Proiectarea Algoritmilor 2009-2010

Laborator 10

Flux maxim

1. Obiective laborator

- formalizarea notiunilor de retea de transport si flux in retea
- prezentarea unei metode de rezolvare a problemei de flux maxim
- analiza unei implementari a metodei oferite

2. Importanta - Aplicatii practice

Un graf orientat poate fi utilizat pentru modelarea unui proces de transport intr-o retea intre un producator s si un consumator t. Destinatia nu poate consuma mai mult decat se produce, iar cantitatea trimisa pe o cale nu poate depasi capacitatea sa de transport.

Retelele de transport pot modela curgerea lichidului in sisteme cu tevi, deplasarea pieselor pe benzi rulante, deplasarea curentului prin retele electrice, transmiterea informatiilor prin retele de comunicare etc.

3. Descrierea problemei – Flux maxim

O problema des intalnita intr-o retea de transport este cea a gasirii fluxului maxim posibil prin arcele retelei astfel incat:

1. sa nu fie depasite capacitatile arcelor
2. fluxul sa se conserve in drumul sau de la s la t

Definitie 3.1

O retea de transport este un graf orientat G=(V,E) cu proprietatile:

1. exista doua noduri speciale in V: s este nodul sursa (sau producatorul) si t este nodul terminal (sau consumatorul).
2. este definita o functie totala de capacitate $c: V \times V \to \mathbb{R}_+$ astfel incat:
    1. $c(u,v)=0$ daca $(u,v) \notin E$
    2. $c(u,v) \geq 0$ daca $(u,v) \in E$

1. pentru orice nod $v \in V \setminus \{s,t\}$ exista cel putin o cale $s \dashrightarrow v \dashrightarrow t$.

Definitie 3.2

Numim flux in reteaua $G=(V,E)$ o functie totala $f: V \times V \to \mathbb{R}$ cu proprietatile:

1. **Restrictie de capacitate:**

$$f(u,v) \leq c(u,v) \ , \ \forall (u,v) \in V$$

(fluxul printr-un arc nu poate depasi capacitatea acestuia

2. **Antisimetrie:**

$$f(u,v) = -f(v,u) \ , \ \forall u \in V, \forall v \in V$$

3. **Conservarea fluxului:**

$$\sum_{v \in V} f(u,v) = 0, \forall u \in V \setminus \{s,t\}$$

Un flux negativ de la u la v este unul virtual, el nu reprezinta un transport efectiv, ci doar sugereaza ca exista un transport fizic de la v la u (este o conventie asemanatoare cu cea facuta pentru intensitatile curentilor intr-o retea electrica). Ultima proprietate ne spune ca la trecerea printr-un nod fluxul se conserva: suma fluxurilor ce intra intr-un nod este 0(tinand cont de conventia de semn stabilita).

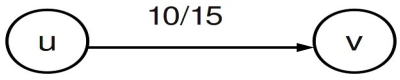
Numim ***capacitate reziduala*** a unui arc

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

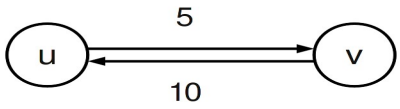
si o interpretam ca fiind contitatea de flux aditional care poate fi transportat de la u la v, fara a depasi capacitatea $c(u,v)$.

Exemplu:

muchie in graf:



muchii reziduale:



Daca avem arcul $(u,v)\in V$ cu $c(u,v)=15$ si $f(u,v)=10$ se pot transporta $c_f(u,v)=5$ unitati suplimentare fara a incalca restrictia de capacitate. Dar, conform definitiei, desi arcul $(v,u)\notin V$ vom avea totusi o capacitate reziduala $c_f(v,u)=c(v,u)-f(v,u)=0-(-10)=10$: as putea transporta 10 unitati in sens opus care sa le anuleze pe cele 10 ale fluxului direct pe muchia (u,v) .

Definitie 3.2

Fie o retea de flux $G=(V,E)$, iar f fluxul prin G. Numim ***retea reziduala*** a lui G, indusa de f, o retea de flux notata cu $G_f=(V,E_f)$, astfel incat

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

Este important de observat ca E_f si E pot fi disjuncte: un arc rezidual (u,v) apare in reteaua reziduala doar daca capacitatea sa este strict pozitiva (ceea ce nu implica existenta arcului in reteaua originala)

Un ***drum de ameliorare*** este o cale $(u_1,u_2,\ldots,u_k)$, unde $u_1=s$ si $u_k=t$, in graful rezidual cu $c_f(u_i,u_{i+1})>0,\ \forall i=\overline{1,k-1}$. Practic, un drum de ameliorare va reprezenta o cale in graf prin care se mai poate pompa flux aditional de la sursa la destinatie.

