Fluxuri în rețele

Într-o rețea de transport (de apă, curent, gaze naturale, rutieră etc.) se pot pune diverse probleme practice. Dintre acestea cele mai cunoscute și studiate sunt problema fluxului maxim (cantitatea maximă ce poate fi transportată de la nodurile sursă la nodurile destinație), precum și problema fluxului de cost minim (pentru a se transporta o anumită cantitate de la nodurile sursă la nodurile destinație se dorește a se determina rutele cu costul cel mai mic pe care se poate face acest lucru). În continuare vom formaliza și vom prezenta algoritmi de rezolvare a acestor probleme.

Pentru graful orientat $G = (V, E)$ cu V mulțimea celor $n \in \boldsymbol{N}^*$ nodurilor şi E mulțimea celor $m \in \boldsymbol{N}^*$ arce ale grafului, se definește funcția *capacitate* $c : E \rightarrow \boldsymbol{R}_+$. Prin $c(x, y)$ se înțelege cantitatea maximă flux ce poate traversa arcul (x, y) (cantitatea ce poate fi maxim transportată de la nodul x la nodul y).

Definiția ?.1: Un triplet de forma $G = (V, E, c)$ poartă denumirea de *rețea de transport*.

Definiția ?.2: O aplicație $v : V \rightarrow \boldsymbol{R}$ care are proprietatea că:

(?.1) $\sum_{x \in V} v(x) = 0$

se numește funcție *valoare a nodurilor*. Nodurile $x \in V$ cu proprietatea că $v(x) > 0$ se numesc *noduri sursă*, nodurile $x \in V$ cu $v(x) < 0$ se numesc *noduri stoc*, iar nodurile $x \in V$ cu $v(x) = 0$ se numesc *noduri de transfer* sau *noduri intermediare*.

Definiția ?.3: *Problema fluxului* de valoare v în rețeaua G constă în a găsi funcția $f : V \rightarrow \boldsymbol{R}_+$ care satisface condițiile:

(?.2) $\sum_{\substack{y \in V \\ (x,y) \in E}} f(x,y) - \sum_{\substack{y \in V \\ (y,x) \in E}} f(y,x) = v(x), \ \forall\, x \in V$

şi

(?.3) $f(x, y) \leq c(x, y), \ \forall (x, y) \in E.$

Definiția ?.4: Relațiile (?.2) se numesc *constrângeri de conservare ale fluxului*, iar (?.3) poartă denumirea de *condiții de mărginire ale fluxului*.

Definiția ?.5: O funcție $f : E \rightarrow \boldsymbol{R}_+$ pentru care există o funcție valoare a nodurilor v şi f satisface simultan constrângerile (2.2) şi (2.3) se numeşte *flux admisibil* pentru rețeaua G.

Definiția ?.6: Prin *valoarea fluxului* admisibil *f* se înțelege $\sum_{\substack{x\in V \\ v(x)>0}} v(x) = -\sum_{\substack{x\in V \\ v(x)<0}} v(x)$. Această valoare se notează cu *v(f)*.

Se observă uşor că, dacă în rețeaua *G* există arce paralele, atunci problema fluxului în *G* este echivalentă cu problema fluxului în rețeaua *G' = (N', A', c)*, care se obține din *G* astfel: pentru fiecare pereche de arce paralele *a, b* $\in E$ ce au capătul inițial *x* şi capătul final *y*, în mulțimea nodurilor se introduce un nod suplimentar *z* și în mulțimea arcelor se introduc arcele b_1 *= (x, z)* şi b_2 *= (z, y)* având capacitățile *c'(x, z) = c'(z, y) = c(b),* iar arcul *b* este eliminat. Pentru fiecare arc *a = (x, y)* care s-a păstrat din rețeaua inițială *G* avem *c'(x, y) = c(a)*. De asemenea, pentru fiecare nod *x* din rețeaua inițială în rețeaua modificată vom avea *v'(x) = v(x)*. Pentru nodurile *z* nou introduse se consideră că *v'(z) = 0* (vezi fig. ?.1).

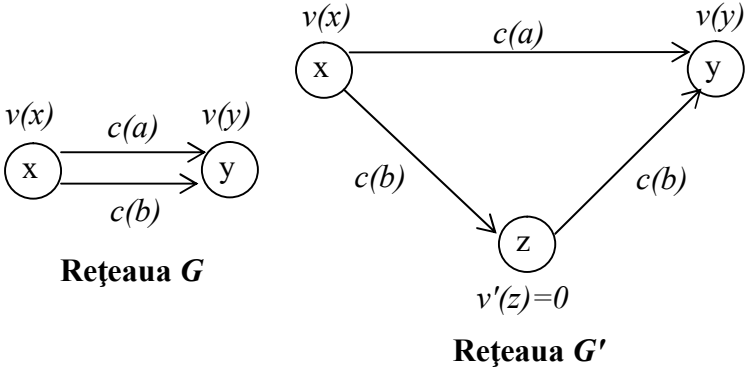

Fig. ?.1: Eliminarea arcelor paralele

Aşadar, fără a restrânge generalitatea problemei fluxului, putem considera că rețeaua *G* este definită pe un digraf, adică un graf fără arce paralele.

Definiția ?.7: Un graf *G = (V, E)* se numeşte *antisimetric*, dacă:

(?.4) $\forall\ (x, y) \in E: (y, x) \notin E.$

Pentru o rețea care nu este antisimetrică se poate construi o rețea antisimetrică echivalentă prin introducerea unor noduri *z* în rețea (vezi fig. ?.2). Odată cu introducerea unui nod *z* în rețeaua *G* se introduc şi arcele *(x, z)* şi *(z,y)* capacitățile *c(x, z) = c(z, y) = c(x, y)*. În final, arcul *(x, y)* este eliminat din rețeaua *G*.

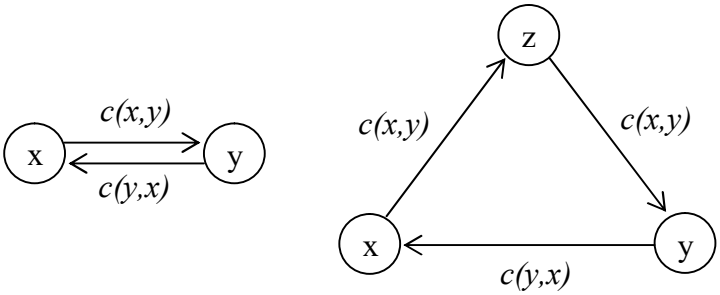

Fig. ?.2: Introducerea nodului *z* în rețeaua *G*

Definiția ?.8: Un flux admisibil f_1 spunem că este *mai mic* decât fluxul admisibil f_2 şi notăm $f_1 < f_2$, dacă $f_1(a) \leq$

$f_2(a)$, pentru orice arc $a \in E$ şi există cel puțin un arc $b \in E$ astfel încât $f_1(b) < f_2(b)$.

