CURSUL 1

CAPITOLUL 1
TEORIA GRAFURILOR.
OPTIMIZĂRI ÎN REŢELE DE TRANSPORT ŞI DISTRIBUŢIE

NOTĂ: Suportul de curs al Capitolelor 1, 2 şi 3 are la bază lucrările:

- Ciobanu, Gh., Nica, V., Mustaţă, Fl., Mărăcine, V. Mitrut, D. (2002), ***Cercetări operaţionale. Optimizări în reţele. Teorie şi aplicatii economice***, Editura MATRIX ROM, Bucureşti;
- Ciobanu, Gh., Nica, V., Mustaţă Fl., Mărăcine, V. (1996), ***Cercetări operaţionale cu aplicaţii în economie. Teoria grafurilor şi Analiza drumului critic***, Editura MATRIX-ROM, Bucureşti

1.1. Modelarea problemelor de transport şi distribuţie

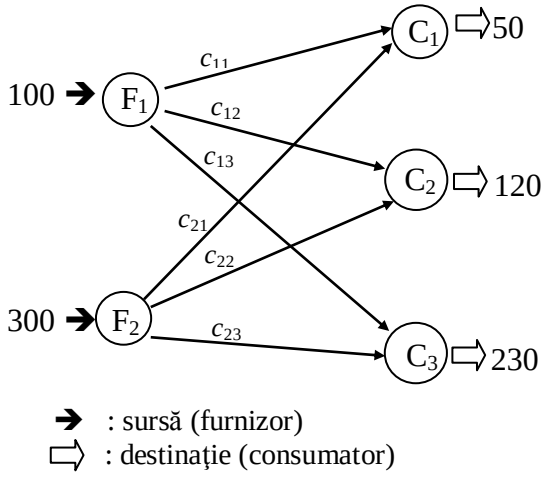
Într-o mare varietate de contexte se pune problema *deplasării* unei cantităţi Q ce poate fi materie, energie, informaţie, etc. din unele locuri numite *surse* în alte locuri numite *destinaţii*, această deplasare realizându-se pe anumite *rute de legătură*. Unităţile indivizibile ale cantităţii Q care se deplasează de-a lungul rutelor se vor numi *unităţi de flux*.

O clasificare a problemelor de transport şi distribuţie

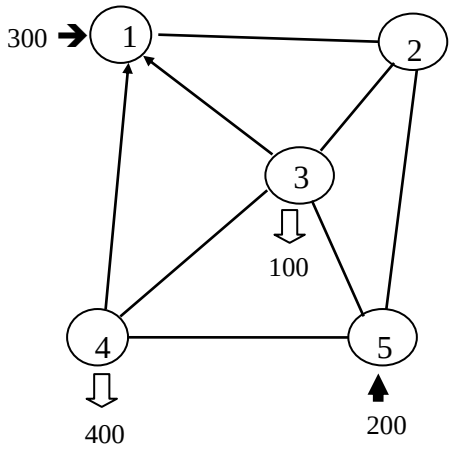
Pentru Cercetarea Operaţională, problema enunţată va prezenta interes numai dacă respectă următoarele ipoteze:

a) cel puţin o sursă poate aproviziona mai multe destinaţii şi cel puţin o destinaţie poate primi unităţi de flux de la mai multe surse.

Rutele de legătură pot avea şi alte puncte comune în afara surselor şi destinaţiilor, numite *puncte intermediare* sau *de tranzit*. Nu sunt excluse legăturile directe între surse sau între destinaţii. În principiu, orice rută poate fi parcursă în ambele sensuri, dar pot exista şi rute cu sens unic.



a)



b)

Figura 1.1.

