Curs 9

Conectivitate: algoritmul lui Dijkstra.
Rețele de transport: algoritmi de flux maxim

Conținutul cursului

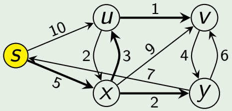
1. Problema celor mai ușoare căi de la un nod sursă într-un digraf ponderat
    - **Algoritmul lui Dijkstra**
2. Rețele de transport și fluxuri
    - Flux maxim
    - Rețele reziduale, drumuri de creștere
    - **Algoritmul Ford-Fulkerson**
    - Aplicații

Drumuri minime de la un nod sursă precizat

Se dă un digraf ponderat simplu $G = (V, E)$ cu $w : E \mapsto \mathbb{R}^+$ și un nod sursă $s \in V$

Se dorește pentru fiecare nod $x \in V$ accesibil din s, o cea mea ușoară cale $\rho : s \rightsquigarrow x$, precum și greutatea ei $w(\rho)$

Exemplu



$[s]$ cu $w([s]) = 0$

$[s, x]$ cu $w([s, x]) = 5$

$[s, x, y]$ cu $w([s, x, y]) = 7$

$[s, x, u]$ cu $w([s, x, u]) = 8$

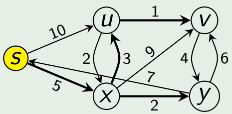
$[s, x, u, v]$ cu $w([s, x, u, v]) = 9$

Drumuri minime de la un nod sursă precizat

Se dă un digraf ponderat simplu $G = (V, E)$ cu $w : E \mapsto \mathbb{R}^+$ și un nod sursă $s \in V$

Se dorește pentru fiecare nod $x \in V$ accesibil din s, o cea mea ușoară cale $\rho : s \rightsquigarrow x$, precum și greutatea ei $w(\rho)$

Exemplu



$[s]$ cu $w([s]) = 0$

$[s, x]$ cu $w([s, x]) = 5$

$[s, x, y]$ cu $w([s, x, y]) = 7$

$[s, x, u]$ cu $w([s, x, u]) = 8$

$[s, x, u, v]$ cu $w([s, x, u, v]) = 9$

Observație

- Problema poate fi rezolvată cu algoritmul lui Warshall:
  - Calculează cele mai ușoare căi existente pentru orice pereche de noduri
  - Are complexitatea $O(|V|^3)$ și calculează prea mult

Există un algoritm mai eficient, dacă nodul sursă este fixat?

Propus de E. Dijkstra în 1956 pentru a rezolva problema anterioară

1. Se atribuie
   - o greutate tentativă $d(x)$ pentru calea cea mai ușoară la fiecare nod x.
   - un nod precedent $\pi(x)$ fiecărui nod x pe calea cea mai ușoară de la s la x.

   Inițial avem $d(x) = \begin{cases} 0 & \text{dacă } x = s, \\ \infty & \text{dacă } x \neq s \end{cases}$ $\quad \pi(x) = \begin{cases} \textit{nedef} & \text{dacă } x = s \\ s & \text{dacă } x \neq s \end{cases}$
   unde *nedef* este o valoare specială care indică inexistența unui nod părinte.
2. Crează o mulțime Q de noduri nevizitate. Inițial, $Q := V$, și ține evidența unui nod curent crt.
3. Alege crt :=nod din Q cu $d(crt) = \min\{d(x) \mid x \in Q\}$, și elimină crt din Q.
4. Pentru fiecare vecin $x \in Q$ al lui crt actualizează valorile tenative ale lui $d(x)$ și $\pi(x)$ astfel:

   Dacă $d(crt) + w((crt, x)) < d(x)$ atunci
   $d(x) := d(crt) + w((crt, x))$ și $\pi(x) := crt$.

   Acest pas de actualizare se numește pas de relaxare a arcului $(crt, x) \in E$.
5. Dacă $Q = \emptyset$ atunci **stop**, altfel **goto 3**.

- Inițializare

  INITOSURSA(G, s)

  pentru fiecare $v \in V$

  $d(v) := \infty$

  $\pi(v) := s$

  $d(s) := 0$

  $\pi(s) := null$

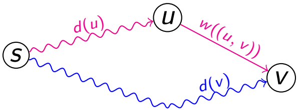
- Pasul de relaxare a unui arc (u, v)

  RELAXEAZA(u, v)

  Dacă $d(v) > d(u) + w((u, v))$

  $d(v) := d(u) + v((u, v))$

  $\pi(v) := \pi(u)$



Algoritmul lui Dijkstra
Pseudocod

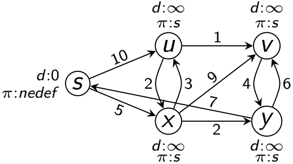
```
DIJKSTRA(G, w, s)
InitOSursa(G, s)
Q := V
while Q ≠ ∅
    u := ExtractMin(Q)
    for fiecare vecin v al lui u pentru care v ∉ Q
        Relaxeaza(u, v)
```

