Proiectarea Algoritmilor

Curs 10 – Rețele de flux. Flux maxim.

Bibliografie

[1] C. Giumale – Introducere în Analiza Algoritmilor - cap. 5.6

[2] Cormen – Introducere în algoritmi - cap. Flux Maxim (27)

[3] Wikipedia - http://en.wikipedia.org/wiki/Ford-Fulkerson_algorithm

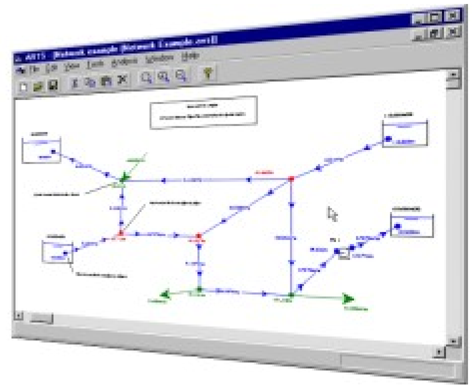
[4] R. Sedgewick, K Wayne – curs de algoritmi Princeton 2007 www.cs.princeton.edu/~rs/AlgsDS07/ 01UnionFind si 14MST

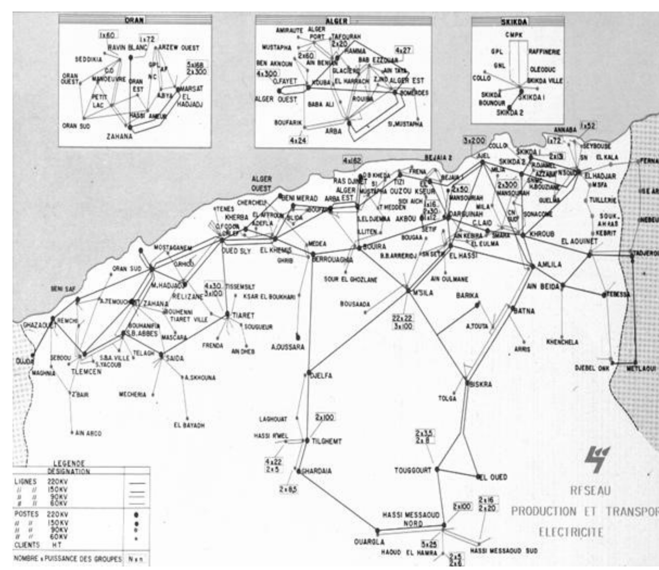
Obiective

- Definirea conceptului de rețea de flux (sau de transport).
- Identificarea principalilor algoritmi ce calculează fluxul maxim printr-o rețea.

Definirea problemei

- Rețea ce transportă diferite materiale între un producător și o destinație.
- Fiecare arc are o capacitate maximă de transport.
- Trebuie identificat fluxul maxim ce poate fi transportat prin rețea.
- Rețele:
  - Electrice;
  - Apă;
  - Informații;
  - Drumuri.

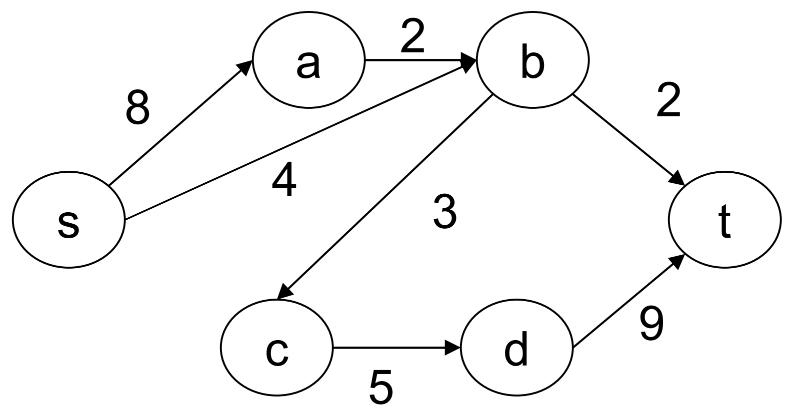




Rețea de flux – Definiție

- G(V,E) graf orientat;
- $c(u,v) \geq 0 \quad \forall(u,v)$ – c = capacitatea arcelor;
- Dacă $(u,v) \notin E \rightarrow c(u,v) = 0$;
- S – sursa traficului;
- T – destinația traficului (drena);
- Presupunem că $\forall u \in V \setminus \{s, t\} \ \exists$ s..u..t.

Exemplu de rețea de flux

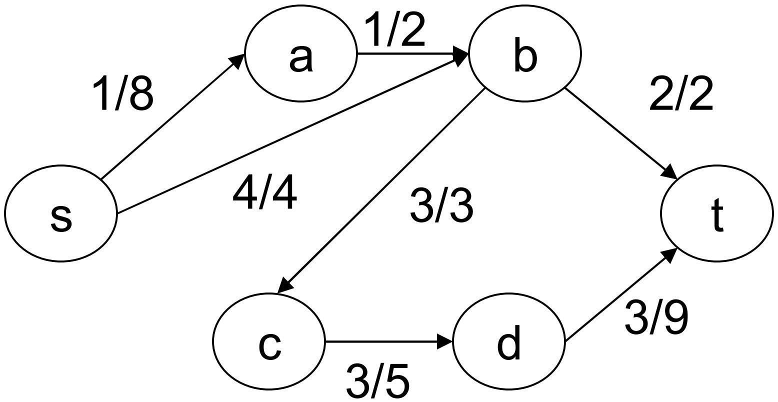


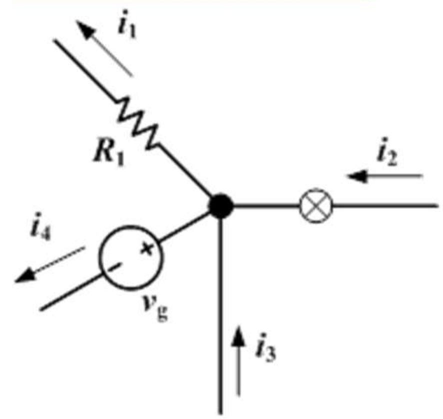
- s – sursa, t – destinația.
- Pe arce este reprezentată capacitatea arcului.

Flux. Definiție. Proprietăți.

- $G = (V,E)$ – rețea de flux;
- $c: V \times V \rightarrow \Re$ - capacitatea rețelei;
- $f: V \times V \rightarrow \Re$ - fluxul prin rețeaua G;
- Proprietăți:
  - $\forall u, v \in V, f(u,v) \leq c(u,v)$ (fluxul printr-un arc este mai mic sau egal cu capacitatea arcului) – respectarea capacității arcelor;
  - $\forall u, v \in V, f(u,v) = -f(v,u)$ – simetria fluxului;
  - $\Sigma f(u,v) = 0$ pentru $\forall u \in V \setminus \{s,t\}$ – conservarea fluxului.