Definiția ?.9: Fluxul admisibil f este *suma fluxurilor* admisibile f_1 şi f_2, dacă $f(a) = f_1(a) + f_2(a)$, pentru oricare arc $a \in A$. Se notează $f = f_1 + f_2$.

Definiția ?.10: Un flux admisibil f se numeşte *flux elementar* dacă există un drum D de la un nod sursă la un nod stoc sau dacă există un circuit $\overset{o}{D}$ în rețeaua G, astfel încât fluxul este 0 pe arcele ce nu aparțin drumului D, respectiv circuitului $\overset{o}{D}$, iar pe arcele drumului, respectiv ale circuitului, fluxul are o valoare nenegativă u, adică:

$$(?.5) \qquad f(x,y) = \begin{cases} 0, \text{ daca } (x,y) \notin D \quad (\overset{o}{D}) \\ u, \text{ daca } (x,y) \in D \quad (\overset{o}{D}) \end{cases}.$$

Orice flux admisibil poate fi descompus într-o sumă de fluxuri elementare cu ajutorul unui algoritm care are complexitatea $O(m^2)$ [1].

?.1 Flux maxim

Una dintre cele mai cunoscute probleme de optimizare combinatorie este cea a găsirii fluxului admisibil cu valoarea cea mai mare. Această problemă poartă denumirea de *problema fluxului maxim* şi poate fi enunțată astfel:

$$(?.6) \quad \begin{cases} \max v(f) \\ f \text{ este un flux admisibil în } G \end{cases}.$$

Definiția ?.11: O rețea cu un singur nod sursă s şi un singur nod stoc t se numeşte *rețea s-t*. O astfel de rețea se mai notează $G = (V, E, c, s, t)$.

O rețea care are mai mult de un nod sursă şi/sau mai mult de un nod stoc poate fi transformată într-o rețea *s-t* echivalentă, introducând două noduri suplimentare denumite nod *super-sursă* şi, respectiv, nod *super-stoc*. Pentru nodul super-sursă se introduc arce suplimentare care leagă acest nod de sursele rețelei, iar pentru nodul super-stoc se introduc arce care leagă nodurile stoc de super-stoc (vezi figura 2.3).

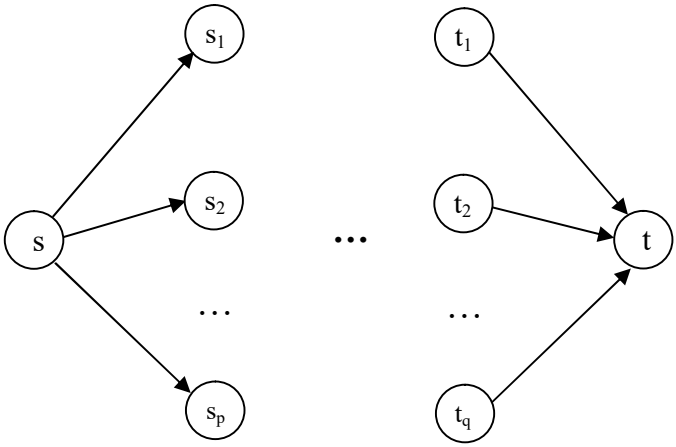

Fig. ?.3: ***Introducere super-sursă şi super-stoc***

Fie s_i, cu $i \in \{1, 2, ..., p\}$ nodurile sursă şi t_j, $j \in \{1, 2, ..., q\}$ nodurile stoc ale rețelei. Notăm cu s super-sursa şi respectiv cu t super-stocul. Arcele nou introduse odată cu super-sursa sunt (s, s_i), $i \in \{1, 2, ..., p\}$. Arcele care se introduc pentru super-stoc sunt (t_j, t), cu $j \in \{1, 2, ..., q\}$ (vezi figura ?.3). Pentru aceste arce nou introduse, de obicei, în literatura de specialitate, capacitatea se consideră a fi $+\infty$. Este uşor de văzut că în loc de $+\infty$ putem considera capacitățile pentru arcele nou introduse ca fiind:

$$(?.7) \quad \begin{cases} c(s, s_i) = \sum_{(s_i, y) \in E} c(s_i, y) \ , \ \forall \ i \in \{1, \ 2, \ \ldots, \ p\} \\ c(t_j, t) = \sum_{(x, t_j) \in E} c(x, t_j) \ , \forall \ j \in \{1, \ 2, \ \ldots, \ q\} \end{cases} .$$

Deci, problema fluxului în rețeaua inițială este echivalentă cu o problemă a fluxului într-o rețea s-t. De aceea, în continuare vom lucra numai cu rețele s-t, fără a restrânge generalitatea problemei fluxului (maxim).

Constrângerile (?.2) de conservare ale fluxului se rescriu (mai simplu) pentru o rețea *s-t* astfel:

$$(?.2') \quad \sum_{(x,y)\in E,\ y\in V} f(x,y) - \sum_{(y,x)\in E,\ y\in V} f(y,x) = \begin{cases} v, \text{ daca } x = s \\ 0, \text{ daca } x \in V\backslash\{s,t\}, \ \forall x \in V, \\ -v, \text{ daca } x = t \end{cases}$$

unde v este un număr real pozitiv.

?.2 Rețea reziduală

Noțiunea de rețea reziduală ataşată unei rețele $G = (V, E, c)$ pentru un flux admisibil f prezintă un interes deosebit, deoarece abordarea problemei fluxului maxim se simplifică prin faptul că în rețeaua reziduală se lucrează cu o singură valoare, ataşată unui arc, denumită *capacitate reziduală*, în loc de capacitate şi flux, aşa cum se întâmplă în rețeaua G. De asemenea, în rețeaua reziduală se lucrează cu drumuri şi nu cu lanțuri.

Rețeaua reziduală ataşată rețelei $G = (V, E, c)$ şi fluxului admisibil f se notează cu $G_f = (V, E_f, r)$, unde E_f este

mulțimea arcelor rețelei reziduale, iar $r : E_f \rightarrow \boldsymbol{R}_+$ este aplicația care ataşează capacitatea reziduală arcelor rețelei reziduale. Funcția r se defineşte pentru fiecare pereche $(x, y) \in V \times V$ astfel:

$$(?.9) \quad r(x,y) = \begin{cases} 0, (x,y) \notin A \text{ şi } (y,x) \notin A \\ c(x,y) - f(x,y), (x,y) \in A \text{ şi } (y,x) \notin A \\ f(y,x), (x,y) \notin A \text{ şi } (y,x) \in A \\ c(x,y) - f(x,y) + f(y,x), \text{ altfel} \end{cases}$$

Mulțimea E_f a arcelor rețelei reziduale este formată de perechile ordonate de noduri pentru care capacitatea reziduală este pozitivă:

(?.10) $E_f = \{(x, y) \in V \times V \mid r(x, y) > 0\}$.