Asa cum era de intuit, ***capacitatea reziduala a unui drum de ameliorare*** p este cantitatea maxima de flux ce se poate transporta de-a lungul lui:

$$c_f(p)=min\{c_f(u,v)\mid (u,v)\in p\}$$

Acum ca am introdus notiunile necesare pentru formalizarea problemei de flux maxim intr-un graf, putem sa prezentam si cea mai utilizata metoda de rezolvare.

Algoritmul Ford-Fulkerson

Aceasta este o metoda iterativa de gasire a fluxului maxim intr-un graf care pleaca de la ideea: cat timp mai exista un drum de ameliorare(o cale de la sursa la destinatie) pot pompa pe aceasta cale un flux suplimentar egal cu capacitatea reziduala a caii.

Ford_Fulkerson
Input $G(V,E),\ s,\ t$
Output $|f_{max}|$

$f(u,v) \leftarrow 0,\ \forall (u,v) \in V \times V$
while $\exists\ a\ path\ p(s \dashrightarrow t)$ in G_f *such* that $c_f(u,v) > 0,\ \forall (u,v) \in p$
find $c_f(p) = min\{c_f(u,v) \mid c_f(u,v) \in p\}$
for-each $(u,v) \in p$
$f(u,v) \leftarrow f(u,v) + c_f(p)$
$f(v,u) \leftarrow -f(u,v)$
return $|f_{max}|$

Acest algoritm reprezinta mai mult un sablon de rezolvare pentru ca nu detaliaza modul in care se alege drumul de ameliorare din reteaua reziduala.

Complexitatea va fi $O(E \cdot f_{max})$ pentru ca in ciclul while putem gasi, in cel mai rau caz, doar cai care duc la cresterea fluxului cu doar o unitate la fiecare pas.

Ne punem acum problema daca acest algoritm este corect, daca la final vom avea cu adevarat fluxul maxim posibil prin graf. Corectitudinea algoritmului deriva imediat din teorema Flux maxim-taietura minima.

Mai intai, numim o ***taietura a unui graf*** o partitionare (S,T) a nodurilor sale cu proprietatea $s \in S\ si\ t \in T$.

Teorema flux maxim-taietura minima

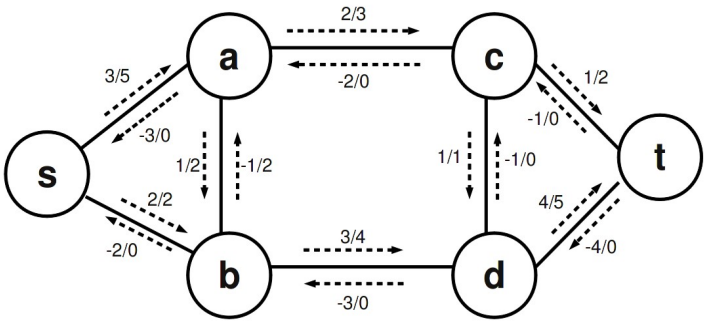
Pentru o retea de flux $G(V,E)$ urmatoarele afirmatii sunt echivalente:

1. f este fluxul maxim in G
2. Reteaua reziduala G_f nu contine drumuri de ameliorare
3. Exista o taietura (S,T) a lui *G* astfel incat fluxul net prin taietura este egal cu capacitatea acelei taieturi.

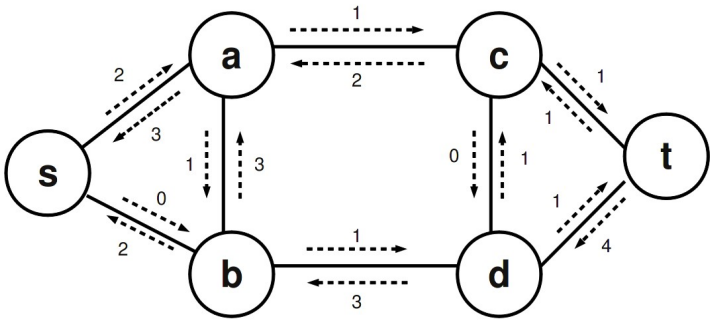
Obs: Prin orice taietura fluxul este egal cu cel maxim pentru ca nu exista o alta cale pe care ar putea ajunge flux de la sursa la destinatie si care sa nu treaca prin taietura (ar incalca tocmai definitia ei). Sau, altfel spus, valoarea unui flux intr-o retea este data de fluxul oricarei taieturi. Astfel, fluxul total va fi marginit de cea mai mica capacitate a unei taieturi. Daca este indeplinit punctul 3. al teoremei atunci stim ca acea taietura nu poate fi decat una de capacitate minima.