Complexitate temporală:

- Algoritmul original: $O(|V|^2)$
- Algoritmul îmbunătățit cu o coadă cu prioritate minimă: $O(|E| + |V| \cdot \log |V|)$

Algoritmul lui Dijkstra
Exemplu ilustrat: prima buclă **while**

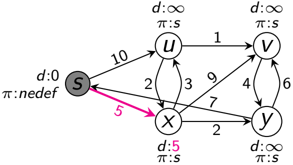
CONVENŢIE: Nodurile nemarcate încă (cele din Q) sunt albe; celelalte sunt cenușii

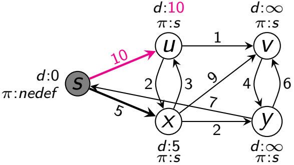


Configurația produsă de INITIALIZESINGLESOURCE(G, s):

$Q = \{s, x, y, u, v\}$
Se selectează $s = \text{EXTRACTMIN}(Q)$

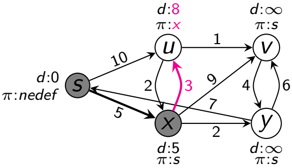
Se relaxează toate arcele care pleacă din s spre noduri nevizitate încă:

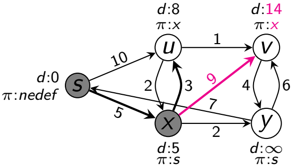


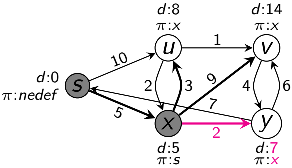


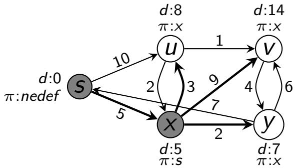
Algoritmul lui Dijkstra
Exemplu ilustrat: a doua buclă **while**

Se selectează și marchează x, și se relaxează toate arcele care pleacă din x spre noduri nemarcate:

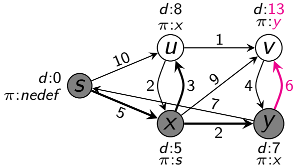


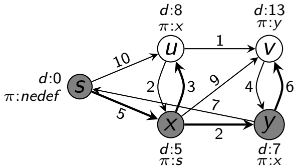




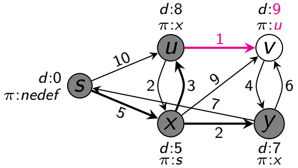


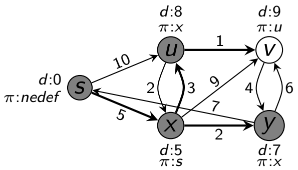
Se selectează și marchează y, și se relaxează toate arcele care pleacă din y spre noduri nemarcate:





Se selectează și marchează u, și se relaxează toate arcele care pleacă din u spre noduri nemarcate:





$d(s) = 0$ $\quad \pi(s) = nedef$
$d(x) = 5$ $\quad \pi(x) = s$
$d(u) = 8$ $\quad \pi(u) = x$
$d(y) = 7$ $\quad \pi(y) = x$
$d(v) = 9$ $\quad \pi(v) = u$

- Se selectează și marchează v
- N-a mai rămas de relaxat nici un arc $\Rightarrow$ algoritmul se oprește.

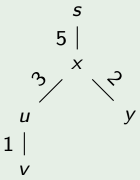
Din valorile lui π și d putem extrage căile cele mai ușoare:

- la s: $[s]$ cu greutatea $w([s]) = d(s) = 0$
- la x: $[s, x]$ cu greutatea $w([s, x]) = d(x) = 5$
- la u: $[s, x, u]$ cu greutatea $w([s, x, u]) = d(u) = 8$
- la y: $[s, x, y]$ cu greutatea $w([s, x, y]) = d(y) = 7$
- la v: $[s.x, u, v]$ cu greutatea $w([s, x, u, v]) = d(v) = 9$

Arborele celor mai ușoare căi de la un nod sursă

Funcția π calculată de algoritmul lui Dijkstra determină un arbore G_π cu rădăcina s, în care fiecare nod $x \neq s$ are părintele $\pi(x)$.

Exemplu (Arborele G_π pentru digraful simplu ponderat G ilustrat)



Observație

Orice ramură a lui G_π de la nodul sursă s la un nod x este o cale cea mai ușoară de la s la x.

Referințe

1. T. H. Cormen, C. E. Leiserson, R. L. Rivest. Secțiunea **25**.2 din *Introduction to Algorithms*. MIT Press, 2000.
2. O implementare în `C++` a algoritmului lui Dijkstra poate fi descărcată de pe pagina web a cursului (click aici)

Rețea de transport: Graf orientat în care muchiile modelează fluxuri de materiale între noduri (litri de lichid, amperi de electricitate, etc.)