Ansamblul surselor, destinaţiilor, al punctelor intermediare şi al rutelor de legătură se va numi ***reţea de transport***; el se identifică cu un *graf neorientat sau parţial orientat* ca în figura 1.1.

b) Unele rute de legătură pot avea limitări superioare şi / sau inferioare pentru volumul unităţilor de flux ce se deplasează într-un sens sau altul. Aceste limitări poartă numele de ***capacităţi*** (inferioare, respectiv superioare). În continuare, vom avea în vedere numai cazul în care toate capacităţile inferioare sunt egale cu zero, capacităţile superioare fiind exprimate prin numere pozitive.

c) Există un cost al deplasării unei unităţi de flux de la un punct al reţelei la altul, cost care poate fi exprimat în *bani, timp* sau *distanţă*. Sunt situaţii în care acest cost poate semnifica *profitul* obţinut de pe urma deplasării. Pe aceeaşi rută, costurile şi capacităţile pot fi diferite în funcţie de sensul de parcurgere al rutei.

Ipoteza a) va fi întotdeauna presupusă în timp ce ipotezele b) şi c) pot fiinţa separat sau simultan.

1) În prezenţa ipotezei c) şi absenţa condiţiei b) se pune problema deplasării cantităţii de flux Q de la surse la destinaţii *la un cost total minim.* Dacă sursele sunt în legătură directă cu destinaţiile obţinem ***problema clasică de transport***, care va face obiectul Capitolului 4 al cursului. Cazul general, în care exisă şi puncte intermediare, este cunoscut sub numele de *problema transferului* şi el nu face obiectul cursului de faţă. În cazul particular al unei singure surse *s*, al unei singure destinaţii *t* şi a unei singure unităţi de flux se obţine ***problema drumului de cost minim de la s la t,*** problemă de care ne vom ocupa în secţiunea 1.4 a acestui capitol.

2) În prezenţa ipotezei b) şi absenţa ipotezei c) se pune problema dacă reţeaua, ale cărei rute sunt *capacitate, este capabilă să permită acoperirea integrală a cererilor în punctele de destinaţie.* Pentru aceasta, se va rezolva problema determinării volumului *maxim* Q* de unităţi de flux ce pot fi deplasate de la surse la destinaţii. Dacă Q* < Q vor exista destinaţii a căror cerere este acoperită doar în parte şi atunci se ridică problema *măririi capacităţii de transfer a reţelei.* Am descris succint ***problema fluxului maxim,*** problemă ce va fi abordată în cadrul cursului de Cercetări Operaţionale din anul 3.

3) În prezenţa simultană a ipotezelor b) şi c) se pune *problema satisfacerii cererilor în punctele de destinaţie la un cost de transport minim.* Ca şi în cazul precedent vom avea în vedere o problemă modificată: vom determina mai întâi cantitatea maximă de flux ce poate fi deplasată de la surse la destinaţii şi apoi modul de organizare al deplasării astfel încât costul operaţiei să fie minim. Aceasta este ***problema fluxului (maxim) de cost minim,*** problemă abordată, de asemenea, în cadrul cursului de Cercetări Operaţionale din anul 3.

În secţiunile următoare vom trata două probleme de optimizare în reţele de transport: problema determinării unui ***arbore maximal (de acoperire) de valoare (cost) minim*** şi ***problema drumului de cost minim de la s la t*** .

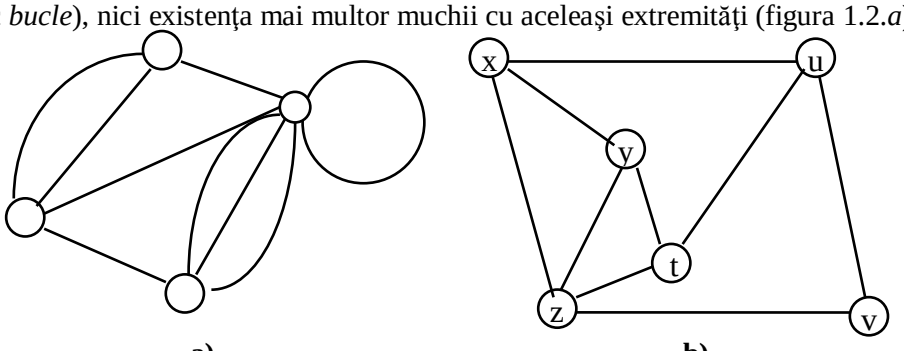
1.2 Concepte utilizate în teoria grafurilor

În secţiunea 1.1 am vizualizat elementele unei reţele de transport printr-un desen compus din puncte şi arce care unesc unele din aceste puncte (vezi figura 1.1). Un asemenea desen constituie forma uzuală de prezentare a unui concept deosebit de important, atât în sine cât şi pentru studiul unei impresionante varietăţi de probleme din cele mai diverse domenii, domeniul economic fiind prioritar.