Pentru a înțelege semnificația capacității reziduale să observăm $c(x, y) - f(x, y)$ reprezintă capacitatea nefolosită a arcului (x, y), adică este valoarea cu care fluxul mai poate fi mărit pe arcul (x, y), iar $f(y, x)$ reprezintă valoarea maximă cu care fluxul poate fi micşorat pe arcul (y, x) şi, implicit, mărit de la x la y.

?.3 Tăietură minimă

Din punct de vedere practic, o tăietură într-o rețea înseamnă a determina o mulțime de arce care, dacă sunt eliminate, nodul sursă nu mai comunică cu nodul stoc. Prezintă evident interes să găsim tăietura cu suma capacităților minimă ce deconectează sursa de stoc.

Fie $G = (V, E, c)$ o rețea şi X o mulțime de noduri ($X \subset V$). Notăm cu $\bar{X} = V \backslash X$.

Definiția ?.12: Mulțimea de arce

$$[X, \bar{X}] = (X, \bar{X}) \cup (\bar{X}, X)$$

se numeşte *tăietură*, unde:

$$(X, \bar{X}) = \{(x, y) \in E | x \in X \text{ şi } y \in \bar{X}\}$$

şi

$$(\bar{X}, X) = \{(x, y) \in E | x \in \bar{X} \text{ şi } y \in X\}$$

Arcele din $(X, \bar{X})$ se numesc *arce directe ale tăieturii*, iar arcele din $(\bar{X}, X)$ se numesc *arce inverse ale tăieturii*.

Definiția ?.13: Dacă $[X, \bar{X}]$ este o tăietură într-o rețea *s-t* şi $s \in X$, $t \in \bar{X}$, atunci $[X, \bar{X}]$ se numeşte *tăietură s-t*.

Definiția ?.14: *Capacitatea unei tăieturi s-t* $[X, \overline{X}]$ se defineşte ca fiind diferența dintre suma capacităților arcelor directe ale tăieturii:

(?.11) $c[X, \bar{X}] = \sum_{(x,y)\in(X,\bar{X})} c(c, y)$.

Definiția ?.15: O tăietură s-t $[X, \overline{X}]$ se numeşte *tăietură minimă* dacă, dintre toate tăieturile s-t, $[X, \overline{X}]$ are capacitatea minimă.

?.4 Determinarea fluxului maxim și a tăieturii minime

Definiția ?.16: *Valoarea fluxului pentru tăietura* $[X, \overline{X}]$ se defineşte ca fiind diferența dintre suma valorilor fluxului pe arcele directe ale tăieturii şi suma valorilor fluxului pe arcele inverse tăieturii:

(?.12)

$$f[X, \overline{X}] = f(X, \overline{X}) - f(\overline{X}, X) = \sum_{(x,y)\in(X,\overline{X})} f(x, y) - \sum_{(x,y)\in(\overline{X},X)} f(x, y)$$

Teorema ?.1: Într-o rețea *s-t*, valoarea unui flux admisibil *f* este egală cu valoarea fluxului pe tăietura s-t

$[X, \overline{X}]$ şi este mai mică sau egală cu capacitatea tăieturii $[X, \overline{X}]$, adică avem:

$$v(f) = f[X, \overline{X}] \leq c[X, \overline{X}].$$

Demonstrație: Se găseşte în [1].
□

Definiția ?.17: Un lanț $L = (s=x_1, x_2, ..., x_k=t)$ de la s la t în rețeaua G se numeşte *lanț de mărire a fluxului f* dacă avem că $f(x_i, x_{i+1}) < c(x_i, x_{i+1})$ sau $f(x_{i+1}, x_i) > 0$, pentru oricare $i \in \{1, 2, ..., k-1\}$.

Definiția ?.18: Orice drum $D = (s=x_1, x_2, ..., x_k=t)$ de la s la t în rețeaua reziduală G_f se numeşte drum *de mărire a fluxului f.*

Este uşor de observat ca un drum de mărire a fluxului în rețeaua reziduală corespunde unui lanț de mărire a fluxului în rețeaua G şi invers. De aceea, studierea problemei fluxului maxim se preferă a fi făcută în rețeaua reziduală (fiind mai simplă), iar orice raționament sau modificare ce se face în rețeaua reziduală se reflectă imediat şi în rețeaua G.

Teorema ?.2 (Legătura flux maxim şi tăietură minimă):

Valoarea fluxului maxim f^* într-o rețea este egală cu valoarea capacității tăieturii *s*-*t* minime $[X^*, \bar{X}^*]$:

$$v(f^*) = c[X^*, \bar{X}^*].$$

Demonstrație: Se găseşte în [1].

□

Definiție ?.19: Într-o rețea, un nod y se numeşte *accesibil din* x, dacă există un drum de la x la y.

Teorema ?.3 (Determinarea tăieturii minime pornind de la fluxul maxim):
Nodurile accesibile din s în rețeaua reziduală ataşată unui flux maxim într-o rețea G determină mulțimea X^* a tăieturii minime $[X^*, \bar{X}^*]$ în rețeua G.

Demonstrație: Se găseşte în [1].

□

Teorema ?.3 arată modul în care se determină tăietura minimă pornind de la fluxul maxim. Cu alte cuvinte, după ce găsim fluxul maxim, ataşăm rețeaua reziduală acestui flux şi, folosind un algoritm de parcurgere în rețeaua reziduală, determinăm nodurile accesibile X^* din s. $[X^*, \bar{X}^*]$ va fi tăietură minimă, unde, evident, $\bar{X}^* = V - X^*$.

Teorema ?.4: Un flux f admisibil într-o rețea G este maxim dacă şi numai dacă nu există nici un drum de mărire a fluxului în rețeaua reziduală G_f.

Demonstrație: Se găseşte în [1].
□

Această ultimă teoremă stă la baza algoritmului generic sau Ford-Fulkerson de găsire a fluxului maxim într-o rețea G, adică, atâta timp cât există drum de mărire a fluxului în rețeaua reziduală G_f se efectuează mărirea de flux de-a lungul unui astfel de drum.

Algoritmul Ford-Fulkerson pentru flux maxim:

```
Se porneşte cu un flux admisibil f;
Se construieşte reţeaua reziduală Gf;
Cât timp există drum în Gf execută
      Se determină un drum D în Gf;
      r(D):=min{r(x,y)|(x,y)∈D};
      Se măreşte fluxul f pe drumul D;
      Se actualizează Gf;
sfârşit cât timp;
```

După cum se poate vedea mai sus, algoritmul pornește cu un flux adimisibil în rețeaua G. Să observăm că fluxul identic zero (adică f valoarea 0 pe fiecare arc) este admisibil. Deci se poate porni cu acest flux.

