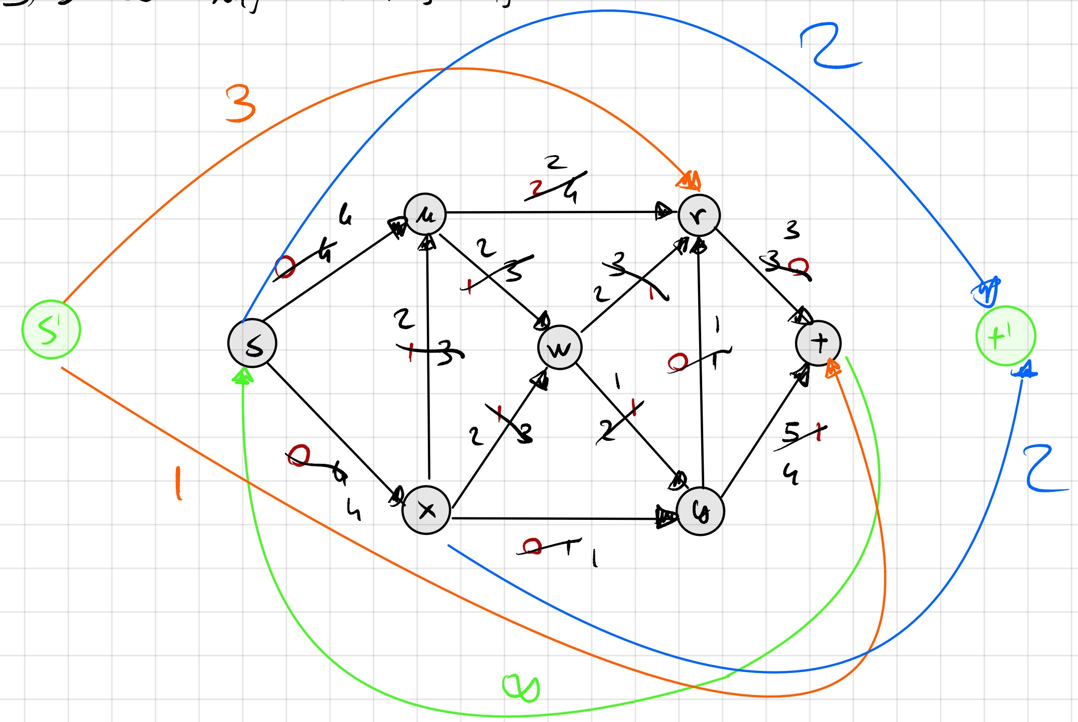
$\Rightarrow x'_{uv} + x'_{uw} + x'_{su} = -2$

- Ora trasformiamo ciò che abbiamo in un FLOW PROBLEM

(1) Per ogni nodo i con $RHS_i > 0$ aggiungi un arco (s', i) con $u_{s'i} = RHS_i$

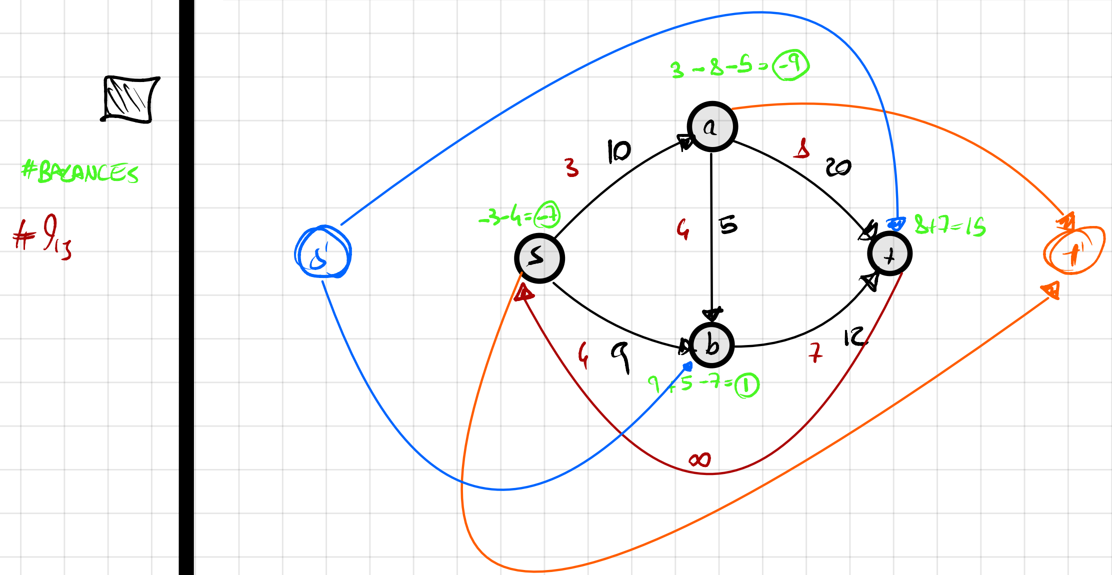
(2) Per ogni nodo j con $RHS_j < 0$ aggiungi un arco (j, t') con $u_{ij} = RHS_j$

(3) SCALO u_{ij} con $u_{ij} - x_{ij}$



$\Rightarrow$ s' e t' sono connessi a UNBALANCED NODE, cioè con $RHS \neq 0$

- Se riusciamo ora a trovare un (s',t')-PATH che satura gli archi nuovi, troviamo un feasible flow per il problema originale
- Se non riusciamo $\Rightarrow \nexists$ FEAS SOL.



- ORA BASTA APPLICARE F&F, se non riusciamo $\Rightarrow$ G non ammette feas. flow