Am socotit deci necesar să includem aici câteva elemente privitoare la conceptul de ***graf,*** căci despre el este vorba. Aceste elemente vor fi foarte utile pentru înţelegerea consideraţiilor teoretice dezvoltate în legătură cu rezolvarea problemei de transport şi de asemenea constituie cadrul natural de abordare şi a celorlalte probleme amintite în clasificarea dată în secţiunea precedentă.

Un ***graf*** este un cuplu G = (V, E) format dintr-o mulţime *nevidă* V, ale cărei elemente se numesc *vârfuri* sau *noduri* şi o mulţime E de elemente, zise *muchii*, cu proprietatea că fiecărei muchii $e \in E$ îi sunt asociate două noduri $x, y \in V$, nu neapărat distincte, numite *extremităţile* muchiei *e*. Un *subgraf* al grafului G = (V, E) este un graf G' = (V', E') în care $V' \subseteq V$, $E' \subseteq E$ şi orice muchie $e \in E'$ are aceleaşi extremităţi atât în G' cât şi în G.

După cum se vede, definiţia generală nu exclude existenţa muchiilor cu o singură extremitate (aceste muchii se numesc *bucle*), nici existenţa mai multor muchii cu aceleaşi extremităţi (figura 1.2.*a*)



a) **b)**

Figura 1.2.

Graful G se va numi *simplu* dacă nu are bucle şi oricare două noduri sunt extremităţi pentru cel mult o muchie. Vom spune că G este *finit* dacă vârfurile şi muchiile sale sunt în număr finit.

*În continuare, vom avea în vedere în exclusivitate grafuri **finite şi simple** aşa cum este cel reprezentat grafic în figura 1.2.b).*

Fie $e = \{x, y\}$ o muchie a grafului G = (V, E); extremităţile sale pot fi ordonate în două moduri: (x, y) şi (y, x). Cele două perechi se numesc *rute orientate* sau *arce* generate de muchia subiacentă *e*; spunem că arcul (x, y) are *extremitatea iniţială x* şi *extremitatea finală y* şi că (y, x) este arcul *opus* lui (x, y). A *orienta* muchia $\{x, y\}$ înseamnă a *alege* unul din arcele (x, y) sau (y, x); dacă a fost ales arcul (x, y) vom spune că (x, y) este un arc *permis* şi că arcul (y, x) este *blocat*. Dacă o asemenea alegere nu a fost făcută vom spune că muchia $\{x, y\}$ este *neorientată*. În acest caz, convenim ca ambele arce (x, y) şi (y, x), generate de muchia $\{x, y\}$, să fie considerate permise. În fine, a da o *orientare* în graful G înseamnă a orienta unele din muchiile sale; orientarea poate fi *totală* sau *parţială* după cum toate muchiile grafului au fost orientate sau numai o parte din ele. Este clar că în acelaşi graf G pot fi date mai multe orientări; dacă G are *m* muchii, atunci există 2^m *orientări totale* diferite ale acestuia!

Un *graf orientat* (*parţial orientat, neorientat*) este un graf în care s-a dat o orientare totală (o orientare parţială, respectiv nu s-a dat nici o orientare); de exemplu, graful din figura 1.1.*a*) este (total) orientat iar cel din figura 1.1.*b*) numai parţial. Graful din figura 1.2.*b*) este neorientat.

În unele situaţii este util să se pună în evidenţă arcele permise ale unui graf (parţial) orientat; realizarea grafică, intuitivă a acestei operaţii este făcută în figura 1.3.