Exemplu de fluxuri





i2+i3-i4-i1=0 (P3)

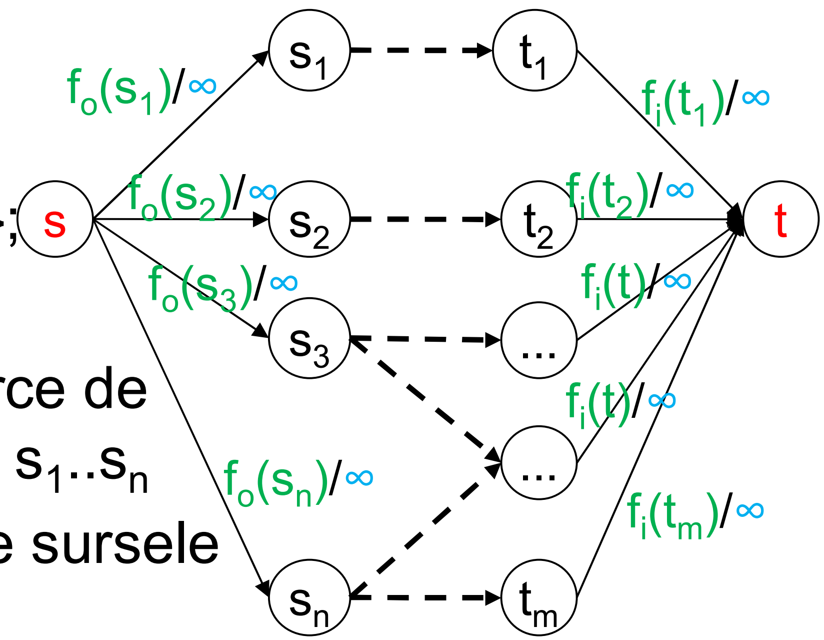
- Σf(u,v) = 0 pentru ∀u ∈ V \ {s,t} – fluxul se conservă;
- Proprietatea 3 = legea curentului (Kirchoff) ☺ - suma I. curenților ce intră într-un nod = suma I. curenților ce ies din nodul respectiv.

Flux. Notații.

- $f(u,v)$ – fluxul din u spre v;
- $f_i(u) = \Sigma f(v,u)$ – fluxul total care intra în nodul u;
- $f_o(u) = \Sigma f(u,v)$ – fluxul total care iese din nodul u;
- Valoarea totală a fluxului:
  - $|f| = \Sigma f(s,v) = f_o(s)$;
  - $|f|$ = fluxul ce părăsește sursa;
  - Cf. proprietăților P1-P3: $|f| = \Sigma f(s,v) = \Sigma f(v,t) = f_i(t)$.

Surse multiple, destinații multiple

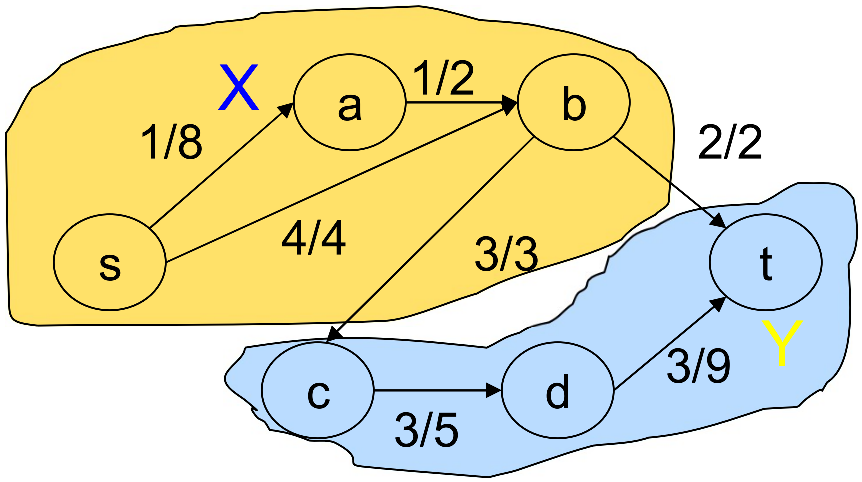
- Surse multiple $\{s_1, s_2, \ldots, s_n\}$;
- Destinații multiple $\{t_1, t_2, \ldots, t_m\}$;
- Se adaugă o sursă unică cu arce de capacitate infinită spre sursele $s_1..s_n$ și flux egal cu fluxul generat de sursele respective;
- Se adaugă o destinație unică t și arce de capacitate infinită între $t_1..t_m$ și t și flux egal cu fluxul ce intră în destinațiile respective.



Operații cu fluxuri

- X,Y – mulțimi de noduri;
- $f(X,Y) = \Sigma_{x\in X}\Sigma_{y\in Y} f(x,y)$ = fluxul între X și Y;
- Operații:
  - $\forall$X *inclus in* V: f(X,X) = 0;
  - $\forall$X,Y *inclus in* V: f(X,Y) = -f(Y,X);
  - $\forall$X,Y,Z *inclus in* V și Y *inclus in* X:
    - $f(X \setminus Y, Z) = f(X,Z) - f(Y,Z)$;
    - $f(Z, X \setminus Y) = f(Z,X) - f(Z,Y)$;
  - $\forall$X,Y,Z *inclus in* V și $X\cap Y=\emptyset$:
    - $f(X \cup Y, Z) = f(X,Z) + f(Y,Z)$;
    - $f(Z, X \cup Y) = f(Z,X) + f(Z,Y)$
  - f(s,V) = f(V,t)

Exemplu operații fluxuri (1)

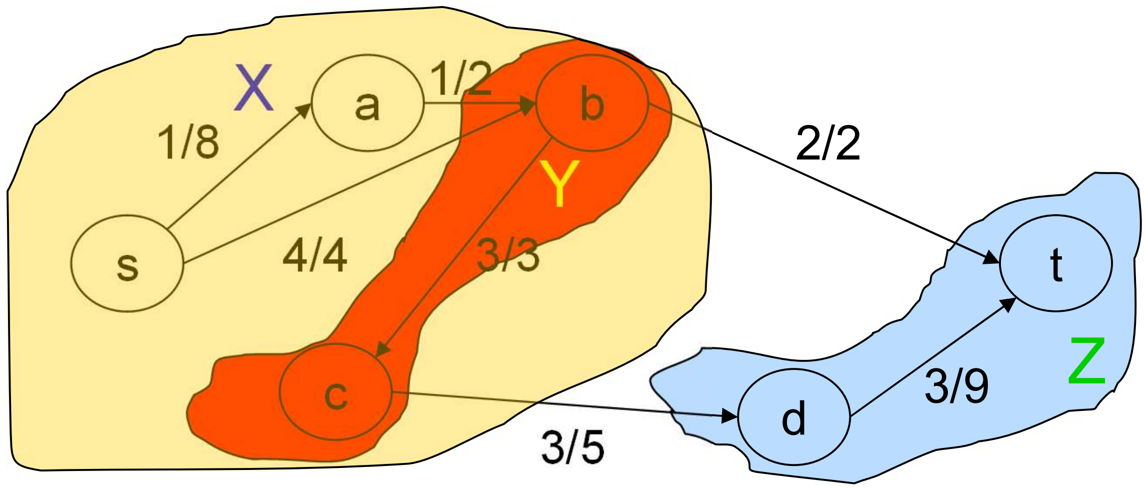


$$f(X,Y) = \Sigma_{x \in X} \Sigma_{y \in Y} f(x,y)$$

$f(X,X) = f(s,a) + f(a,s) + f(s,b) + f(b,s) + f(a,b) + f(b,a) = 0$

$f(X,Y) = f(b,c) + f(b,t) = -f(c,b) - f(t,b) = -f(Y,X)$

Exemplu operații fluxuri (2)