În algoritm, după determinarea unui drum D, pentru arcele (x, y) care nu aparțin lui D, fluxul și rețeaua reziduală rămân nemodificate. Mărirea de flux se efectuează pentru fiecare arc (x, y) al drumului D din rețeaua reziduală după cum urmează:

1. Dacă (x, y) este arc în rețeaua G, atunci fluxul pe acest arc crește cu valoarea $r(D)$, adică avem $f(x,y) := f(x,y) + r(D)$.
2. Dacă (y, x) este arc în rețeaua G, atunci fluxul pe arcul (y, x) scade cu valoarea $r(D)$, adică $f(y,x) := f(y,x) - r(D)$.

Facem observația că, dacă penru un arc (x, y) al drumului D în rețeaua reziduală avem în G atât arcul (x, y) cât și arcul (y, x), atunci putem aplica oricare dintre cele două măriri de flux descrise mai sus sau, parțial, atât pe arcul (x, y), cât și pe arcul (y, x) astfel încât suma celor două măriri să fie $r(D)$. Mărirea pe ambele arcuri devine obligatorie atunci când $f(x, y) + r(D) > c(x, y)$ și $f(y, x) < r(D)$, deoarece, după mărire, condițiile (2.3) ar fi incălcate sau fluxul ar ajunge negative pe unele arce.

În ambele cazuri descrise mai sus, în rețeaua reziduală pe fiecare arc (x, y) al drumului D capacitatea reziduală scade cu valoarea $r(D)$, adică: $r(x, y) := r(x, y) - r(D)$, iar pe arcul (y, x) crește cu valoarea $r(D)$, adică: $r(y, x) := r(y, x) + r(D)$.

Teorema ?.5 (Corectitudinea alg. Ford-Fulkerson):
Dacă în rețeaua G capacitatea şi fluxul inițial sunt cu valori întregi, atunci algoritmului Ford-Fulkerson se termină în număr finit de pași și determină fluxul maxim în rețeaua G.

Demonstrație: La fiecare iterație a ciclului "cât timp" valoarea fluxului crește cu valoarea capacității reziduale a drumului, adică cel puțin cu o unitate. În consecință, după cel mult $v(f^*)$ iterații execuția algoritmului se încheie, unde f^* este fluxul maxim în rețeaua G. Din ciclul "cât timp" se iese atunci când nu mai există drum de mărire a fluxului, ceea ce înseamnă că la încheierea algoritmului fluxul f este maxim în rețeaua G, conform cu teorema ?.4.
□

Este uşor de observat că algoritmul Ford-Fulkerson poate fi aplicat şi rețelelor cu valori raționale (capacități şi flux inițial) din cauză că toate aceste valori raționale pot fi aduse la numitor comun.

Există exemple în literatura de specialitate care arată însă că dacă se aplică algoritmul Ford-Fulkerson pentru o rețea cu capacități iraționale, rezultatul obținut poate să nu fie corect, iar algoritmul poate să aibă un număr infinit de iterații.

Teorema ?.6 (Complexitatea alg. Ford-Fulkerson):

Dacă în rețeaua *G* capacitatea şi fluxul inițial sunt cu valori întregi, atunci complexitatea algoritmului Ford-Fulkerson este *O(m·n·C)*, unde:

$$C = \max\{\, c(x,y) \mid (x,y) \in A\}\,.$$

Demonstrație: După cum am văzut în demonstrația teoremei ?.5, după cel mult $v(f^*)$ iterații execuția algoritmului se încheie, unde f^* este fluxul maxim in rețeaua G. În plus, folosind (?.2') avem:

$$v(f^*) = \sum_{\substack{y\in V\\(s,y)\in E}} f^*(s,y) - \sum_{\substack{x\in V\\(x,s)\in E}} f^*(x,s) \le \sum_{\substack{y\in V\\(s,y)\in E}} f^*(s,y) \le \sum_{\substack{y\in V\\(s,y)\in E}} c(s,y) \le n\cdot C$$

Un drum de mărire a fluxului se determină în complexitate $O(m)$ folosind un algoritm de parcurgere în adăncime sau în lățime. Așadar, complexitatea algoritmului Ford-Fulkerson este: $O\big(m\cdot v(f^*)\big) = O(m\cdot n\cdot C)$.

□

Este evident că pentru execuția algoritmului Ford-Fulkerson cu valori întregi este nevoie de *O(m)* spațiu de memorie, dacă rețeaua este memorată folosind liste de adiacență.

Algoritmul Ford-Fulkerson este uşor de implementat, însă marele său dezavantaj este dat de faptul că el este un algoritm non-polinomial, complexitatea lui depinzând liniar de capacitatea maximă a arcelor. De multe ori, în practică avem capacitatea arcelor mare sau foarte mare. De aceea prezintă interes găsirea unor algoritmi care au complexitatea ce depinde de *log(C)* (așa numiții algoritmi slab polinomiali) sau care nu depinde deloc de capacitate (algoritmi tare polinomiali). În continuare vom prezenta un tabel cu algoritmii cunoscuți în prezent pentru flux maxim, iar pentru fiecare dintre aceștia este prezentată complexitatea.

Algoritm	**Complexitate**
Algoritmul Ford-Fulkerson	$O(m \cdot n \cdot C)$
Algoritmul Gabow al scalării bit a capacității	$O(m^2 \cdot log(C))$
Algoritmul de scalare Ahuja-Orlin al scalării maxime a capacității	$O(m^2 \cdot log(C))$
Algoritmul Edmons-Karp al drumului cel mai scurt	$O(n^2 \cdot m)$
Algoritmul Ahuja-Orlin al drumului cel mai scurt	$O(n^2 \cdot m)$
Algoritmul Dinic al rețelelor stratificate	$O(n^2 \cdot m)$
Algoritmul Ahuja-Orlin al rețelelor stratificate	$O(n^2 \cdot m)$

Algoritmul preflux FIFO	$O(n^3)$
Algoritmul preflux cu eticheta cea mai mare	$O(n^2 \cdot m^{1/2})$
Algoritmul preflux cu fluxuri deblocare	$O(n^2 \cdot m)$
Algoritmul de scalare a excesului	$O(nm + n^2 \cdot \log(C))$
Algoritmul lui Goldberg şi al lui Rao	$O(\min\{n^{2/3}, m^{1/2}\} \cdot m \cdot \log(n^2/m) \cdot \log(C))$
Algoritmul lui Goldberg şi al lui Tarjan	$O(n \cdot m \cdot \log(n^2/m))$
Algoritmul lui Orlin	$O(n \cdot m)$

În tabelul de mai sus cu n este notat numărul de noduri al rețelei G, m este numărul de arce al rețelei, iar cu C s-a notat valoarea maximă a capacității pe arce, adică $C = max\{c(x,y) \mid (x,y) \in A\}$.

În continuare vom lua un exemplu pentru a ilustra funcționalitatea algoritmului Ford-Fulkerson. Rețeaua G în care se caută fluxul maxim este dată în figura ?.4.