- Fiecare muchie are o capacitate maximă.
- Se dorește stabilirea unui flux de resurse de la un nod sursă (**producătorul**) la un nod destinație (**consumatorul**).

Flux $\approx$ debitul de deplasare a resurselor de-a lungul muchiilor.

Problema debitului maxim: Care este debitul maxim al unui flux de resurse de la sursă la destinație, fără să se violeze nici o constrângere de capacitate maximă ale muchiilor?

Definiție (Rețea de transport)

Un graf orientat $G = (V, E)$ în care fiecare muchie $(u, v) \in E$ are o capacitate $c(u, v) \geq 0$ și două noduri speciale:

- o sursă s și
- o destinație t.

Dacă $(u, v) \notin E$, se consideră că $c(u, v) = 0$.
Scriem $u \rightsquigarrow v$ pentru a indică existența unei căi de la u la v, și presupunem că fiecare nod $v \in G$ este pe o cale de la s la t, adică există o cale $s \rightsquigarrow v \rightsquigarrow t$.

Observație

O rețea de transport este un graf conex, deci $|E| \geq |V| - 1$.

Definiție

Un flux în o rețea de transport G este o funcție $f : V \times V \to \mathbb{R}$ care satisface 3 condiții:

Restricție de capacitate: Pentru toți $u, v \in V$, $f(u, v) \leq c(u, v)$.

Antisimetrie: pentru toți $u, v \in V$, $f(u, v) = -f(v, u)$.

Conservarea fluxului: Pentru toți $u \in V - \{s, t\}$, $\sum_{v \in V} f(u, v) = 0$.

$f(u, v)$ se numește fluxul net de la nodul u la v. Valoarea fluxului f este $|f| = \sum_{v \in V} f(s, v)$, adică, fluxul total de rețea care pleacă din sursă.

Problema fluxului maxim

Se dă o rețea de transport G cu sursa s și destinația t,

Să se găsească un flux cu valoare maximă de la s la t.

- Fluxul pozitiv de rețea care intră într-un nod v este

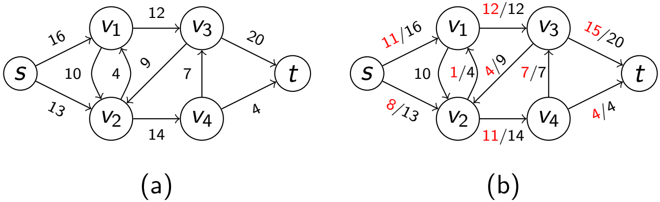
$$\sum_{\substack{u \in V \\ f(u,v)>0}} f(u,v)$$

- Fluxul pozitiv de rețea care pleacă dintr-un nod v este

$$\sum_{\substack{u \in V \\ f(v,u)>0}} f(v,u)$$

$\Rightarrow$ Din conservarea fluxului rezultă că pentru orice nod $v \notin \{s,t\}$: fluxul pozitiv care intră în v = fluxul pozitiv care pleacă din v.

Exemplu de rețea de transport

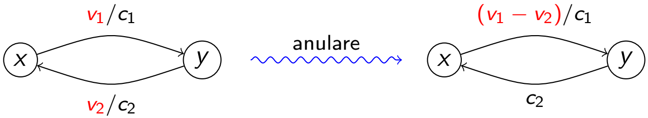


(a) Rețea de transport $G = (V, E)$ cu muchiile etichetate cu capacitățile lor. Sursa este s, destinația este t.

(b) Un flux f în rețeaua de transport G cu valoarea $|f| = 19$. Sunt indicate doar fluxurile pozitive. Dacă $f(u, v) > 0$, muchia (u, v) este etichetată cu $f(u, v)/c(u, v)$. (Notația cu '/' separă fluxul de capacitate; *nu* reprezintă împărțire.) Dacă $f(u, v) \leq 0$, muchia (u, v) este etichetată doar cu capacitatea ei.

Rețele de transport
Ștergerea tuturor fluxurilor negative de rețea – regula de anulare

Dacă $v_1 \geq v_2 > 0$ atunci

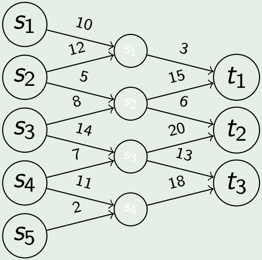


- Doar fluxurile pozitive de rețea reprezintă transporturi ce au loc.
- Aplicații ale regulii de anulare
  - elimină fluxurile negative de rețea.
  - nu violează cele 3 restricții ale rețelelor de transport:
    1. constrângerea de capacitate
    2. antisimetria
    3. conservarea fluxului