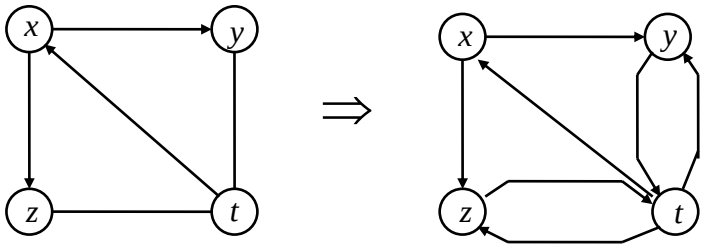


Figura 1.3.

Se constată uşor că dacă G este (total) neorientat, numărul arcelor permise este de două ori mai mare decât numărul muchiilor din G.

În continuare vom introduce o serie de noţiuni frecvent utilizate în teoria grafurilor. Unele din ele se referă la muchii şi de aceea se numesc "generic" *concepte neorientate* altele se referă la rutele orientate permise, drept care se mai numesc şi *concepte orientate*. Vom considera un graf (*finit, simplu*) G = (V,E), în care (eventual) s-a dat o orientare pe unele din muchii (posibil pe toate).

Un ***lanţ*** în graful G este o succesiune de noduri $\lambda = (x_0, x_1, \ldots, x_{p-1}, x_p)$ cu proprietatea că $\{x_0, x_1\}$, $\{x_1, x_2\}$, ..., $\{x_{p-1}, x_p\}$ sunt muchii în G. Vom spune că x_0 şi x_p sunt *extremităţile* lanţului λ şi că λ "*trece*" prin nodurile *intermediare* $x_1, \ldots, x_{p-1}$. Lanţul λ se zice ***simplu*** dacă nu trece de două ori prin acelaşi nod. Prin definiţie *lungimea* lanţului λ este dată de numărul muchiilor componente. Astfel, lanţurile de lungime unu se identifică cu muchiile grafului G; un lanţ de lungime doi este constituit din două muchii *adiacente* (au o extremitate în comun).

Un ***ciclu*** este un lanţ ale cărui extremităţi coincid.

În figura 1.2.*b*) succesiunile de noduri $\lambda = (x, y, t, u, v)$ şi $\lambda' = (x, z, y, t, z, v)$ sunt lanţuri: λ este un lanţ simplu de lungime 4 în timp ce λ' are lungimea 5 şi nu este simplu. Succesiunea $\mu = (x, y, t, u, x)$ este un ciclu de lungime 4.

Un ***drum*** în graful G este o succesiune de noduri $\delta = (x_0, x_1, \ldots, x_{p-1}, x_p)$ cu proprietatea că (x_0, x_1), (x_1, x_2), ..., (x_{p-1}, x_p) sunt *arce* permise. Nodul x_0 este *extremitatea iniţială* a drumului δ iar x_p *extremitatea finală.* Un ***circuit*** este un drum ale cărui extremităţi coincid. Este clar că orice drum este un lanţ, reciproca nefiind adevărată întotdeauna. În figura 1.3, $\delta = (z, t, x)$ este un drum de lungime 2 iar $\mu = (x, y, t, x)$ este un circuit de lungime 3; în acelaşi graf, $\lambda = (z, x, y)$ este un lanţ care nu este un drum deoarece arcul (z, x) este blocat.

Graful G = (V,E) se zice ***conex*** dacă oricare două noduri ale sale sunt extremităţile unui *lanţ*. Dacă G nu este conex, există *partiţiile* $V = V_1 \cup V_2 \cup \ldots \cup V_s$ şi $E = E_1 \cup E_2 \cup \ldots \cup E_s$ astfel încât $G_1 = (V_1,E_1)$, $G_2 = (V_2,E_2)$, ..., $G_s = (V_s,E_s)$ sunt grafuri conexe. Subgrafurile $G_1, G_2, \ldots, G_s$ se numesc ***componentele conexe*** ale grafului G. Graful din figura 1.3 este conex (ca şi cele din figurile anterioare); graful din figura 1.4 are trei componente conexe.