∀X, Y, Z *inclus in* V si Y inclus in X

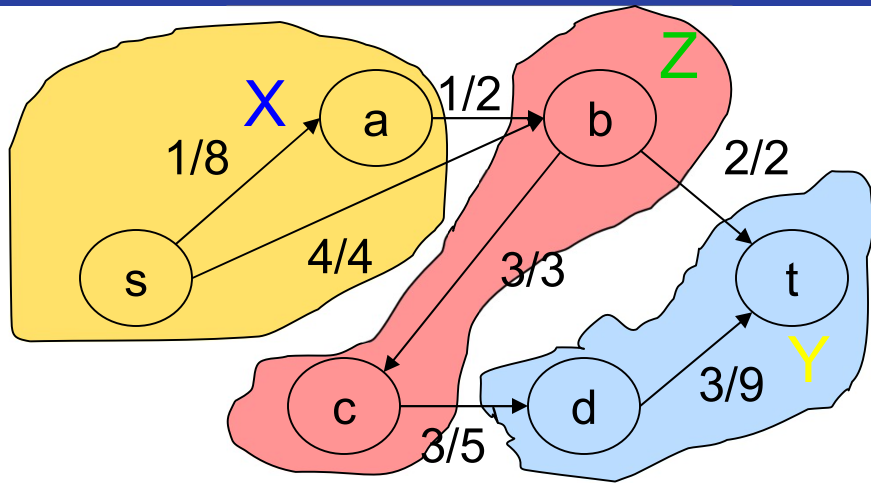
f(X \ Y, Z) = f(X, Z) - f(Y, Z)

f(Z, X \ Y) = f(Z, X) - f(Z, Y)

f(X \ Y, Z) = 0 = f(b,t) + f(c,d) - f(b,t) - f(c,d) = f(X,Z) - f(Y,Z)

f(Z, X \ Y) = 0 = f(t,b) + f(d,c) - f(t,b) - f(d,c) = f(Z,X) - f(Z,Y)

Exemplu operații fluxuri (3)



$\forall$X, Y, Z *inclus in* V si $X \cap Y = \varnothing$

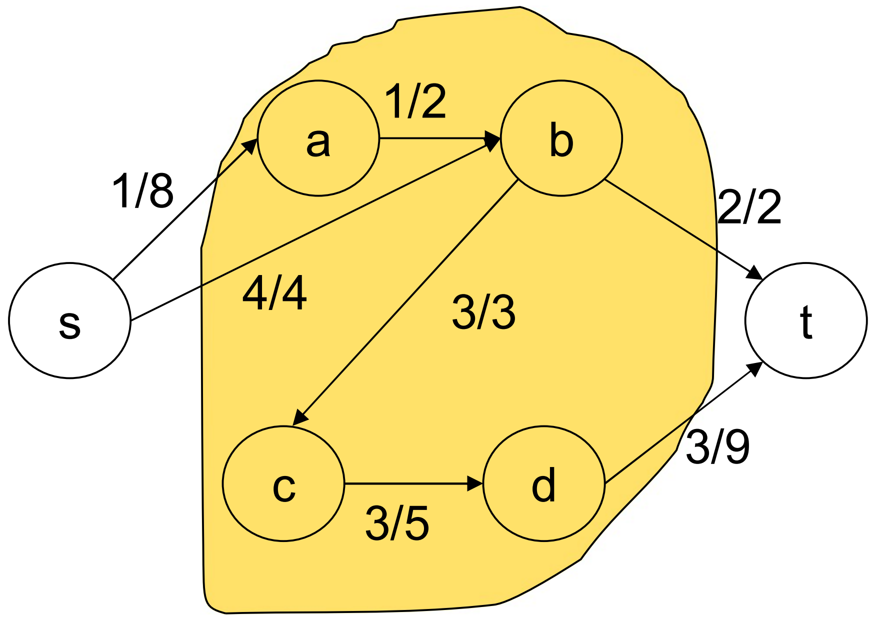
$f(X \cup Y, Z) = f(X,Z) + f(Y,Z)$

$f(Z, X \cup Y) = f(Z,X) + f(Z,Y)$

$f(X \cup Y, Z) = f(s,b) + f(a,b) + f(t,b) + f(d,c) = f(X,Z) + f(Y,Z)$

$f(Z, X \cup Y) = f(b,a) + f(b,s) + f(b,t) + f(c,d) = f(Z,X) + f(Z,Y)$

Exemplu operații fluxuri (4)



f(s, V) = f(V, t)

f(s, V) = f(s,a) + f(s,b) = 5 = f(d,t) + f(b,t) = f(V, t)

Arc rezidual. Capacitate reziduală.

- Definiție: Un arc (u,v) pentru care $f(u,v) < c(u,v)$ se numește arc rezidual.
- ➔Fluxul pe acest arc se poate mări.
- Definiție: Cantitatea cu care se poate mări fluxul pe arcul (u,v) se numește capacitatea reziduală a arcului (u,v) ($c_f(u,v)$):

$$c_f(u,v) = c(u,v) - f(u,v)$$

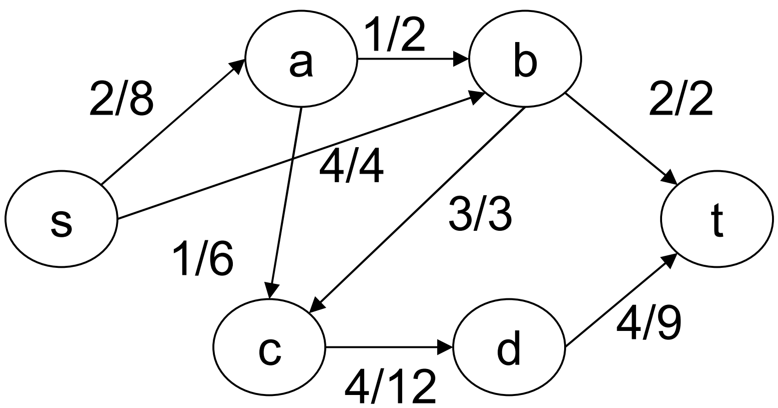
Rețea reziduală. Cale reziduală.

- G = (V,E) rețea de flux cu funcția de capacitate c.
- Definiție: Rețeaua reziduală (G_f = (V,E_f)) este o rețea de flux formată din arcele ce admit creșterea fluxului:

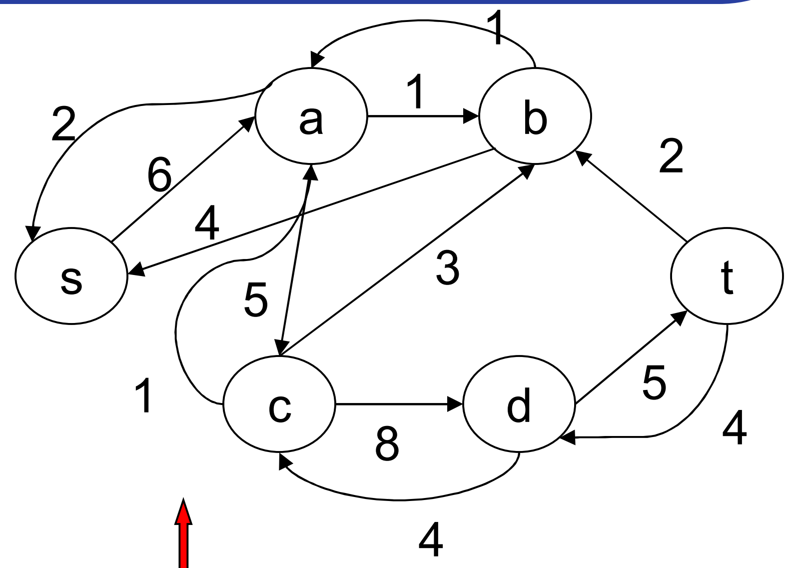
$$E_f = \{(u,v) \in V \times V \mid c_f(u,v) > 0\}.$$

- Observație: $E_f \not\subseteq E$!!!
- Definiție: O cale reziduală este un drum s..t din G_f.
- Definiție: Capacitatea reziduală a căii = capacitatea reziduală minimă de pe calea s..t descoperită.