Surse și destinații multiple

- O problemă de debit maxim poate avea surse multiple $s_1, \ldots, s_m$ și destinații multiple $t_1, \ldots, t_n$.
- O astfel de problemă poate fi redusă la o problemă echivalentă cu o singură sursă și o singură destinație:
  - adaugă o **supersursă** s și o **superdestinație** t
  - adaugă muchii orientate (s, s_i) cu $c(s, s_i) = \infty$ pentru $i = 1..m$
  - adaugă muchii orientate (t_j, t) cu $c(t_j, t) = \infty$ pentru $j = 1..n$

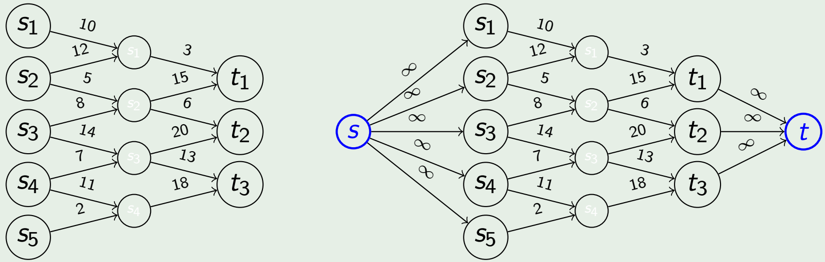
Exemplu



Surse și destinații multiple

- O problemă de debit maxim poate avea surse multiple $s_1, \ldots, s_m$ și destinații multiple $t_1, \ldots, t_n$.
- O astfel de problemă poate fi redusă la o problemă echivalentă cu o singură sursă și o singură destinație:
  - adaugă o **supersursă** s și o **superdestinație** t
  - adaugă muchii orientate (s, s_i) cu $c(s, s_i) = \infty$ pentru $i = 1..m$
  - adaugă muchii orientate (t_j, t) cu $c(t_j, t) = \infty$ pentru $j = 1..n$

Exemplu



- Se presupun date:

  o rețea de transport $G = (V, E)$
  o funcție f de la $V \times V$ la $\mathbb{R}$
  mulțimile de noduri X, Y (adică, $X \subseteq V$, $Y \subseteq V$)
  nodul $u \in V$.
- Atunci
  - $f(X, Y)$ reprezintă suma $\sum_{x \in X} \sum_{y \in Y} f(x, y)$.
  - $f(u, X)$ reprezintă suma $\sum_{x \in X} f(u, x)$.
  - $f(Y, u)$ reprezintă suma $\sum_{y \in Y} f(y, u)$.
  - $X - u$ reprezintă mulțimea $X - \{u\}$.

Remarcă. Dacă f este un flux pentru $G = (V, E)$ atunci $f(u, V) = 0$ pentru toți $u \in V - \{s, t\}$. Acest fapt rezultă din conservarea fluxului $\Rightarrow f(V - \{s, t\}, V) = 0$.

Proprietăți ale rețelelor de transport

Lemă

Fie $G = (V, E)$ o rețea de transport și f un flux în G. Dacă $x, Y, Z \subseteq V$ atunci

- $f(X, X) = 0$
- $f(X, Y) = -f(Y, X)$
- Dacă $X \cap Y = \emptyset$ atunci
  - $f(X \cup Y, Z) = f(X, Z) + f(Y, Z)$, și
  - $f(Z, X \cup Y) = f(Z, X) + f(Z, Y)$

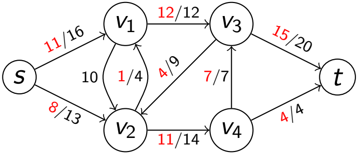
Se observă că:

$$
\begin{aligned}
{\color{red}|f|} &= f(s, V) && \text{cf. definiției} \\
&= f(V, V) - f(V - s, V) && \text{cf. lemei precedente} \\
&= f(V, V - s) && \text{cf. lemei precedente} \\
&= f(V, t) + f(V, V - \{s, t\}) && \text{cf. lemei precedente} \\
&{\color{red}= f(V, t)} && \text{cf. conservării fluxului}
\end{aligned}
$$

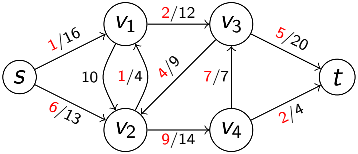
Definiție

Dacă f_1, f_2 sunt fluxuri în o rețea de transport G iar $\alpha \in \mathbb{R}$, atunci

- suma fluxurilor f_1 și f_2 este funcția $f_1 + f_2$ de la $V \times V$ la $\mathbb{R}$ definită de
$$(f_1 + f_2)(u, v) := f_1(u, v) + f_2(u, v) \quad \text{pentru toți } u, v \in V.$$
- produsul scalar αf_1 este fluxul definit de funcția de la $V \times V$ la $\mathbb{R}$ astfel încât
$$(\alpha f_1)(u, v) := \alpha f_1(u, v) \quad \text{pentru toți } u, v \in V.$$