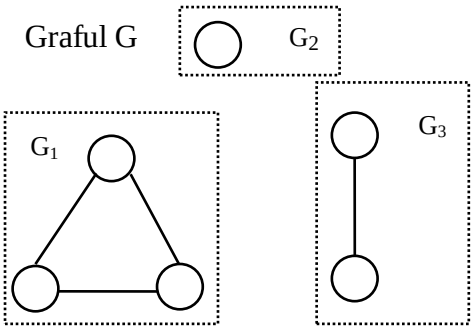


Figura 1.4.

Graful G = (V, E) se numeşte ***bipartit*** dacă mulţimea nodurilor sale poate fi descompusă în două submulţimi nevide şi disjuncte S şi T, astfel încât orice muchie din G are o extremitate în S şi cealaltă în T.

Este clar că graful asociat unei probleme clasice de transport este bipartit (figura 1.1.a), nodurile care reprezintă furnizorii formând mulţimea S iar nodurile corespunzătoare consumatorilor alcătuind mulţimea T.

Calitatea unui graf de a fi bipartit nu depinde de modul particular de reprezentare aşa cum arată exemplul din figura 1.5. O caracterizare completă a grafurilor bipartite este oferită de următoarea teoremă:

Teorema König: *Un graf este bipartit dacă şi numai dacă nu conţine cicluri de lungime impară* (altfel spus, orice ciclu al său are un număr par de muchii).

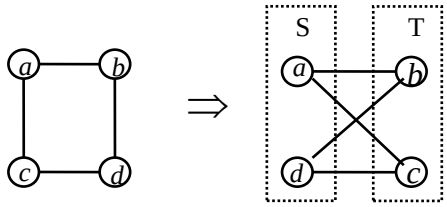


Figura 1.5.

Adiacenţă şi incidenţă. Fie $G = (V, E)$ un graf şi $x_i, x_j \in V$. Nodurile x_i şi x_j sunt *noduri adiacente* dacă există o muchie $e = [x_i, x_j]$ care le uneşte. Dacă $e = [x_i, x_j]$, muchia e şi nodul x_i sau muchia e şi nodul x_j sunt *incidente*. Două muchii e_1 şi e_2 ale grafului sunt *muchii adiacente* dacă sunt incidente în acelaşi nod (au o extremitate în comun).

Gradul unui nod $x \in V$, notat cu $d(x)$ reprezintă numărul de muchii din graful G incidente cu nodul x (care au o extremitate în x). Un nod izolat are gradul 0. Dacă $|V| = n$ şi $|E| = m$ atunci pentru orice (m, n) – graf (un graf cu n noduri şi m muchii) se verifică relaţia: $\sum_{i=1}^{n} d(x_i) = 2 \cdot m$. Această relaţie rezultă din faptul că fiecare muchie este incidentă în două noduri.

Un lanţ elementar care trece prin toate nodurile unui graf este un *lanţ hamiltonian*. Un lanţ hamiltonian ale cărui extremităţi coincid se numeşte *ciclu hamiltonian.*

Un lanţ simplu care parcurge odată şi numai o dată toate muchiile grafului este un *lanţ eulerian*. Un lanţ eulerian ale cărui extremităţi coincid se numeşte *ciclu eulerian.*

Conceptele anterioare se aplică şi în cazul grafurilor în care s-a dat o orientare a muchiilor, ele generând noţiunile de *drum/circuit hamiltonian*, respectiv *drum/circuit eulerian*.

1.3. Arbori maximali de valoare minimă

Fie $G = (V, E)$ un graf neorientat, cu mulţimea nodurilor $V = \{x_1, x_2, \ldots, x_n\}$. Notăm cu $E(x)$ mulţimea muchiilor incidente într-un nod x, cu numărul elementelor $|E(x)|$. Un nod se numeşte *suspendat* (sau nod *frunză*) dacă există o singură muchie incidentă în x (altfel spus $|E(x)| = 1$). Un nod x este *izolat* dacă nu există nici o muchie incidentă în x ($|E(x)| = 0$).