Exemplu:

Reteaua initiala:



Reteaua reziduala:



Observam ca desi muchia (d,c) are capacitate 0 in reteaua originala (ea nu ar putea transporta flux) in reteaua reziduala avem $c(d,c)=1$ ceea ce-i permite sa faca parte din drumul de ameliorare $p_1=(s,a,b,d,c,t)$ de capacitate $c_f(p_1)=1$, astfel ca adaugarea acestui flux suplimetar pe muchia (d,c) nu va duce la incalcarea restrictiei de capacitate. Sau am fi putut alege drumul $p_2=(s,a,c,t)$ cu $c_f(p_2)=1$ sau $p_3=(s,a,b,d,t)$ cu $c_f(p_3)=1$.

Performanta algoritmului tine de modul in care va fi ales drumul de ameliorare, se poate intampla ca pentru $|f_{max}|$ de valoarea mare o alegere nepotrivita sa duca la timpi de executie foarte mari.

Implementarea Edmonds-Karp

Asa cum am vazut, algoritmul Ford-Fulkerson nu defineste o metoda de alegere a drumului de ameliorare pe baza caruia se modifica fluxul in graf. Implementarea Edmonds-Karp alege intotdeauna cea mai scurta cale folosind o cautare in latime in graful rezidual unde fiecare arc are ponderea 1. Se poate demonstra ca lungimea cailor gasite astfel creste monoton cu fiecare noua ameliorare.

Edmonds-Karp

Input $G(V,E),\ s,\ t$

Output $|f_{max}|$

$f(u,v) \leftarrow 0,\ \forall (u,v) \in V \times V$

while true

$p(s \dashrightarrow t) \leftarrow BFS(G_f, s, t)$

if not $\exists\ p(s \dashrightarrow t)$

break;

find $c_f(p) = min\{c_f(u,v) \mid c_f(u,v) \in p\}$

for-each $(u,v) \in p$

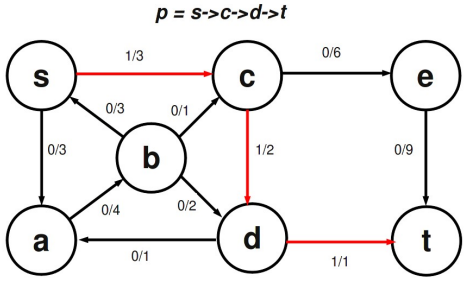
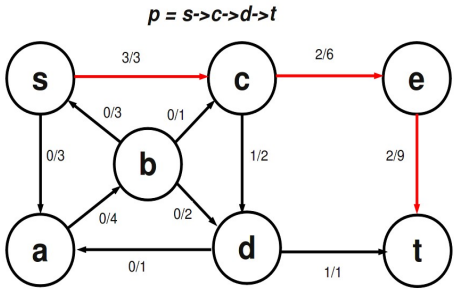
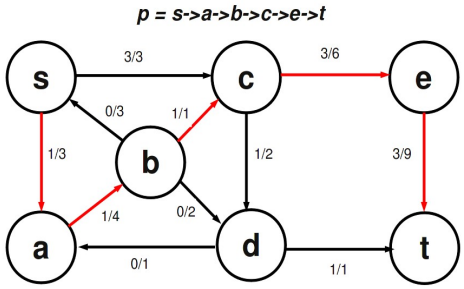
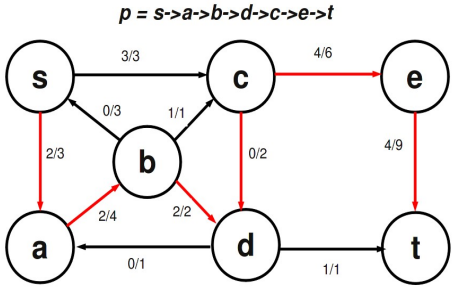
$f(u,v) \leftarrow f(u,v) + c_f(p)$

$f(v,u) \leftarrow -f(u,v)$

return $|f_{max}|$

Plecand de la ideea ca drumurile de ameliorare gasite au lungimi din ce in ce mai mari se poate arata ca in aceasta implementare fluxul se mareste de cel mult $O(V \cdot E)$ ori([1]->pag.513). Complexitatea algoritmului va fi $O(V \cdot E^2)$.