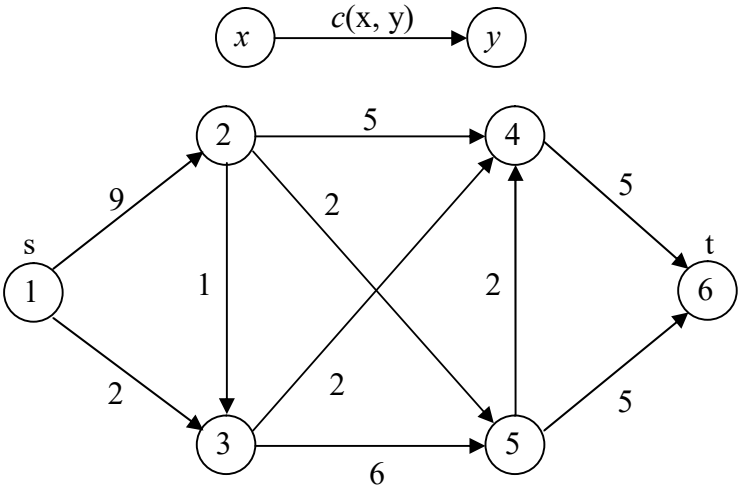

Fig. ?.4 Rețeaua *G*

Plecăm cu fluxul inițial $f = 0$. În consecință, se observă că, aplicând formula (?.9), rețeaua reziduală inițială G_f este identică cu cea din figura ?.4, considerând pe arce capacități reziduale. Aplicând algoritmul Ford-Fulkerson putem avea următoarele iterații:

Iterația 1: Considerăm drumul D_1 = (1, 2, 4, 6) în G_f cu $r(D_1) = \min(9,5,5) = 5$.

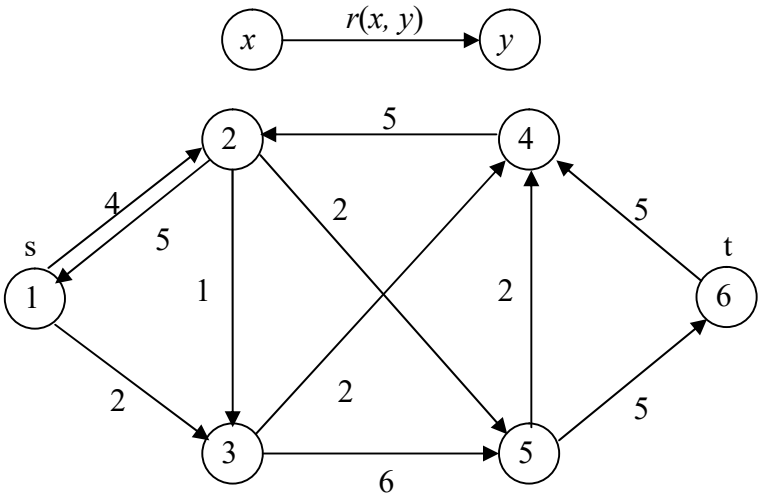

Fig. ?.5 Rețeaua reziduală G_f

Iterația 2: Alegem drumul D_2 = (1, 3, 5, 6) în G_f din fig. ?.5, având r(D_2) = 2.

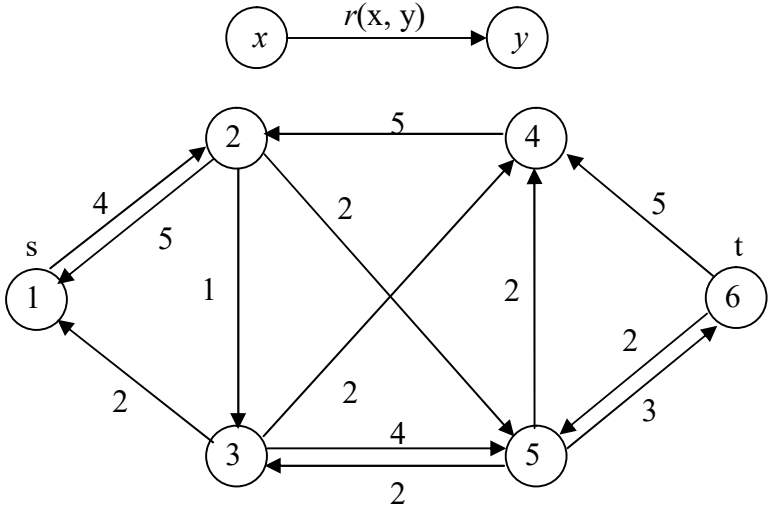

Fig. ?.6 Rețeaua reziduală G_f

Iterația 3: Considerăm drumul D_3 = (1, 2, 5, 6) în G_f, din fig. ?.6 cu r(D_3) = 2.

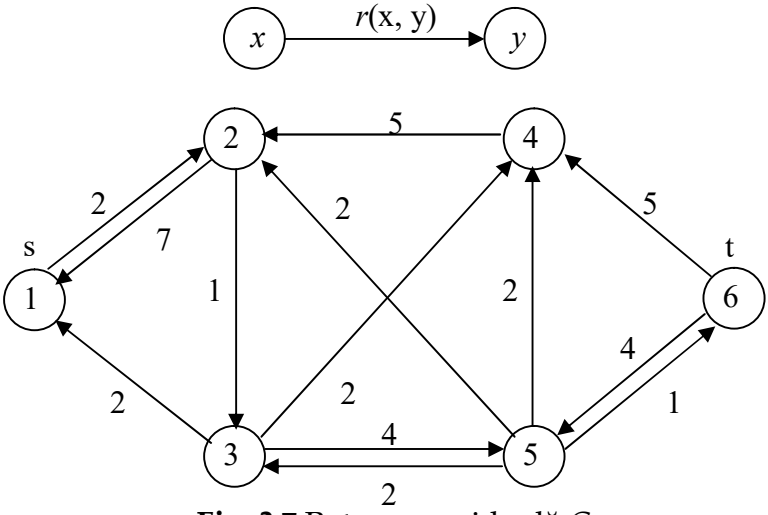

Fig. ?.7 Rețeaua reziduală G_f

Iterația 4: Considerăm drumul D_4 = (1, 2, 3, 5, 6) în G_f, din fig. ?.7 cu r(D_4) = 1.