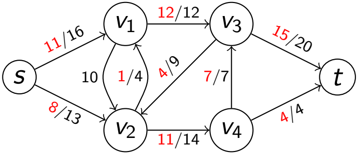


(a) G și f_1

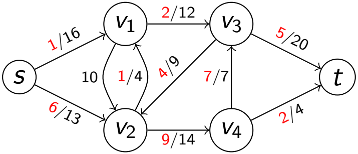


(b) G și f_2

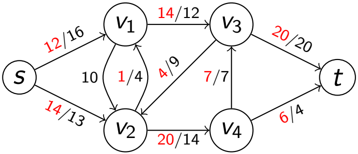
Operații cu fluxuri
Exemple



(a) G și f_1

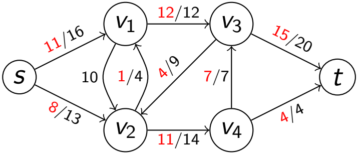


(b) G și f_2

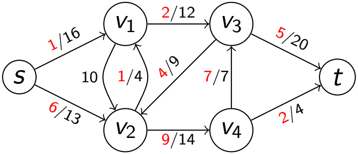


(c) G și $f_1 + f_2$

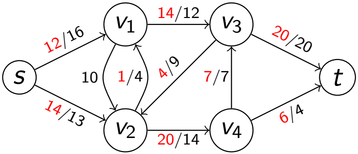
Operații cu fluxuri
Exemple



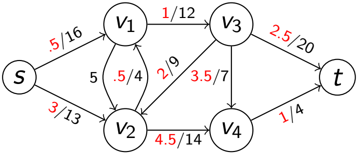
(a) G și f_1



(b) G și f_2



(c) G și $f_1 + f_2$



(d) G și αf_2 când $\alpha = \frac{1}{2}$

Un flux trebiue să satisfacă 3 constrângeri: restricția de capacitate, antisimetria și conservarea fluxului.

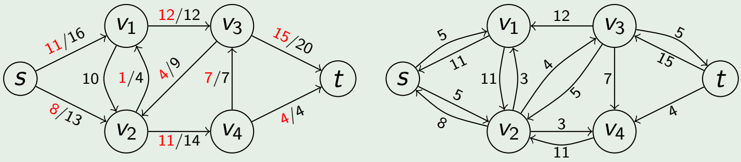
1. Care proprietăți nu sunt păstrate de sumele de flux?
2. Care proprietăți nu sunt păstrate de produsul scalar al fluxului?
3. Să se arate că dacă f_1, f_2 sunt fluxuri și $0 \leq \alpha \leq 1$, atunci $\alpha f_1 + (1 - \alpha) f_2$ este flux.

Rețele reziduale

Ipoteze: $G = (V, E)$: rețeaua de transport; f: flux în G.

- capacitatea reziduală a unei muchii (u, v) este $c_f(u, v) := c(u, v) - f(u, v)$.
- rețea reziduală a lui G indusă de f este rețeaua de transport $G_f = (V, E_f)$ unde $E_f = \{(u, v) \in V \times V \mid c_f(u, v) > 0\}$, iar capacitatea fiecărei muchii (u, v) este $c_f(u, v)$.

Exemplu



(a) G și f (b) G_f

Remarcă. În general, $|E_f| \leq 2\,|E|$.

Fie G o rețea de transport, f un flux în G, și rețeaua reziduală G_f. Dacă f' este un flux în G_f atunci $f+f'$ este flux în G cu valoarea $|f+f'| = |f| + |f'|$.

DEMONSTRAȚIE.

- **Antisimetria** are loc deoarece $(f+f')(u,v) = f(u,v) + f'(u,v) = -f(v,u) - f'(v,u) = -(f(v,u) + f'(v,u)) = -(f+f')(v,u)$.
- Pentru **constrângerile de capacitate** se observă că $f'(u,v) \leq c_f(u,v)$ pentru toți $u,v \in V$, deci $(f+f')(u,v) = f(u,v) + f'(u,v) \leq f(u,v) + (c(u,v) - f(u,v)) = c(u,v)$.
- Pentru **conservarea fluxului** observăm că

$$\begin{aligned}\sum_{v\in V}(f+f')(u,v) &= \sum_{v\in V}(f(u,v)+f'(u,v))\\ &= \sum_{v\in V} f(u,v) + \sum_{v\in V} f'(u,v) = 0+0=0.\end{aligned}$$

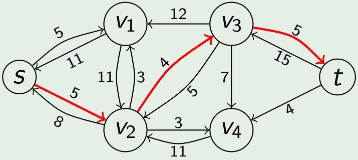
În final avem că

$$|f+f'| = \sum_{v\in V}(f+f')(s,v) = \sum_{v\in V}(f(s,v)+f'(s,v)) = \sum_{v\in V} f(s,v) + \sum_{v\in V} f'(s,v) = |f|+|f'|.$$

Drumuri de creștere

Un drum de creștere al unei rețele de transport G și a unui flux f este o cale simplă de la s la t în rețeaua reziduală G_f.