Exemplu rețea reziduală

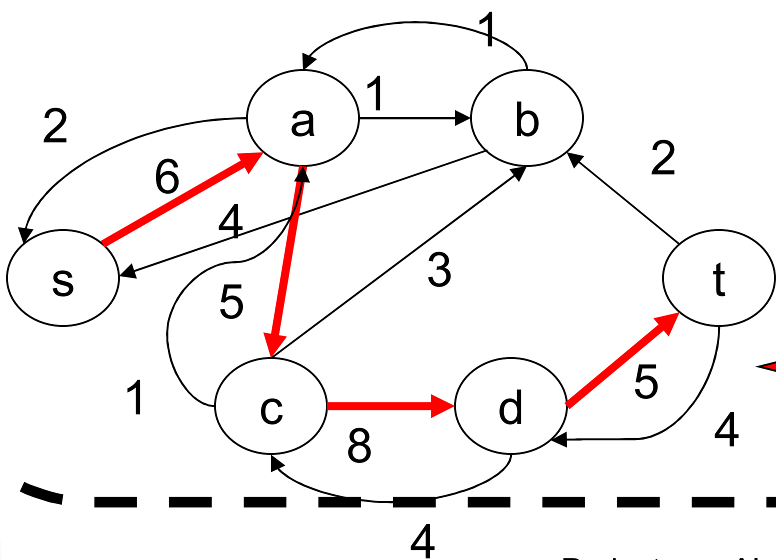


$c_f(u,v) = c(u,v) - f(u,v)$



Rețeaua reziduală $G_f = (V,E_f)$ unde

$E_f = \{(u,v) \in V \times V \mid c_f(u,v) > 0\}$



Calea reziduală: s → a → c → d → t

Capacitatea reziduală a căii:

$c_f(p) = \min\{6, 5, 8, 5\} = 5$

Rețea reziduală

- Lemă 5.16: Fie G = (V,E) rețea de flux, f fluxul în G, G_f rețeaua reziduală a lui G. Fie f' un flux prin G_f și f+f' o funcție definită astfel:

$$f+f'(u,v) = f(u,v) + f'(u,v).$$

- Atunci f+f' reprezintă un flux în G și

$$|f+f'| = |f| + |f'|$$

- Această Lemă ne spune cum putem mări fluxul printr-o rețea de flux.

Flux în rețeaua reziduală

- Lemă 5.17: G – rețea de flux, f flux în G, p = s..t – cale reziduală în G_f, f_p:V x V->$\Re$ se definește ca fiind:

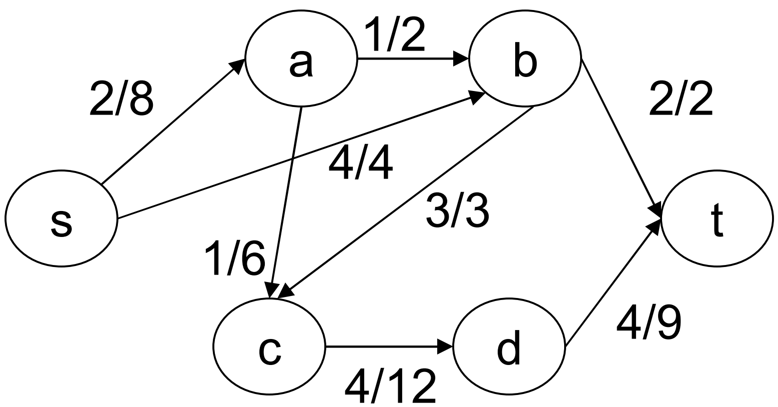
$$f_p(u,v) = \begin{cases} c_f(p), \text{ dacă } (u,v) \in p \\ -c_f(p), \text{ dacă } (v,u) \in p \\ 0, \text{ dacă } (u,v) \text{ și } (v,u) \notin p \end{cases}$$

Atunci f_p = flux în G_f; $|f_p| = c_f(p)$

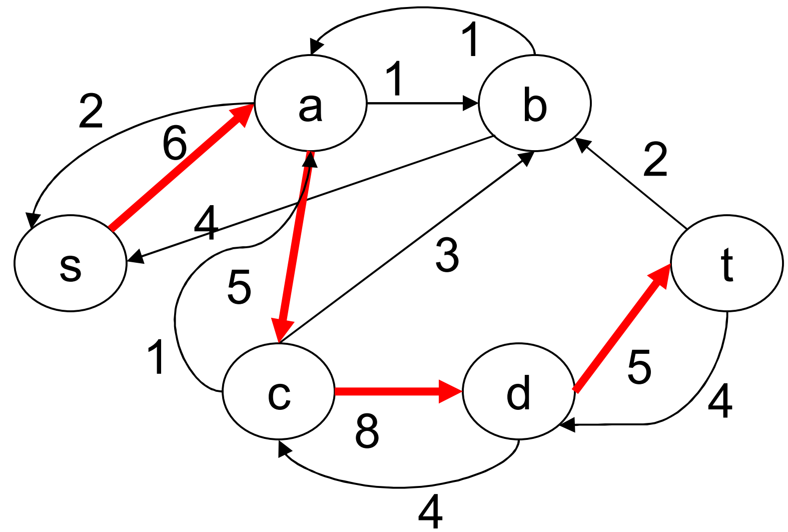
- Corolar 5.4: f' = f + f_p = flux în G, astfel încât $|f'| = |f| + |f_p| > |f|$.

- Această Lemă ne spune cum se definește fluxul printr-o rețea reziduală.