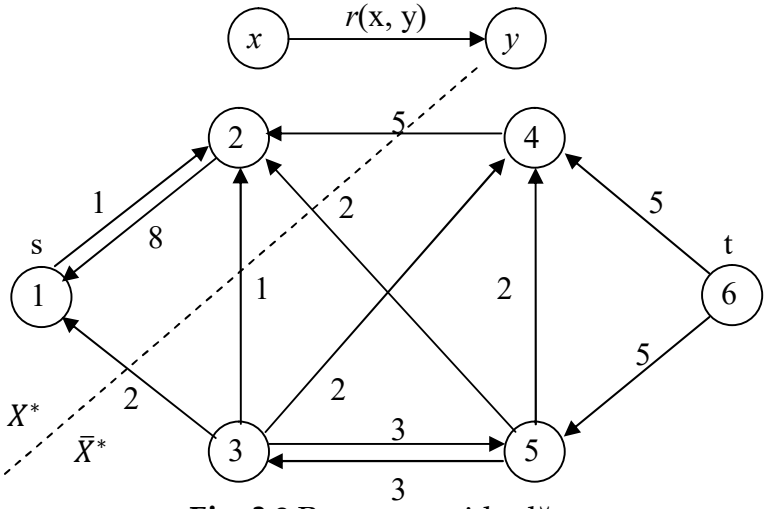

Fig. ?.8 Rețeaua reziduală

Cum nu mai există drum de la *s* la *t* in rețeaua G_f din fig. ?.8, înseamnă că execuția algoritmului se încheie. Se observă că mulțimea nodurilor accesibile în G_f din *s* este $X^* = \{1, 2\}$, iar $\bar{X}^* = \{3, 4, 5, 6\}$, mulțimi care dau tăietura minimă în rețeaua *G*, conform cu teorema 3. Fluxul maxim și tăietura minimă în rețeaua *G* sunt prezentate în figura ?.9.

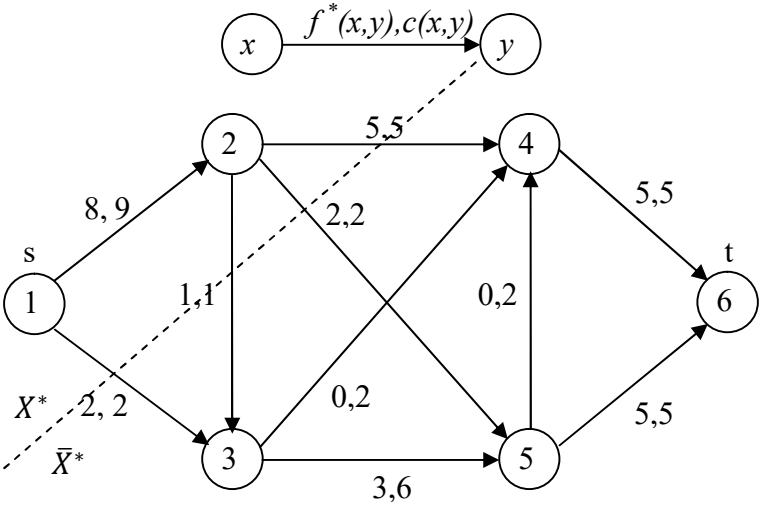

Fig. ?.9 Fluxul maxim și tăietura minimă

De exemplu, pentru arcul *(1, 2)* din rețeaua *G* fluxul este *f(1, 2) = r(2, 1) = 8*. În mod similar se obține valoarea fluxului maxim pentru fiecare arc al rețelei *G*.

Valoarea fluxului maxim este $v(f^*) = f(1, 2) + f(1, 3) = f(4, 6) + f(5, 6) = 10$. De asemenea, capacitatea tăieturii minime este $c[X^*, \bar{X}^*] = c(1,2) + c(2,3) + c(2,4) + c(2,5) = 10$, ceea ce verifică, evident, rezultatul din teorema 2.

?.5 Flux de cost minim

Problema fluxului de cost minim constă în a determina un flux admisibil cu o valoare dată $v > 0$ având costul cel mai mic. Pentru aceasta introducem *funcția cost* $b : E \to R_+$. Prin $b(x, y)$ notăm costul transportului a unei unități de flux de la nodul x la nodul y pe arcul (x, y). Cu alte cuvinte, dacă avem $f(x, y)$ unități de flux pe arcul (x, y), costul transportului lor de este $b(x, y) f(x, y)$. Matematic, problema fluxului de cost minim într-o rețea s-t se scrie astfel:

$$(?.13)\quad \begin{cases} \min b(f) = \sum_{(x,y)\in E} b(x,y)\cdot f(x,y) \\ \sum\limits_{\substack{y\in V \\ (x,y)\in E}} f(x,y) - \sum\limits_{\substack{y\in V \\ (y,x)\in E}} f(y,x) = 0, \forall x \notin \{s,t\} \\ \sum\limits_{\substack{y\in V \\ (s,y)\in E}} f(s,y) - \sum\limits_{\substack{y\in V \\ (y,s)\in E}} f(y,s) = v \\ \sum\limits_{\substack{y\in V \\ (y,t)\in E}} f(y,t) - \sum\limits_{\substack{y\in V \\ (t,y)\in E}} f(t,y) = v \\ 0 \le f(x,y) \le c(x,y), \forall (x,y) \in E \end{cases}$$

Vom presupune că rețeaua G este antisimetrică. În rețeaua reziduală, fiecărui arc $(x, y) \in E_f$ vom atașa costul $b_f(x, y)$ astfel:

1. Dacă $(x, y) \in E$, atunci $b_f(x, y) = b(x, y)$.
2. Dacă $(y, x) \in E$, atunci $b_f(x, y) = -b(y, x)$.

Următoarea teoremă stă la baza algoritmului pe care îl vom prezenta în cele urmează pentru rezolvarea problemei fluxului de cost minim.

Teorema ?.7 (Condiție optimalitate circuit negativ):
Un flux f de valoare v este soluție optimă pentru (?.13) dacă și numai dacă în rețeaua reziduală G_f nu există circuit de cost negativ.

Demonstrație: Se găsește în [1].
□

Algoritmul de eliminare a circuitelor negative

```
Se porneşte cu un flux admisibil f de
valoare v în reţeaua G;
Se construieşte reţeaua reziduală Gf;
Cât timp ∃ circuit negativ în Gf execută
      Se găseşte circuit negativ H în Gf;
      r(H):=min{r(x,y)|(x,y)∈H};
      Se măreşte fluxul f pe circuitul H;
      Se actualizează Gf;
sfârşit cât timp;
```

Un flux admisibil inițial f de valoare v se poate găsi introducând două noduri s' și t' împreună cu arcele (s', s) și (t, t'). Pentru aceste două arce capacitățile le stabilim ca fiind $c(s', s) = c(t, t') = v$, iar costurile nu ne interesează. În rețeaua modificată nodurile sursă și stoc

vor fi s' și, respectiv, t'. Evident, în noua rețea fluxul nu va putea depăși valoarea v. În rețeaua modificată căutăm fluxul maxim care, dacă are valoarea v, va fi flux adimisibil de în rețeaua inițială, iar dacă fluxul maxim va avea valoarea mai mică decât v, înseamnă că problema fluxului de cost minim nu are soluție.

Un circuit de cost negativ se poate determina adaptând algoritmului Bellman-Ford pentru drum minim.

Mărirea fluxului de-a lungul circuitului H este asemănătoare cu mărirea fluxului de-a lungul drumului D din algoritmul Ford-Fulkerson. Astfel, pentru arcele (x, y) care nu aparțin lui H, fluxul și rețeaua reziduală rămân nemodificate, iar mărirea de flux se efectuează pentru fiecare arc (x, y) al circuitului H din rețeaua reziduală după cum urmează:

1. Dacă (x, y) este arc în rețeaua G, atunci fluxul pe acest arc crește cu valoarea $r(H)$, adică $f(x, y) := f(x, y) + r(H)$.
2. Dacă (y, x) este arc în rețeaua G, atunci fluxul pe arcul (y, x) scade cu valoarea $r(H)$, adică $f(y, x) := f(y, x) – r(H)$.