Sa luam un exemplu de rulare al acestui algoritm. Vom considera starea retelei dupa ce a fost gasita prima cale de pompare flux(initial toate arcele sunt etichetate cu 0/0 conform notatiei stabilite):

Capacitatea reziduala a caii de ameliorare	Graful si calea de ameliorare gasita
$c_f(p)=min(c_f(s,c),c_f(c,d),c_f(d,t))=$ $=min(3-0,2-0,1-0)=min(3,2,1)=1$	
$c_f(p)=min(c_f(s,c),c_f(c,e),c_f(e,t))=$ $=min(3-1,6-0,9-0)=min(2,6,9)=2$	
$c_f(p)=min(c_f(s,a),c_f(a,b),c_f(b,c),$ $c_f(c,e),c_f(e,t))=$ $=min(3-0,4-0,1-0,6-2,9-2)=$ $=min(3,4,1,4,7)=1$	
$c_f(p)=min(c_f(s,a),c_f(a,b),c_f(b,d),$ $c_f(d,c),c_f(c,e),c_f(e,t))=$ $=min(3-1,4-1,2-0,0-(-1),6-3,9-3)=$ $=min(2,3,2,1,3,6)=1$	

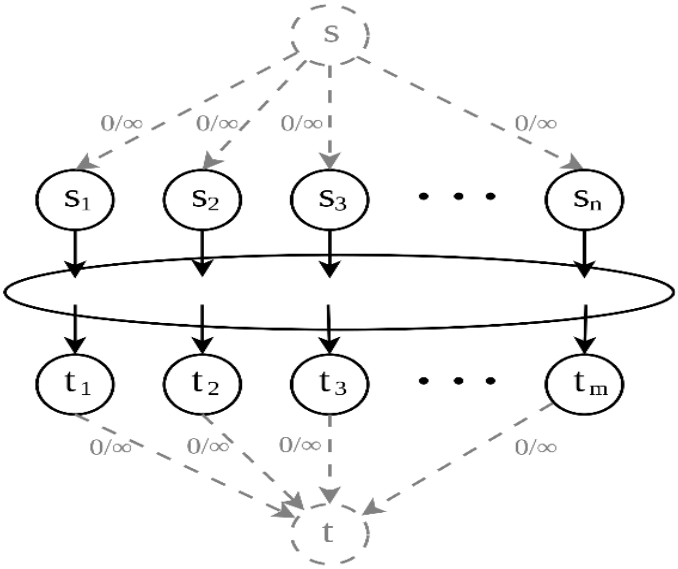
Obs:

In cazul ultimului drum de ameliorare gasit in exemplul dat, se observa ca desi nu exista muchia *(d,c)* se simuleaza un flux pe aceasta printr-unul negativ in sens opus.

4. Variatii ale problemei clasice

In retelele clasice studiate pana acum aveam o sursa unica care putea produce oricat, destinatia unica consuma oricat, orice nod intermediar conserva fluxul, iar singura constrangere a muchiilor era limitarea superioara a fluxului prin capacitate. Sa vedem cum putem generaliza aceste conditii si daca retelele obtinute ar putea fi reduse la una clasica.

1) Surse si destinatii multiple



Se observa ca putem reduce acest graf la cel cunoscut prin adaugarea unei meta-surse legata de sursele mici prin muchii de capacitate nelimitata si analog un meta-terminal cu aceleasi proprietati. Global comportamentul retelei de flux nu se va schimba.

2) Retea cu noduri ce nu conserva fluxul

Spre deosebire de s si t care produc/consuma oricat, un nod intermediar ar putea produce sau consuma o cantitate constanta de flux la trecerea prin el. In acest caz vom avea un invariant la nivel de nod care ia forma:

$f_{\text{in}} - f_{\text{out}} = d_i$, unde d_i este catitatea produsa suplimentar (> 0) sau solicitata (< 0) de un nod. Am putea transforma egalitatea in:

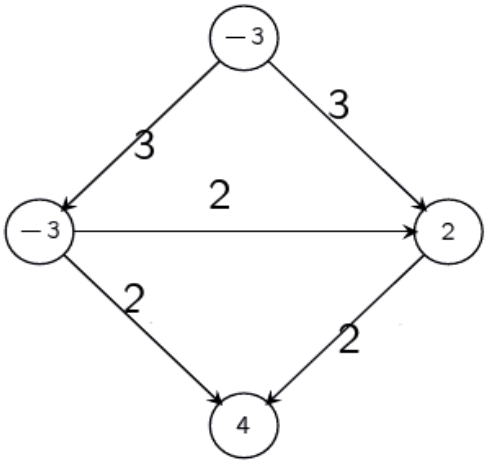
$(f_{\text{in}} + |d_i|) - f_{out} = 0$, daca $d_i < 0$

$f_{\text{in}} - (f_{out} + |d_i|) = 0$, daca $d_i > 0$

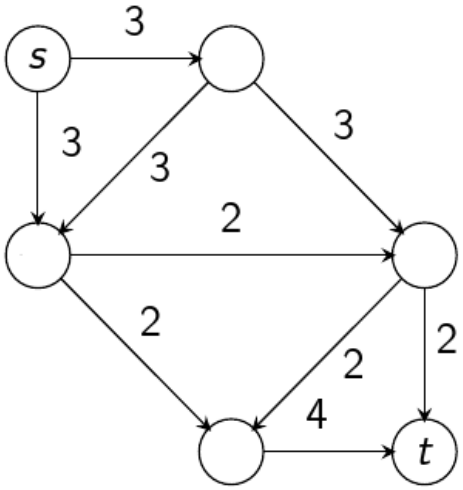
Altfel spus, un nod ce consuma flux poate fi transformat intr-unul ce conserva fluxul si are un in-arc aditional de capacitate $|d_i|$, iar unul ce produce flux va avea un out-arc de aceeasi capacitate.

La nivelul unei retele intregi se adauga un nod sursa cu muchii catre toate nodurile ce consumau flux si un nod destinatie dinspre toate nodurile ce produceau flux.

Exemplu:



se va transforma in:



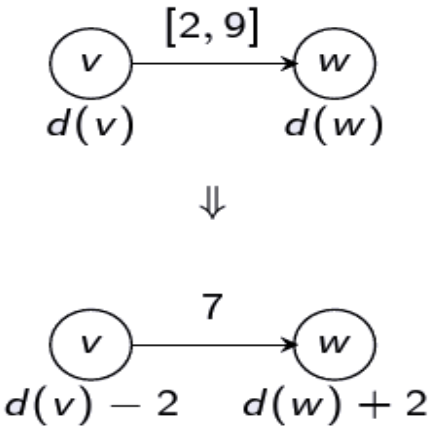
3) Retea cu limite inferioare de capacitate

Exista cazuri in care as vrea ca datele de pe muchiile retelei sa faca parte dintr-un anumit interval [inf, sup]. Pe o astfel de muchie fluxul trebuie sa respecte inegalitatea

$$\inf \le f \le \sup$$

Plecand tot de la conditiile de conservare la nivel de nod putem translata intervalul [inf, sup] in [0, sup-inf] si sa consideram ca nodul sursa a consumat inf unitati de flux iar nodul destinatie a produs inf unitati. Din exterior entitatea alcatuita din doua noduri si o muchie este vazuta ca actionand in acelasi fel asupra fluxului ce o traverseaza.

Iata un exemplu de transformare:



Bineinteles ca toate aceste transformari pot fi combinate pentru a se ajunge la reteaua de flux clasica.

5. Concluzii si observatii

Laboratorul de fata s-a vrut a fi doar o introducere in domeniul fluxurilor intr-un graf – modalitate de a reprezenta probleme de circulatie a materialelor atat de frecvent intalnite. In [1] si [2] gasiti si alti algoritmi interesanti impreuna cu studiul complexitatii lor (algoritmul de pompare preflux, algoritmul „mutare-in-fata”). Spre exemplu, cel mai bun algoritm in prezent pentru cuplajul bipartit maxim se executa in $O\left(\sqrt{V}\cdot E\right)$.