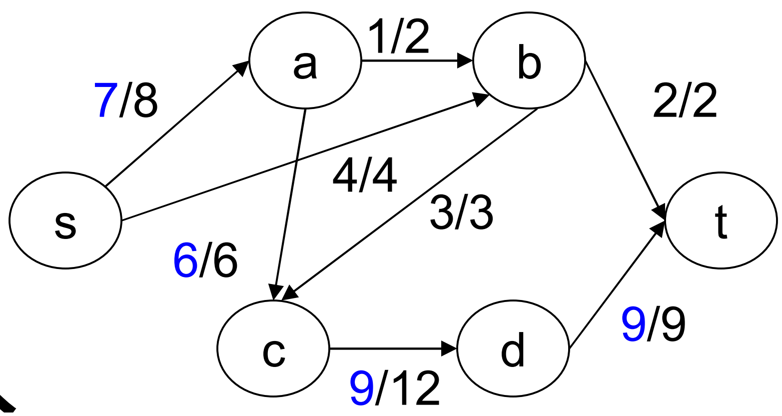
Exemplu maximizare flux



$|f(s..t)| = 2 + 4 = 6$



$|f_p(s..t)| = c_f(s..t) = 5$



$|f'(s..t)| = |f| + |f_p| = 6 + 5 = 11$

Calculul fluxului maxim

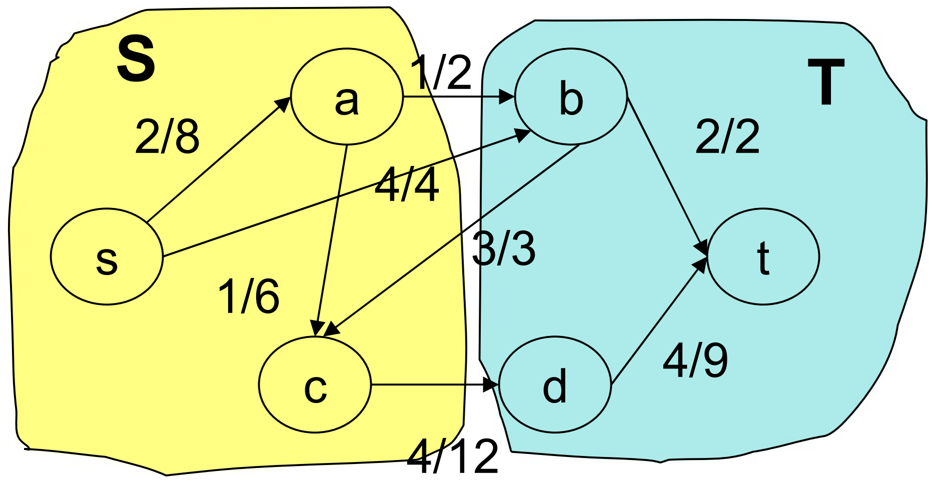
- Metoda Ford-Fulkerson
  - $f(u,v) = 0 \ \forall \ (u,v)$ // inițializarea fluxului
  - **Repetă** // creștere iterativă a fluxului
    - găsește un drum s..p..t pe care se poate mări fluxul (cale reziduală)
    - f = f + flux(s..p..t)
  - **Până când** nu se mai poate găsi nici un drum s..p..t
  - **Întoarce** f

- În funcție de metodele de identificare a căii reziduale există mai mulți algoritmi ce urmează această metodă.

Tăieturi în rețele de flux

- Definiție: O tăietură (S,T) a unei rețele de flux G = partiționare a nodurilor în 2 mulțimi disjuncte S și T = V \ S astfel încât $s \in S$ și $t \in T$.
  - $f(S,T) = \Sigma_{x \in S}\Sigma_{y \in T} f(x,y)$ – fluxul prin tăietura
  - $c(S,T) = \Sigma_{x \in S}\Sigma_{y \in T} c(x,y)$ – capacitatea tăieturii
- Lema 5.18: Fluxul prin tăietură = fluxul prin rețea – $f(S,T) = |f|$
- Corolar 5.5: (S, T) – tăietură oarecare – fluxul maxim este limitat superior de capacitatea tăieturii $|f| \le c(S,T)$

Exemplu de tăietură într-o rețea de flux



- $f(S,T) = f(s,b) + f(a,b) + f(c,d) + f(c,b) = 4 + 1 + 4 - 3 = 6 = f(s,V)$

- $c(S,T) = c(a,b) + c(s,b) + c(c,d) = 18$

Flux maxim – tăietură minimă

- Teorema 5.25 (Flux maxim – tăietură minimă): G = (V,E) rețea de flux – următoarele afirmații sunt echivalente:
  - f este o funcție de flux în G astfel încât |f| este flux maxim total în G;
  - rețeaua reziduală G_f nu are căi reziduale;
  - există o tăietură (S,T) astfel încât |f| = c(S,T).

Algoritmul Ford – Fulkerson

- Ford – Fulkerson(G,s,t)
  - **Pentru fiecare** (u,v) din E
    - f(u,v) = f(v,u) = 0 // inițializare
  - **Cât timp**
    - Există o cale reziduală p între s..t în G_f
      - $c_f(p) = \min\{c_f(u,v) \mid (u,v) \text{ din } p\}$ // capacitatea reziduală
      - **Pentru fiecare** (u,v) din p
        - $f(u,v) = f(u,v) + c_f(p)$
        - $f(v,u) = -f(u,v)$
  - **Întoarce** |f|

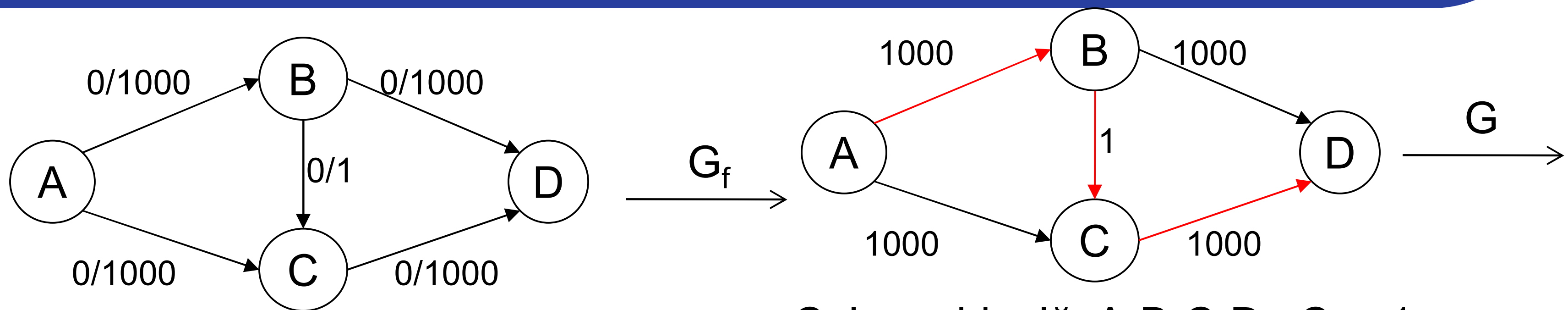
Complexitate?

Algoritmul Ford – Fulkerson (2)

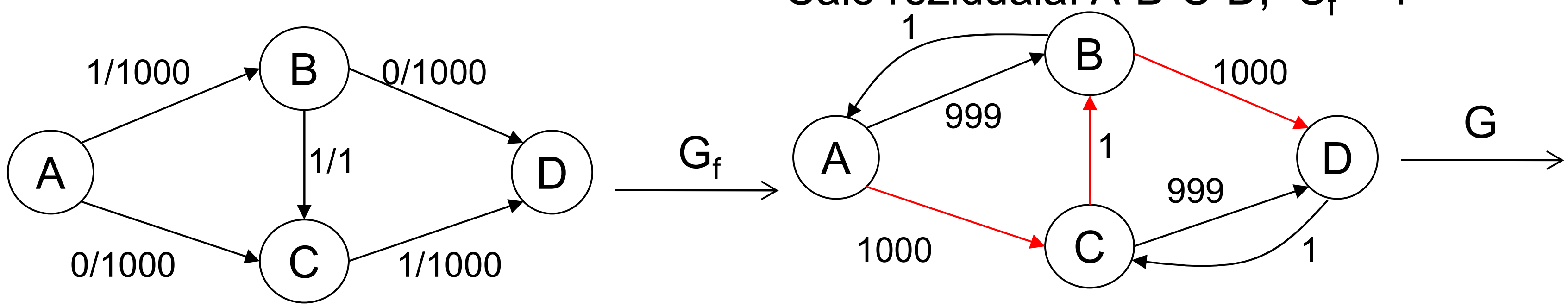
- Ford – Fulkerson(G,s,t)
  - **Pentru fiecare** (u,v) din E
    - f(u,v) = f(v,u) = 0 // O(E)
  - **Cât timp** // O(?)
    - Există o cale reziduală p între s..t în G_f // O(E)
      - $c_f(p) = \min\{c_f(u,v) \mid (u,v) \text{ din } p\}$ // O(E)
      - **Pentru fiecare** (u,v) din p // O(E)
        - f(u,v) = f(u,v) + $c_f(p)$
        - f(v,u) = -f(u,v)
  - **Întoarce** |f|

Complexitate?