În rețeaua reziduală pe fiecare arc (x, y) al circuitului H capacitatea reziduală scade cu valoarea $r(H)$, adică: $r(x, y) := r(x, y) – r(H)$, iar pe arcul (y, x) crește cu valoarea $r(H)$, adică: $r(y, x) := r(y, x) + r(H)$. După

actualizare cel puțin un arc (x, y) din H va avea $r(x, y) = 0$ și, în consecință, arcul (x, y) va dispărea din G_f, iar circuitul H nu va mai exista.

Teorema ?.8 (Corectitudinea algoritmului de eliminare a circuite negative):
Dacă în rețeaua G capacitatea, costul şi fluxul inițial sunt cu valori întregi, atunci algoritmului de eliminare a circuitelor negative se termină în număr finit de pași și determină fluxul de cost minim în rețeaua G.

Demonstrație: La fiecare iterație a ciclului "cât timp" costul fluxului scade cu capacitatea reziduală r(H) a circuitului înmulțită cu costul circuitului, adică scade cu cel puțin o unitate. În consecinnță, după cel mult $b(f)$ iterații execuția se încheie, unde f este fluxul inițial. Din ciclul "cât timp" se iese atunci când nu mai există circuite de cost negativ, ceea ce înseamnă că la încheierea algoritmului, fluxul f este de cost minim în rețeaua G, conform teoremei ?.7.
□

Teorema ?.9 (Complexitatea algoritmului de eliminare a circuitelor negative):
Dacă în rețeaua G capacitatea, costul şi fluxul inițial sunt cu valori întregi, atunci complexitatea algoritmului de eliminare a circuitelor negative este $O(n \cdot m^2 \cdot B \cdot C)$, unde:

$$B = \max\{b(x,y) | (x,y) \in A\}.$$

și

$$C = \max\{c(x,y) | (x,y) \in A\}$$

Demonstrație: Un circuit de cost negativ poate fi determinat folosind o adaptare a algoritmului Bellman-Ford pentru drum minim, care are o complexitate *O(n·m)*. Ciclul "cât timp" are cel mult *b(f)* iterații, așa cum am văzut în demosntrația teoremei ?.8. De asemenea, avem:

$$b(f) = \sum_{(x,y)\in E} b(x,y)\cdot f(x,y) \leq \sum_{(x,y)\in E} b(x,y)\cdot c(x,y) \leq m \cdot B \cdot C$$

Așadar, complexitatea întregului algoritm este $O(n\cdot m\cdot b(f)) = O(n\cdot m^2\cdot B\cdot C)$.

□

În continuare vom prezenta un tabel cu algoritmii cunoscuți în prezent pentru flux de cost minim, iar pentru fiecare dintre aceștia este prezentată complexitatea.

Algoritm	Complexitate
Algoritmul de eliminare a circuitelor negative	$O(n\cdot m^2\cdot B\cdot C)$
Algoritmul de eliminare a	$O(n\cdot m^2\cdot \log(n))$

circuitelor negative de cost mediu minim	
Algoritmul drumului cel mai scurt	$O(n^3 \cdot m)$
Algoritmul prim-dual	$O(n^2 \cdot m \cdot min\{B, C\})$
Algoritmul „out-of-kilter"	$O(m^2 \cdot C + m \cdot n \cdot C \cdot log(n))$

În continuare vom lua un exemplu pentru a ilustra funcționalitatea algoritmului de eliminare a circuitelor negative. Rețeaua G și fluxul admisibil inițial f cu valoarea $v = 6$ sunt date în figura ?.10. Costul fluxului inițial este $b(f) = 4 \cdot 4 + 2 \cdot 5 + 1 \cdot 3 + 3 \cdot 3 + 0 \cdot 4 + 1 \cdot 4 + 2 \cdot 5 + 4 \cdot 4 + 0 \cdot 2 + 2 \cdot 5 = 78$.

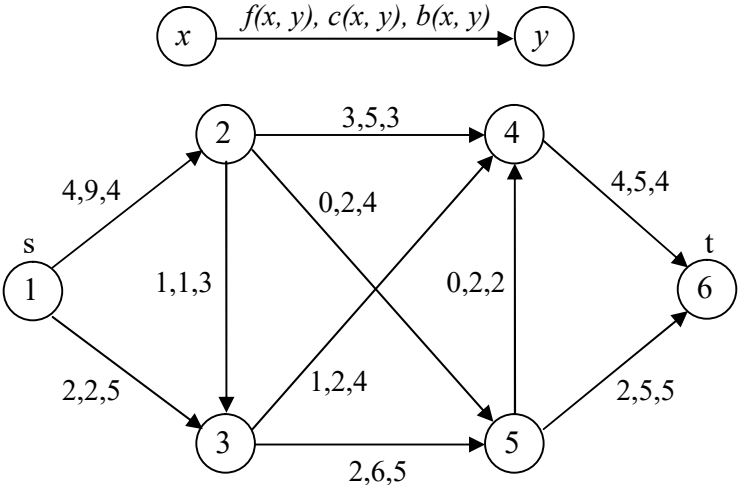

Fig. ?.10 Rețeaua G și fluxul admisibil f

Rețeaua reziduală atașată rețelei G și fluxului f este prezentată în fig. ?.11.

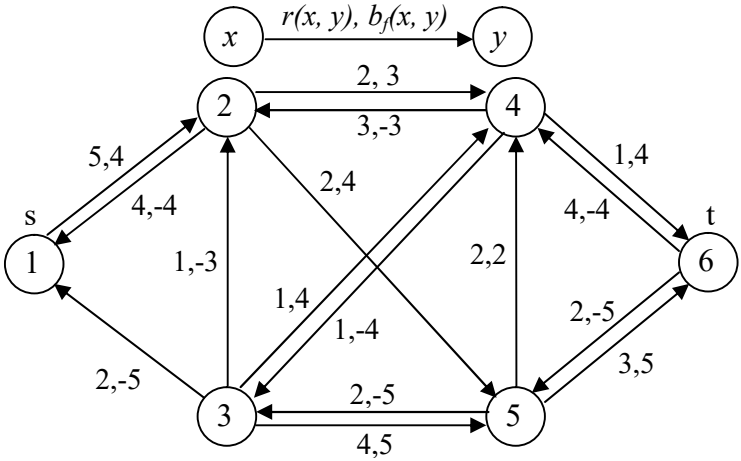

Fig. ?.11 Rețeaua reziduală G_f

Iterația 1: Considerăm circuitul negativ H_1 = (1,2,5,3,1) cu $b(H_1) = 4+4-5-5 = -2 < 0$ și r(H_1) = min(5,2,2,2) = 2.