6. Referinte

[1] – Introducere in algoritmi, Thomas H. Cormen, Charles E. Leiserson, Ronald R. Rivest – Capitolul VI Algoritmi pe grafuri: Flux maxim

[2] – Introducere in analiza algoritmilor, Cristian A. Giumale – Cap. V Algoritmi pr grafuri: Fluxuri maxime intr-un graf

[3] - Un articol de la MIT : A Labeling Algorithm for the maximum flow network problem

http://74.125.77.132/search?q=cache:HuQ-I9RrWAkJ:web.mit.edu/15.053/www/AMP-Appendix-C.pdf+maximum+flow+mit&cd=1&hl=ro&ct=clnk&gl=ro&client=firefox-a

[4] - Resurse wiki – Edmonds Karp

http://en.wikipedia.org/wiki/Edmonds%E2%80%93Karp_algorithm

http://en.wikipedia.org/wiki/Max-flow_min-cut_theorem

[5] – Aplicatii flux maxim

http://www.cs.princeton.edu/~wayne/cs423/lectures/max-flow-applications-4up.pdf

Responsabil laborator: Andra Iacov (andra.iacov@gmail.com)

Proiectarea Algoritmilor 2009-2010

Laborator 10

Aplicaţii

Se considera o retea de flux G(V, E) cu nodul sursa s si destinatia t.

1. 'Flux maxim'

Sa se gaseasca fluxul maxim ce poate fi suportat de o retea plecand din sursa s si sa se afiseze caile urmate de acesta impreuna cu cantitatea aferenta.

Se va folosi algoritmul lui Edmonds-Karp.

2. 'Taietura minimala'

Folosind rezultatele obtinute la problema anterioara aflati o taietura minimala a grafului. Taietura va fi afisata sub forma a doua multimi de noduri.

3. 'Cai disjuncte'

Spunem ca doua cai sunt disjuncte daca ele nu au nicio muchie in comun. Afisati toate caile disjuncte ale unui graf.

Obs: Pentru problemele anterioare fisierul de intrare contine pe prima linie numar noduri n si muchii m din graf, pe a doua nodul sursa s si destinatia t, iar apoi pe m linii muchiile sub forma

node1 node2 cost_muchie.

4. **'Aproximarea unei matrici'**

Se da o matrice $D=\left(d_{ij}\right)\in M_{n,m}\left(\mathbb{R}\right)$ *. Vom borda aceasta matrice adaugand o ultima linie suplimentara* a_n *ce contine suma elementelor de pe coloane si o ultima coloana* b_m *cu suma elementelor de pe linii.*

Exemplu:

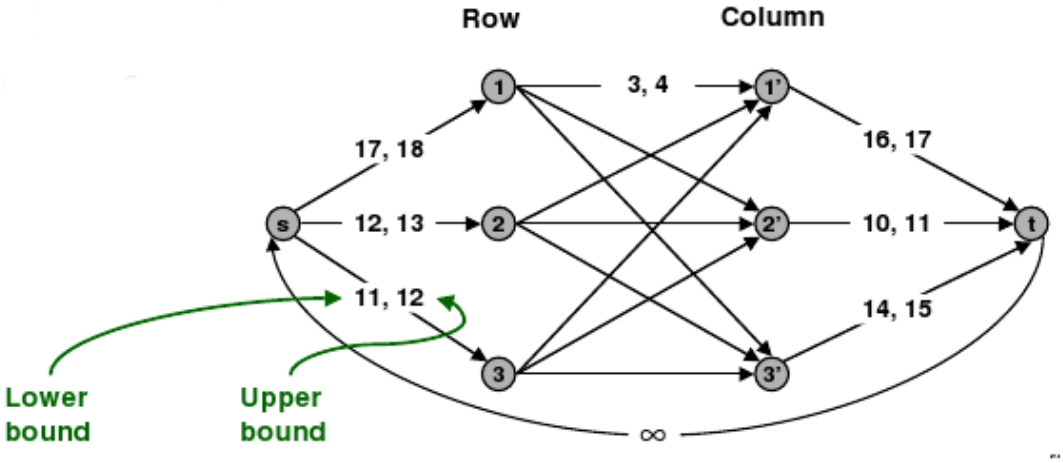
3.14	6.8	7.3	17.24
9.6	2.4	0.7	12.7
3.6	1.2	6.5	11.3
16.34	10.4	14.5	

Sa se rotunjeasca toate elementele d_{ij}, a_i, b_j *la un intreg (ori prin adaos, ori prin lipsa) astfel incat sumele pe linii si coloane sa se pastreze.*

Exemplu posibila rotunjire:

3	7	7	17
10	2	1	13
3	1	7	11
16	10	15	

Problema ar putea fi modelata folosind un graf astfel:



Obs: *pentru fiecare linie si coloana se adauga cate un nod. Pe muchia dintre nod i si nod j se afla cei doi intregi intre care e cuprins elementul* d_{ij} .