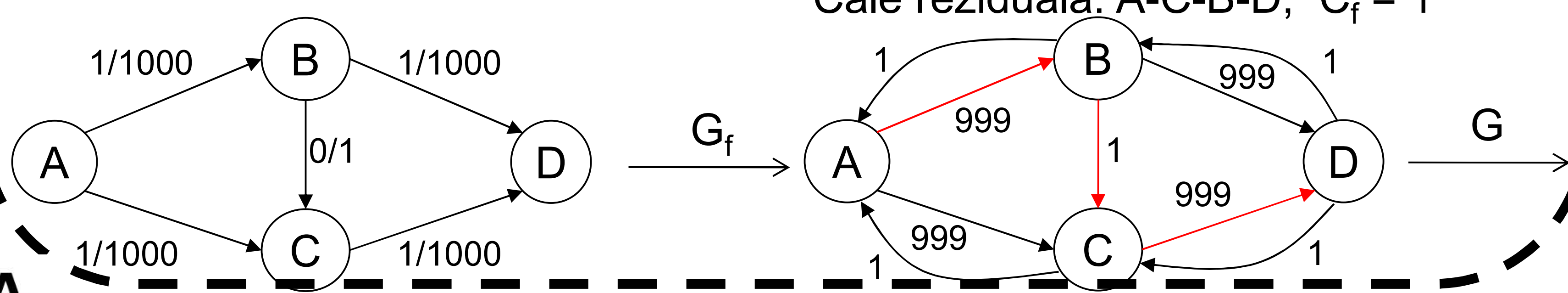
Exemplu Ford – Fulkerson (1)



Cale reziduală: A-B-C-D; C_f = 1

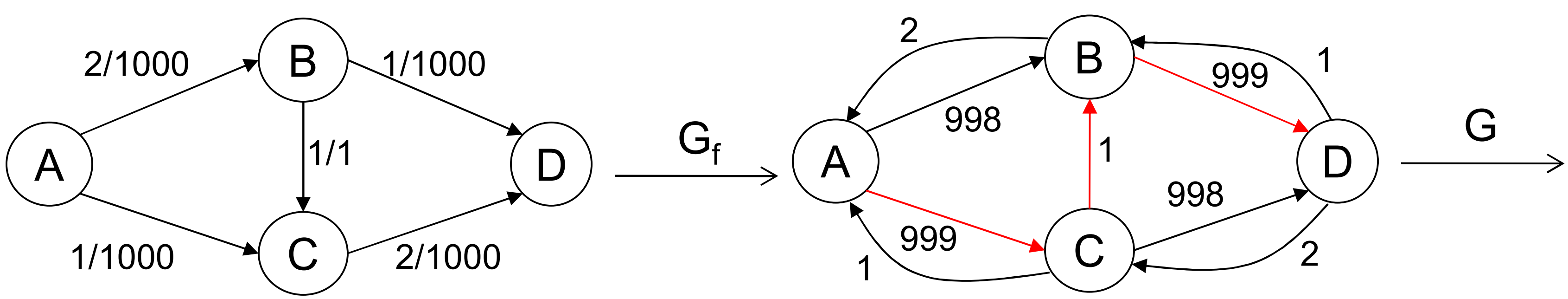


Cale reziduală: A-C-B-D; C_f = 1

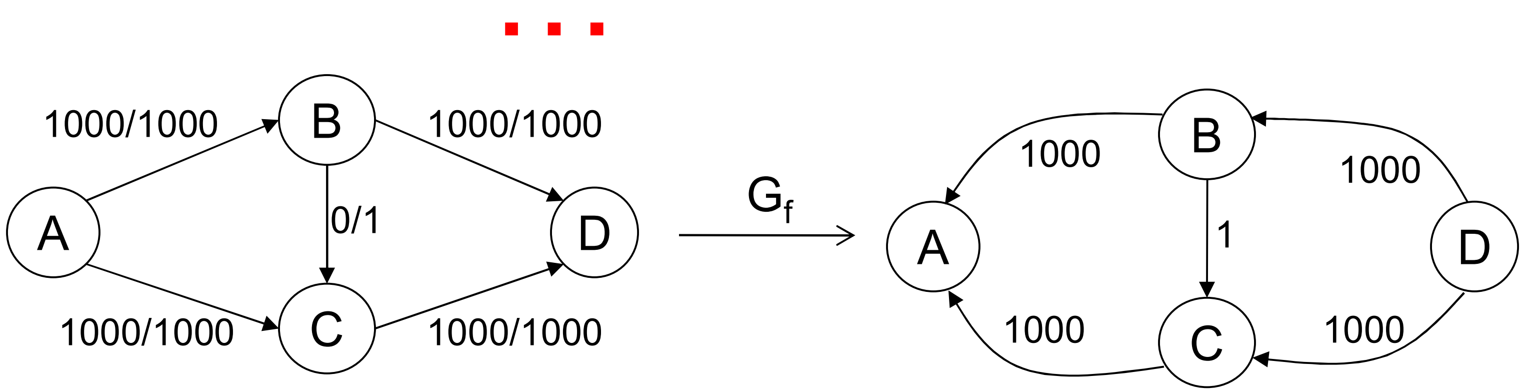


Cale reziduală: A-B-C-D; C_f = 1

Exemplu Ford – Fulkerson (2)



Cale reziduală: A-C-B-D; C_f = 1



Cale reziduală:Ø

După câți pași se ajunge la forma finală?

Complexitate Ford – Fulkerson

- Complexitate $O(E * f_{max})$
- f_{max} = fluxul maxim

Algoritmul Ford – Fulkerson – discuție

- Probleme ce pot să apară:
  - Se folosesc căi cu capacitate mică;
  - Se pun fluxuri pe mai multe arce decât este nevoie.

- Îmbunătățiri:
  - Se aleg căile reziduale cu capacitate maximă – complexitatea va depinde în continuare de f_{max} și de valoarea capacităților;
  - Se aleg căile reziduale cele mai scurte → în acest caz complexitatea nu mai depinde de f_{max} ci numai de numărul de arce (ex. Edmonds-Karp: identificarea căilor reziduale minime prin aplicarea unui BFS)

Algoritmul Edmonds – Karp (1)

- Edmonds – Karp(G, s, t)
  - **Pentru fiecare** (u,v) din E
    - f(u,v) = f(v,u) = 0 // inițializare
  - **Cât timp**
    - Există căi reziduale între s..t în G_f
      - Determină calea reziduală minimă p aplicând BFS
      - $c_f(p) = \min\{c_f(u,v) \mid (u,v) \text{ din } p\}$ // capacitatea reziduală
      - **Pentru fiecare** (u,v) din p
        - $f(u,v) = f(u,v) + c_f(p)$
        - $f(v,u) = -f(u,v)$
  - **Întoarce** |f|

Complexitate?