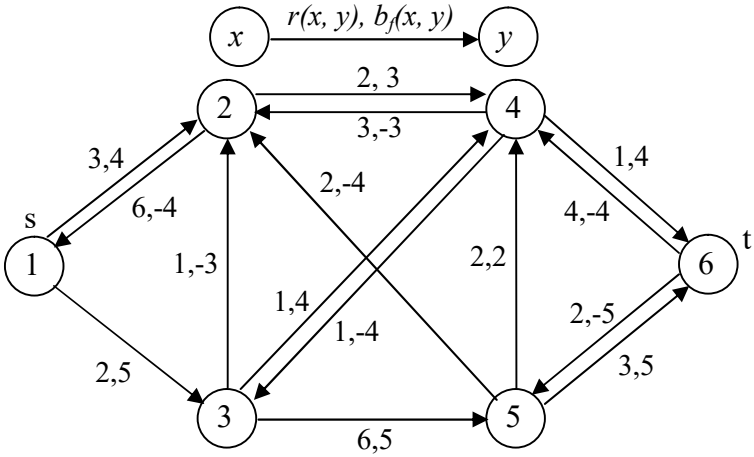

Fig. ?.12 Rețeaua reziduală G_f

Iterația 2: Considerăm circuitul H_2 = (2,4,6,5,2) cu $b(H_2)$ = *-2* și r(H_2) = 1.

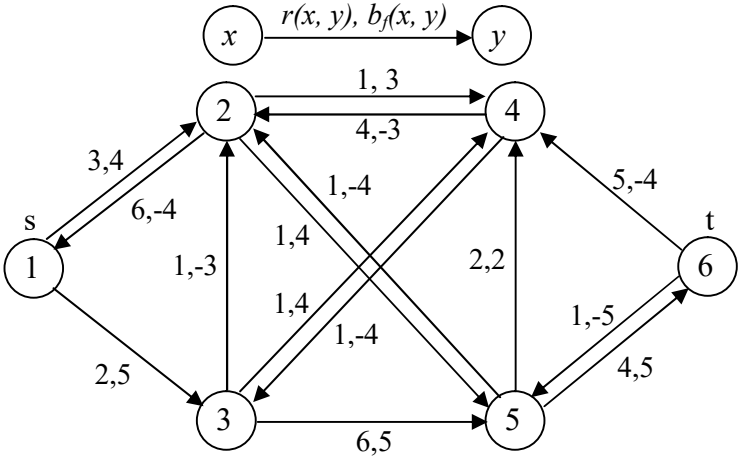

Fig. ?.13 Rețeaua reziduală G_f

Iterația 3: Considerăm circuitul H_3 = (2,4,3,2) cu $b(H_3)$ = *-4* și r(H_3) = 1.

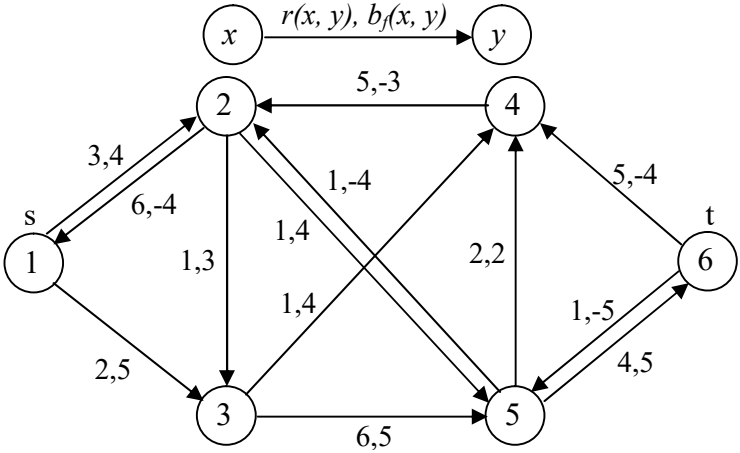

Fig. ?.14 Rețeaua reziduală G_f

Se observă că nu mai există circuite de cost negativ în rețeaua reziduală în figura ?.14. Așadar, execuția algoritmului se încheie. Fluxul de cost minim este prezentat în figura ?.15.

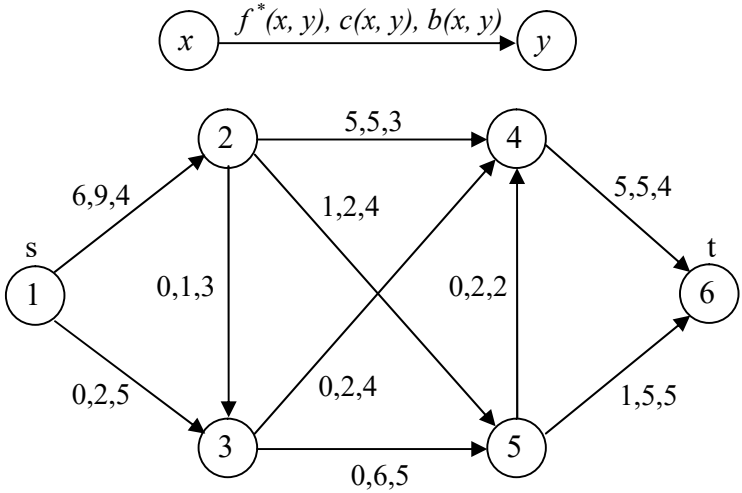

Fig. ?.15 Fluxul de cost minin f^* în G

Costul fluxului a scăzut în cele trei iterații cu $-b(H_1) \cdot r(H_1) - b(H_2) \cdot r(H_2) - b(H_3) \cdot r(H_3) = 2 \cdot 2 + 2 \cdot 1 + 4 \cdot 1 = 10$ unități ajungând la valoarea $b(f^*) = 78 - 10 = 68$, unde f^* este fluxul optim (de cost minim) construit de algoritm. Într-adevăr, după cum se poate vedea din figura ?.15, avem: $b(f^*) = 6 \cdot 4 + 0 \cdot 5 + 0 \cdot 3 + 5 \cdot 3 + 1 \cdot 4 + 0 \cdot 4 + 0 \cdot 5 + 5 \cdot 4 + 0 \cdot 2 + 1 \cdot 5 = 68$.

?.6 Exerciții propuse

1. Implementați algoritmul Ford-Fulkerson pentru determinarea fluxului maxim și a tăieturii minime.
2. Implementați algoritmul de eliminare a circuitelor de cost negativ pentru determinarea fluxului de cost minim pentru o valoare dată $v > 0$. Pentru determinarea unui flux admisibil inițial se va aplica Ford-Fulkerson conform indicațiilor de după algoritm.

Bibliografie

[1] R.K. Ahuja, T.L. Magnanti, J.B. Orlin, *Network Flows. Theory, Algorithms, and Applications*, Prentice Hall, N.J., 1993