Exemplu (Drum de creștere)



Observații.

- Fiecare muchie (u, v) a unui drum de creștere admite un flux net pozitiv suplimentar fără ca să violeze capacitatea muchiei.
- În acest exemplu am fi putut trimite cel puțin 4 unități în plus de la s la t de-a lungul drumului de creștere marcat, fără a viola vreo restricție de capacitate (Remarcă: capacitatea reziduală minimă pe drumul de creștere marcat este 4).

Drumuri de creștere (continuare)

- Capacitatea reziduală a unui drum de creștere p este dată de

$$c_f(p) := \min\{c_f(u,v) \mid (u,v) \text{ este pe } p\}.$$

Lemă

Fie $G=(V,E)$ o rețea de transport cu un flux f, p un drum de creștere în G_f, și $f_p : V \times V \to \mathbb{R}$ definit de

$$f_p(u,v) := \begin{cases} c_f(p) & \text{dacă } (u,v) \text{ este pe } p, \\ -c_f(p) & \text{dacă } (v,u) \text{ este pe } p, \\ 0 & \text{în celelalte cazuri.} \end{cases}$$

Atunci f_p este un flux în G_f cu valoarea $|f_p| = c_f(p) > 0$.

Corolar

Fie $G=(V,E)$ o rețea de transport cu fluxul f, și p un drum de creștere în G_f. Fie f_p fluxul definit ca în lema precedentă. Atunci $f + f_p$ este un flux în G cu valoarea $|f'| = |f| + |f_p| > |f|$.

Metoda Ford-Fulkerson

Produce un flux maxim pentru o rețea de transport G:

FORD-FULKERSON-METHOD(G, s, t)
1 inițializează fluxul f cu 0
2 **while** există un drum de creștere p
3 crește fluxul f de-a lungul lui p
4 **return** f

- Metoda Ford-Fulkerson este corectă deoarece are loc următorul rezultat:

  Un flux este maxim dacă și numai dacă rețeaua lui reziduală nu conține drumuri de creștere.

Metoda Ford-Fulkerson

Produce un flux maxim pentru o rețea de transport G:

FORD-FULKERSON-METHOD(G, s, t)
1 inițializează fluxul f cu 0
2 **while** există un drum de creștere p
3 crește fluxul f de-a lungul lui p
4 **return** f

- Metoda Ford-Fulkerson este corectă deoarece are loc următorul rezultat:

  Un flux este maxim dacă și numai dacă rețeaua lui reziduală nu conține drumuri de creștere.

▷ Vom demonstra acest lucru.

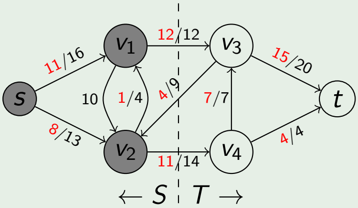
Noțiuni auxiliare: tăietură, capacitate a unei tăieturi.

Tăieturi

Definiție

O tăietură (S, T) a unei rețele de transport $G = (V, E)$ este o partiție a lui V în S și $T = V - S$ astfel încât $s \in S$ și $t \in T$. Fluxul net de-a lungul tăieturii (S, T) este $f(S, T)$. Capacitatea tăieturii (S, T) este $c(S, T)$.

Exemplu



$S = \{s, v_1, v_2\}$
$T = \{v_3, v_4, t\}$

$f(S, T) = f(v_1, v_3) + f(v_2, v_3) + f(v_2, v_4) = 12 + (-4) + 11 = 19$

$c(S, T) = c(v_1, v_3) + c(v_2, v_4) = 12 + 14 = 26$

Proprietăți ale tăieturilor

Lemă

Fluxul net de-a lungul oricărei tăieturi (S, T) este $f(S, T) = |f|$.

Corolar

Pentru orice flux f și orice tăietură (S, T) avem $|f| \leq c(S, T)$.

Teoremă (Flux maxim – tăietură minimă)

Dacă f este un flux în o rețea de transport $G = (V, E)$ cu sursa s și destinația t, atunci condițiile următoare sunt echivalente:

1. *f este un flux maxim în G.*
2. *G_f nu conține drumuri de creștere.*
3. *$|f| = c(S, T)$ pentru o tăietură (S, T) a lui G.*

Teorema flux maxim – tăietură minimă

(1) $\Rightarrow$ (2) Prin contradicție: Presupunem că f este flux maxim în G și că G_f are n drum de creștere p. Atunci $f + f_p$ ar fi un flux G cu valoare strict mai mare decât $|f|$, contradicție.