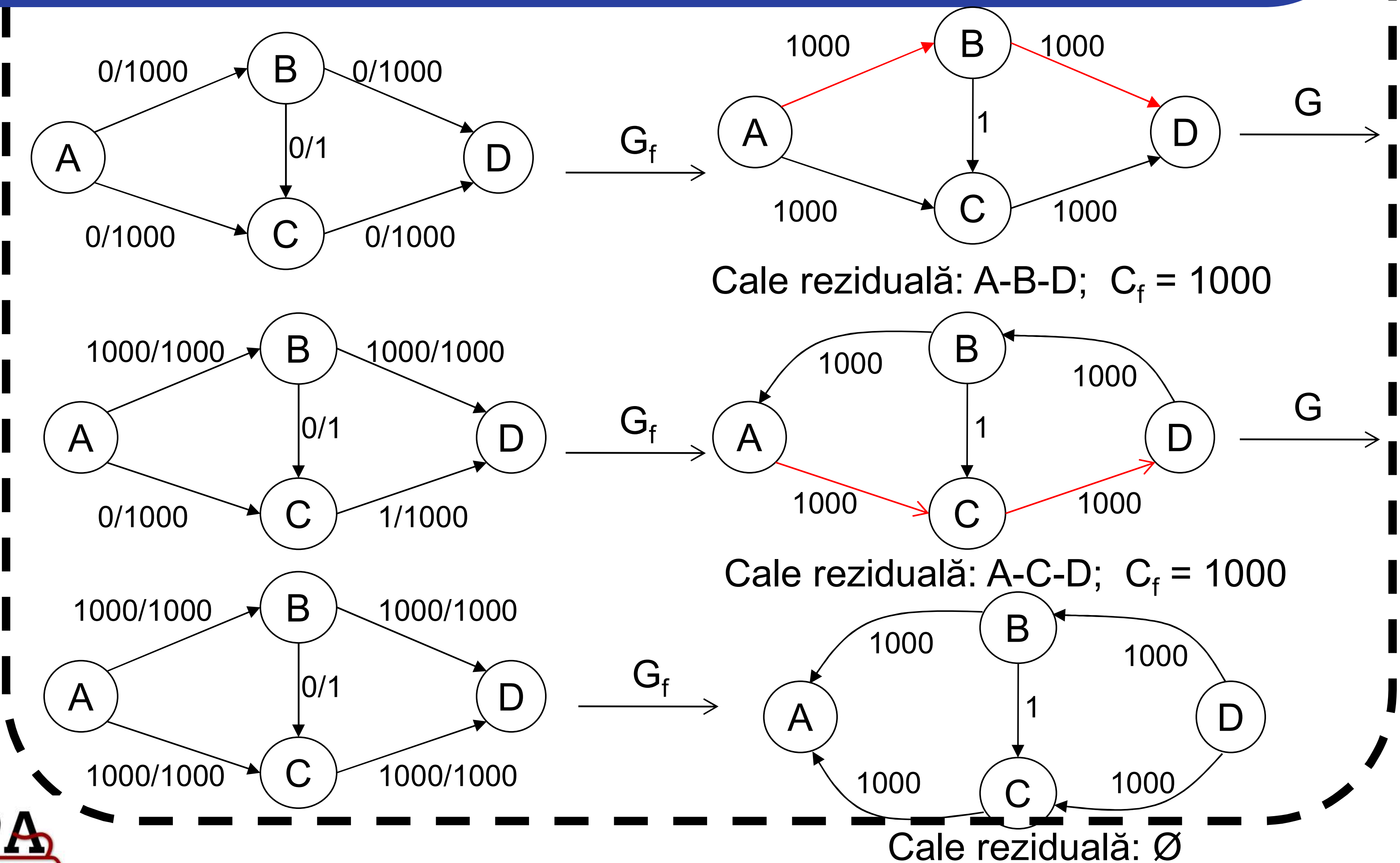
Algoritmul Edmonds – Karp (2)

- Edmonds – Karp(G, s, t)
  - **Pentru fiecare** (u,v) din E
    - f(u,v) = f(v,u) = 0 // O(E)
  - **Cât timp** // O(E*V) [vezi Cormen]
    - Există căi reziduale între s..t în G_f // O(E)
      - Determină calea reziduală minimă p aplicând BFS // O(E)
      - $c_f(p) = \min\{c_f(u,v) \mid (u,v) \text{ din } p\}$ // O(E)
      - **Pentru fiecare** (u,v) din p // O(E)
        - $f(u,v) = f(u,v) + c_f(p)$
        - $f(v,u) = -f(u,v)$
  - **Întoarce** |f|

De câte ori un arc poate fi critic în rețeaua reziduală? O(V)
Câte arce? O(E)

Complexitate?
$O(E^2 * V)$

Exemplu Edmonds-Karp



Pompare preflux (1)

- Idee: Simularea curgerii lichidelor într-un sistem de conducte ce leagă noduri aflate la diverse înălțimi;
- Sursa – înălțime maximă;
- Inițial toate nodurile exceptând sursa sunt la înălțime 0;
- Destinația rămâne în permanență la înălțimea 0!

Pompare preflux (2)

- Există un preflux inițial în rețea obținut prin încărcarea la capacitate maximă a tuturor conductelor ce pleacă din s;
- Excesul de flux dintr-un nod poate fi stocat într-un rezervor al nodului (Notat e(u));
- Când un nod u are flux disponibil în rezervor și o conductă spre un alt nod v nu este încărcată complet → înălțimea lui u este crescută pentru a permite curgerea din u în v.

Pompare preflux – Definiții (1)

- G = (V,E) rețea de flux;
- Definiție: Preflux = f: V x V → $\Re$ astfel încât să fie satisfăcute restricțiile:
  - $f(u,v) \leq c(u,v), \forall(u,v) \in E$ – respectarea capacității arcelor;
  - $f(u,v) = -f(v,u), \forall u,v \in V$ – simetria fluxului;
  - $\Sigma_{v \in V} f(v,u) \geq 0, \forall u \in V \setminus \{s\}$ – ~~conservarea fluxului.~~
- Definiție: Supraîncărcare a unui nod:
  - $e(u) = f(V,u) \geq 0 , \forall u \in V \setminus \{s\}$.

Pompare preflux – Definiții (2)

- Definiție: O funcție h: V → N este o funcție de înălțime dacă îndeplinește restricțiile:
  - $h(s) = |V|$ – fixă;
  - $h(t) = 0$ – fixă;
  - $h(u) \leq h(v) + 1$ pentru orice arc rezidual $(u,v) \in G_f$ – variabilă.

- Lema 5.19: G – rețea de flux, h: V → N este o funcție de înălțime. Fie $u, v \in V$; dacă $h(u)>h(v)+1$ atunci arcul (u,v) nu este arc rezidual.

