18/04/2024 | LEZ 27

NEW ARGO:

PROBLEMA MASSIMO FLUSSO

⇒ da una sorgente, confluire la massima quantità di materiale in una destinazzione

problema studiato gia dagli anni 50 e 60.

ARGO. OGGI:

- DESCRIZIONE PROBLEMA MAX FLUSSO/MIN CUT.
- ALGO. RISOLUTIVO.

DESCRIZIONE PROBLEMI

- → MAX FLOW
- → MIN CUT

⇒ DIPENDONO 1 DALL'ALTRO X RISOLVERE PROBLEMA PRINCIPALE (SEE NEXT LEZ).

SI BASA SULLO STESSO INPUT, UNA RETE DI FLUSSO

- GRAFO G:
  - SORGENTE s
  - CONNESSO
  - DIRETTO
  - CAPACITÀ → VALORE ASSOCIATO ∀ ARCO e. ⇒ $c(e)$.

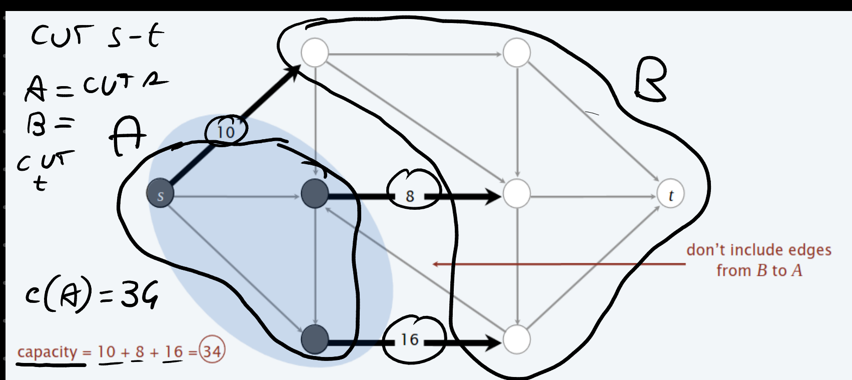
1° PROBLEMA MIN-CUT

DEF

- UN s-t-CUT: PARTIZIONE DI 2 SET (A, B) DI NODI, t.c. $s \in A$, $t \in B$. ⇒ PARTIZIONE ARBITRARIA.
- CAPACITÀ s-t = SOMMA CAPACITÀ ARCHI USCENTI ~~DA~~ A a B.

ESEMPIO DI:

- s-t CUT
- CAPACITÀ CUT



PROBLEMA: 🔍 s-t CUT CON CAPACITÀ MIN.

HOW?

BRUTE FORCE? NOP. ⇒ 2^{m-2} POSSIBILI PARTIZIONI DA CONTROLLARE!

2° PROBL.

MAX FLOW

DEF

- UN **St - FLOW** è UNA FUNZIONE, $E \rightarrow \mathbb{N}$, DOVE:

INSIEME EDGE

REQUISITI DA SODDISFARE
- FOR EACH e: $0 \leq f(e) \leq c(e)$ → MANTENIMENTO DI CAPACITÀ
- $\forall v \in V - \{s,t\}$: $\sum_{e \text{ IN } v} F(e) = \sum_{e \text{ OUT } v} F(e)$ → CONSERVAZIONE DEL FLUSSO

- VALORE DI UN FLOW F:

$$VAL(F) = \sum_{e \text{ IN } s} f(e) - \sum_{e \text{ OUT } s} F(e)$$

PROBLEMA: ⚲ F con VAL(F) MAX.

ALGORITMI

1° APPROCCIO: ALGORITMO GREEDY.

non funzionarà ma ci darà una idea su come risolvere il problema

PROCEDIMENTO:

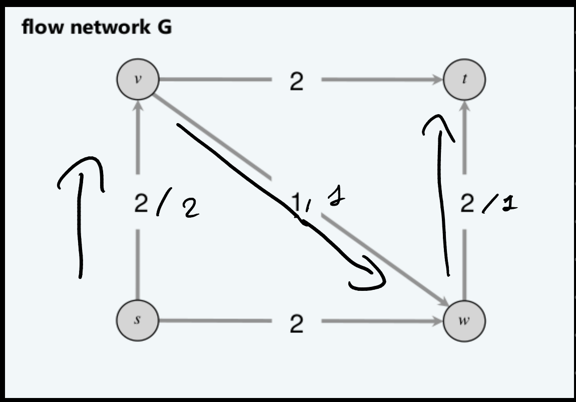
- BEGIN $\forall$ OGNI $w = 0$
- ⚲ PATH $s \rightsquigarrow t$ DOVE $f(e) \leq c(e)$
- ++ FLOW $\times \forall$ ARCO DEL PATH → VALORE = VALORE CAPACITÀ MINIMA DI ARCO.
- RIPETO UNTIL $\nexists$ PATH $s \rightsquigarrow t$.