(2) $\Rightarrow$ (3) Presupunem că G_f nu are drum de creștere de la s la t. Fie $S = \{v \in V \mid$ există un drum de la s la v în $G_f\}$ și $T = V - S$. Atunci (S, T) este tăietură deoarece $s \in S$ și $t \notin S$. Pentru orice pereche de noduri $(u, v) \in S \times T$ avem $v(u, v) = c(u, v)$ deoarece în caz contrar $(u, v) \in E_f$ și $v \in S$. Rezultă că $|f| = f(S, T) = c(S, T)$.

(3) $\Rightarrow$ (1) Știm că $|f| \leq c(S, T)$ pentru toate tăieturile (S, T) ale lui G. Din condiția $|f| = c(S, T)$ rezultă că f este flux maxim.

Algoritmul Ford-Fulkerson de bază

FORD-FULKERSON(G, s, t)

for fiecare muchie (u, v) a lui G
    $f(u, v) := 0$
    $f(v, u) := 0$
while $\exists$ drum p de la s la t în G_f
    $c_f := \min\{c_f(u, v) \mid (u, v)$ este în $p\}$
    for fiecare muchie (u, v) în p
        $f(u, v) := f(u, v) + c_f(p)$
        $f(v, u) := -f(u, v)$

Algoritmul Ford-Fulkerson de bază

Exemplu ilustrat: inițial, $|f| = 0$

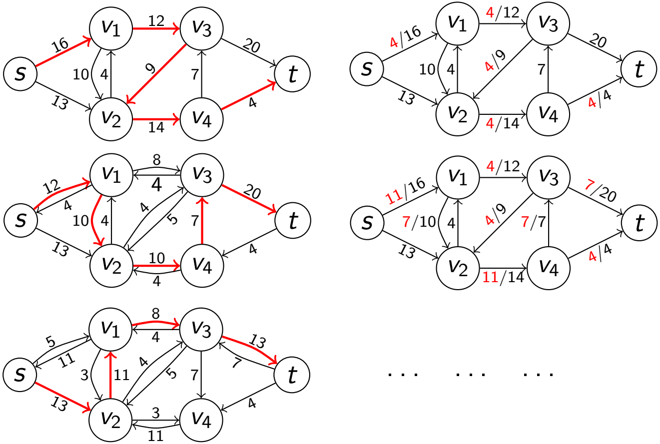
Rețea reziduală G_f cu drum de creștere (linia 4)

Flux net ce rezultă din adăugarea lui f_p la f

(a)

(b)

(c)



Exercițiu: să se deseneze grafurile pentru pașii rămași ai algoritmului lui Ford-Fulkerson.

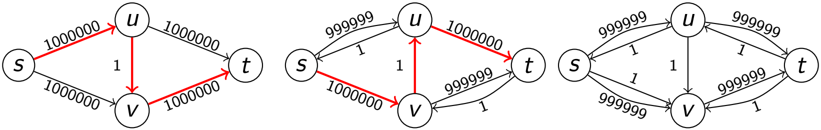
- Timpul de execuție depinde de felul cum se calculează drumul de creștere p în linia 4 a algoritmului.

- Timpul de execuție depinde de felul cum se calculează drumul de creștere p în linia 4 a algoritmului.
- IPOTEZĂ: toate muchiile au numere întregi ca și capacități (adică 0,1,2,...).

- Timpul de execuție depinde de felul cum se calculează drumul de creștere p în linia 4 a algoritmului.
- IPOTEZĂ: toate muchiile au numere întregi ca și capacități (adică 0,1,2,...).
    - Dacă capacitățile sunt numere raționale, le putem înlocui cu numere întregi făcând o operație de scalare corespunzătoare.

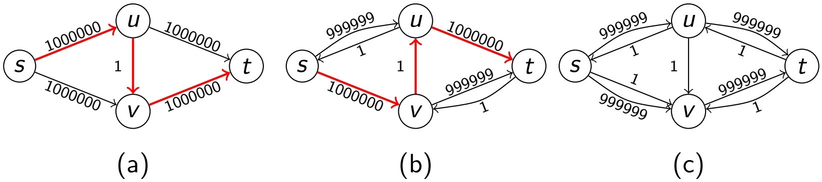
- Timpul de execuție depinde de felul cum se calculează drumul de creștere p în linia 4 a algoritmului.
- IPOTEZĂ: toate muchiile au numere întregi ca și capacități (adică 0,1,2,...).
  - Dacă capacitățile sunt numere raționale, le putem înlocui cu numere întregi făcând o operație de scalare corespunzătoare.
- O implementare directă a algoritmului FORD-FULKERSON durează $O(E \cdot |f^*|)$ unde f^* este fluxul maxim găsit de algoritm.

  Motiv: bucla **while** din liniile 4-8 se execută de cel mult $|f^*|$ ori deoarece valoarea fluxului crește cu cel puțin 1 de fiecare dată.