Pompare preflux – Metode folosite

- Pompare(u,v) // pompează fluxul în exces (e(u) > 0)
  // are loc doar dacă diferența de înălțime dintre u și v este 1
  // (h(u) = h(v) + 1), altfel nu e arc rezidual și nu ne interesează
  - d = min(e(u), c_f(u,v)); // cantitatea de flux pompată
  - f(u,v) = f(u,v) + d; // actualizare flux pe arcul (u,v)
  - f(v,u) = -f(u,v); // respectarea simetriei
  - e(u) = e(u) – d; // actualizare supraîncărcare la sursă
  - e(v) = e(v) + d; // actualizare supraîncărcare la destinație

- Înălțare(u) // mărește h(u) dacă u are flux în exces
  // (e(u) > 0) și u $\notin$ {s, t} $\forall$(u,v) $\in$ G_f avem h(u) ≤ h(v)
  - h(u) = 1 + min{h(v) | (u,v) $\in$ G_f}

Complexitate?

Pompare preflux – Inițializare

- Init_preflux(G, s, t)
  - **Pentru fiecare** (u ∈ V)
    - e(u) = 0 // inițializare exces flux în nodul u
    - h(u) = 0 // inițializare înălțime nod u
    - **Pentru fiecare** (v ∈ V) // inițializare fluxuri
      - f(u,v) = 0
      - f(v,u) = 0
  - h(s) = |V| // inițializare înălțime sursă
  - **Pentru fiecare** (u ∈ succs(s) \ {s})
    // actualizare flux + exces
    - f(s,u) = c(s,u);
    - f(u,s) = -c(s,u);
    - e(u) = c(s,u)

Complexitate?

Pompare preflux – Algoritm

- Pompare_preflux(G, s, t)
  - Init_preflux(G, s, t) // inițializarea prefluxului
  - **Cât timp** (1) // cât timp pot face pompări sau înălțări
    - **Dacă** ($\exists u \in V \setminus \{s, t\}$ | $e(u) > 0$ **și** $c_f(u,v) > 0$ **și** $h(u) = h(v) + 1$) // încerc să pompez
      - Pompare(u,v); ***continuă;***
    - **Dacă** ($\exists u \in V \setminus \{s, t\}, v \in V$ | $e(u) > 0$ **și** $\forall (u,v) \in E_f, h(u) \leq h(v)$)
      - Înălțare(u); ***continuă;*** // încerc să înalț
    - **Întrerupe;** // nu mai pot face nimic → am ajuns la flux max
  - **Întoarce** e(t) // e(t) = |f| = fluxul total în rețea

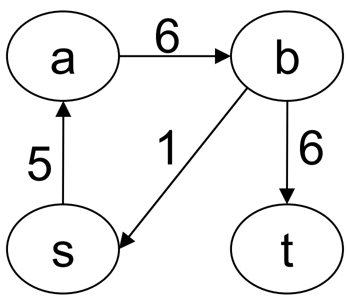
- Complexitate?

Pompare preflux – Complexitate

- Init_preflux: O(V * E)
- Pompare(u,v): O(1)
- Înălțare(u): O(V) – implică găsirea minimului dintre nodurile succesoare
- Cât timp: [vezi Cormen]
  - Câte înălțări?
    - Care e înălțimea maximă? 2 |V| - 1 – drum rezidual de lungime maximă
    - Care este numărul maxim total de înălțări? (2 |V| - 1) (|V| - 2)
  - Câte pompări?
    - Pompări saturate: 2 |V| |E| - de câte ori un arc poate fi saturat? (în funcție de suma h(u) + h(v))
    - Pompări nesaturate: 4 $|V|^2$ (|V| + |E|) – sumă înălțimi noduri excedentare
- Complexitate totală: O(V^2 * E) [vezi Cormen]

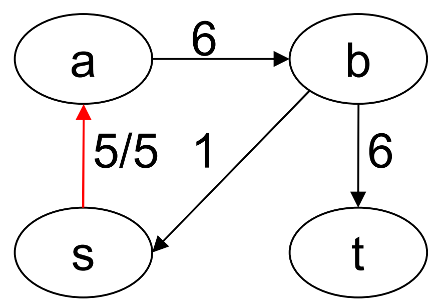
Exemplu Pompare preflux (1)

G

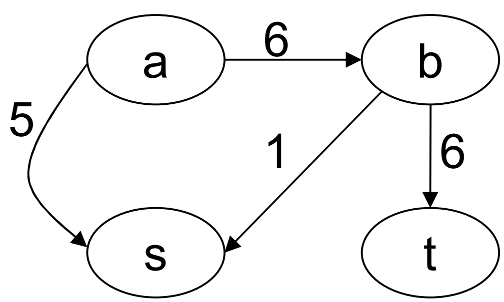


Init_preflux →

$h(s) = 4$
$h(a) = h(b) = h(t) = 0$
$e(a) = 5$
$e(s) = e(b) = e(t) = 0$



$\xrightarrow{G_f}$



G_f

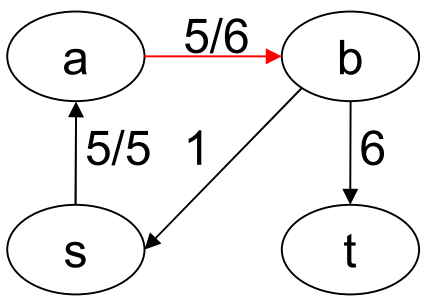
Înălțare (a) →

$h(s) = 4$
$h(a) = 1$
$h(b) = h(t) = 0$
$e(a) = 5$
$e(s) = e(b) = e(t) = 0$

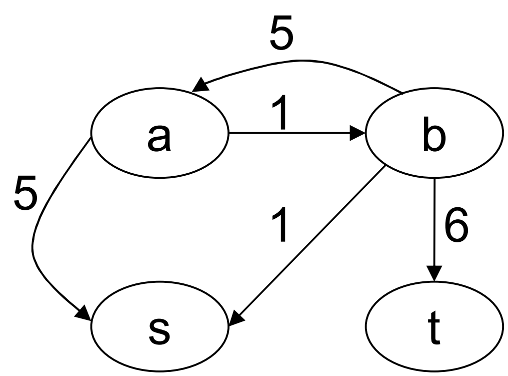
Pompare (a,b) →

Exemplu Pompare preflux (2)

h(s) = 4
h(a) = 1
h(b) = h(t) = 0
e(b) = 5
e(s) = e(a) = e(t) = 0



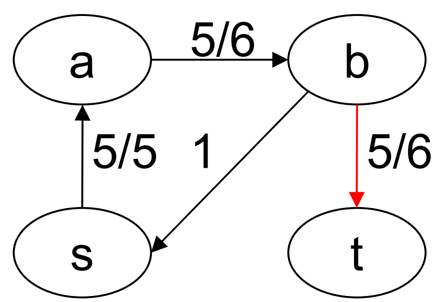
$\xrightarrow{G_f}$



$\xrightarrow[\text{(b)}]{\text{Înălțare}}$

h(s) = 4
h(a) = h(b) = 1
h(t) = 0
e(b) = 5
e(s) = e(a) = e(t) = 0

$\xrightarrow[\text{(b,t)}]{\text{Pompare}}$



h(s) = 4
h(a) = h(b) = 1
h(t) = 0
e(t) = 5
e(s) = e(a) =
= e(b) = 0

ÎNTREBĂRI?