CORRETTEZZA → SBAGLIATA!

WHY NON FUNZIONA?

- QUANDO ++ VAL FLOW IN UN ARCO, $\not\exists$ LO -- PIÙ, → quando gli sceglie il valore, quello rimane
  ↳ E QUESTO ALLE VOLTE, $\not\exists$ PORTA AL MASSIMO

CONTROESEMPIO.



IL GREEDY POTREBBE SCEGLIERE IL PATH $S \to V \to W \to t$,

DA QUI:

- ARCO CENTRALE INUTILIZZABILE
- ALTRI ARCHI NON POSSONO ARRIVARE A 2,

(ARCO IN MEZZO ROMPE IL CAZZO X MANTENIMENTO FLUSSO.)

$VAL(F) = 3$

$MAX_{VAL} = 4$

COSA FARE? DOBBIAMO FARE IN MODO CHE UNA 'CATTIVA SCELTA' POSSA ESSERE ANNULLATA.

DA QUI NUOVI STRUMENTI.

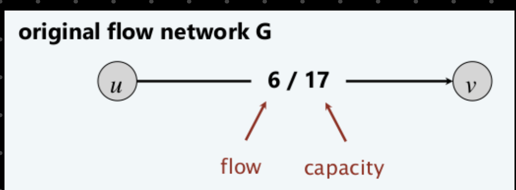
DEF

- UNA RETE RESIDUA $G' = (V, E_F, s, t, c_F)$:

① $\forall$ ARCO $(u, v) \to$ AL PIÙ 2 ARCHI, DOVE:

IF $F(e) > 0 \Rightarrow$, G_F: NUOVO ARCO $e_{REVERSE}$ (ARCO INVERSO)

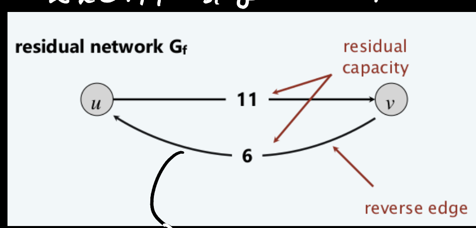
EDGE DI G



$\Rightarrow$

ARCHI DI G'

$c_F(e) = c(e) - F(e)$



QUANTO POSSO SPINGERE

POSSO "TOGLIERE" X RIMETTERLO IN U.

② CON 2 CARATT: CAP. $= c(e) - f(e)$

- $E_F = \{e : f(e) < c(e)\} \cup \{e : f(e^{REVERSE}) > 0\}$

$\hookrightarrow$ ARCHI CON CAPAC. RESIDUA

- PROPRIETÀ KEY: f' = FLOW IN $G_F \iff F + F'$ FLOW IN G

- UN CAMMINO AUMENTANTE P' = PATH $s \rightsquigarrow t$ IN G_F.
- LA CAPACITÀ BOTTLENECK IN P = VALOR. MINIMO DI CAPACITÀ DEGLI ARCHI IN P.

DA QUI SFRUTTIAMO UN PSEUDOCODICE (AUGMENT), CHE, SFRUTTA LA PROPIETÀ KEY x ++ FLOW.

CODICE

AUGMENT(f, c, P)

$\delta \leftarrow$ bottleneck capacity of augmenting path P.

FOREACH edge $e \in P$:

  IF ($e \in E$) $f(e) \leftarrow f(e) + \delta$.

  ELSE $f(e^{\text{reverse}}) \leftarrow f(e^{\text{reverse}}) - \delta$.

RETURN f.

→ CONTROLLA CHE SIA IN e o e^{REVERSE}

→ ++ FLOW DI f

→ -- f DI e DI δ

QUESTO CI SERVIRÀ x L'ALGO

ALGORITMO → FORD-FULKERSON.

PROCEDIMENTO.

1. BEGIN → $\forall e$ $F(e) = 0$
2. P IF $\exists$ $s \rightsquigarrow t$ PATH IN G_F
3. AUGMENT FLOW SU P → AGGIORNO G_F
4. RIPETO FINO A QUANDO PUOI.

CODICE

FORD–FULKERSON(G)

FOREACH edge $e \in E$: $f(e) \leftarrow 0$.

$G_f \leftarrow$ residual network of G with respect to flow f.

WHILE (there exists an $s \rightsquigarrow t$ path P in G_f)

  $f \leftarrow$ AUGMENT(f, c, P).

  Update G_f.

RETURN f.

augmenting path

→ IMMAGINO CREI NUOVI ARCHI IN BASE AL VALORE DI $F(e)$.

QUESTO CODICE DÀ 2 RISULTATI:

- MAX FLOW

- MIN CUT → TUTTI I NODI RAGG. DA S IN G_F

DOVE $C(A) = VAL(F)$ → SEMPRE

(A = MIN_CUT)

NEXT PDF → CORRETTEZZA ∧ COMPLEXITY DI ALGO FORD-FULKERSON.