(a) (b) (c)

- Un flux maxim f^* în rețeaua de transport (a) are $|f^*| = 2000000$. A alegere proastă de drum de creștere cu capacitatea 1 este marcată cu roșu.
- (b) și (c) ilustrează rețelele reziduale produse după creșterea cu drumul de creștere marcat cu roșu.



- Un flux maxim f^* în rețeaua de transport (a) are $|f^*| = 2000000$. A alegere proastă de drum de creștere cu capacitatea 1 este marcată cu roșu.
- (b) și (c) ilustrează rețelele reziduale produse după creșterea cu drumul de creștere marcat cu roșu.
- Complexitatea algoritmului se îmbunătățește dacă p în linia 4 se calculează folosind o căutare în lățime care garantează că p este o cale *cea mai scurtă* de la s la t în rețeaua reziduală, în care fiecare muchie are o distanță unitară (greutate) $\Rightarrow$ algoritmul Edmonds-Karp cu complexitatea $O(|V| \cdot |E|^2)$.

Fie $B = (V, E)$ un graf bipartit între submulțimile V_1 și V_2 ale lui V.

Definiție (cuplaj, cuplaj maxim)

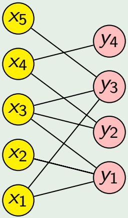
Un cuplaj în B este o mulțime de muchii $C \subseteq E$ cu proprietatea că oricare 2 muchii din C nu au un capăt comun. Cuplajul C este maxim dacă are un număr maxim de muchii.

Calculul unui cuplaj maxim în $B = (V_1 \cup V_2, E)$ se poate face astfel:

1. Extindem B cu 2 noduri noi: s (sursă) și t (destinație), apoi cu arcuri cu capacitatea 1 de la s la toate nodurile din V_1, și de la toate nodurile din V_2 la t. Toate muchiile lui G se orientează de la V_1 la V_2, cu capacitatea 1.
2. Se calculează un flux maxim de la s la t.

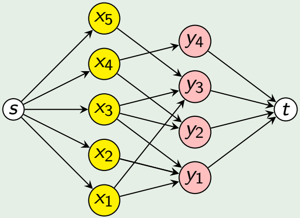
Construirea unei rețele de flux pentru un graf bipartit:

Exemplu



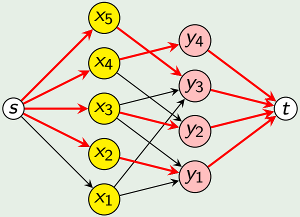
Construirea unei rețele de flux pentru un graf bipartit:

Exemplu



Construirea unei rețele de flux pentru un graf bipartit:

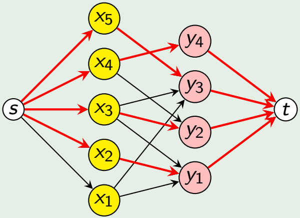
Exemplu



Cuplaj maxim $C = \{(x_2, y_1), (x_3, y_2), (x_4, y_4), (x_5, y_3)\}$

Construirea unei rețele de flux pentru un graf bipartit:

Exemplu



Cuplaj maxim $C = \{(x_2, y_1), (x_3, y_2), (x_4, y_4), (x_5, y_3)\}$

Teoremă

Fie G rețeaua de flux construită pentru un graf bipartit $B = (V_1 \cup V_2, E)$ și f un flux maxim în G. Atunci mulțimea muchiilor (u, v) ale lui f cu $u \in V_1$, $v \in V_2$ și $f(u, v) = 1$ este un cuplaj maxim în B.

Problemă

$G = (V, E)$: rețea de transport în care muchiile (u, v) au asociate, pe lângă o capacitate, și un cost unitar $cost(u, v) \geq 0$.
Un flux maxim de cost minim în G este un flux maxim f în G pentru care suma

$$\sum_{(u,v)\in E} f(u, v) \cdot cost(u, v)$$

este minimă.

Soluție: Adaptare a algoritmului Edmonds-Karp

- Se asociază costuri la toate muchiile din rețelele reziduale ale unui flux f:
  - muchia (u, v) are costul $cost(u, v)$ dacă $c(u, v) > f(u, v)$ în rețeaua originală
  - muchia (u, v) are costul $-cost(u, v)$ dacă $f(u, v) < 0$ în rețeaua originală
- În loc de drum cel mai scurt de la sursa s la destinația t, se caută un drum p de la s la t cu cost minim în rețeaua reziduală.
  - p poate fi găsit cu algoritmul lui Bellman-Ford.
- Apoi, fluxul va fi incrementat pe drumul p cu valoarea maximă posibilă (=minimul diferențelor dintre capacitate si flux pentru fiecare arc de pe drum).

Bibliografie

Capitolul **27** din

- T. H. Cormen, C. E. Leiserson, R. L. Rivest. *Introduction to Algorithms*. MIT Press, 2000.