Definiţia 1.1: Un ***arbore*** este un *graf neorientat, conex şi fără cicluri* (figura 1.6.).

Propoziţia 1.1* *Un arbore* H = (V, *E*), |V| = *n*, |E| = *m are următoarele proprietăţi echivalente:*

1) *H este conex şi fără cicluri*;
2) *H este fără cicluri şi are n* – 1 *muchii;*
3) *H este conex şi are n* – 1 *muchii*;
4) *H este fără cicluri şi dacă se unesc printr-o muchie două noduri neadiacente, se creează un ciclu şi numai unul* (figura 1.7);
5) *H este conex şi dacă i se suprimă o muchie se creează două componente conexe* (*orice muchie din H este o punte între două noduri*), graful *se disconectează* (figura 1.8);
6) *Orice pereche de noduri este legată printr-un lanţ şi numai unul.*

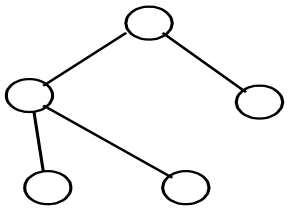


Figura 1.6.

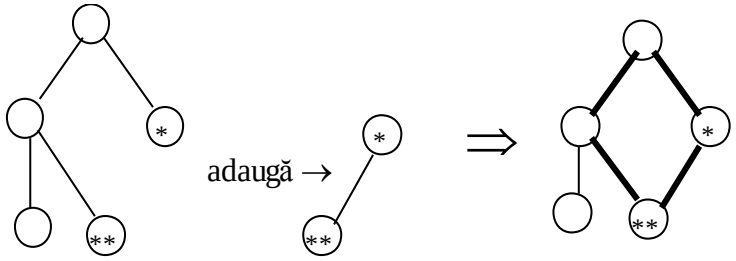


Figura 1.7

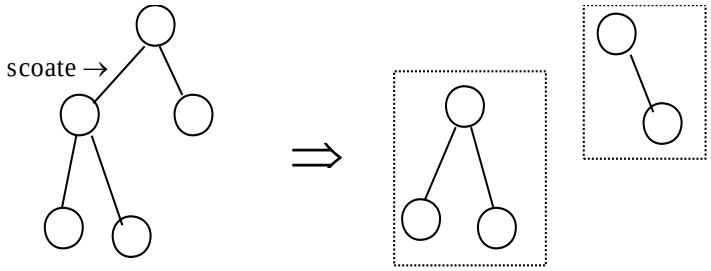
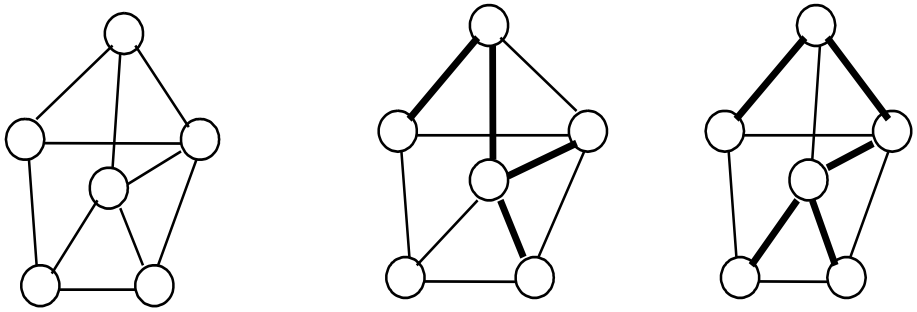


Figura 1.8

Un *arbore* H *într-un graf* G este un subgraf care - de sine stătător - este un arbore. H se zice *maximal* dacă conţine toate nodurile grafului G (figura 1.9).



G Arbore nemaximal în G Arbore maximal în G

Figura 1.9.

* Pentru demonstraţiile teoremelor, lemelor şi propoziţiilor acestui capitol a se vedea: Ciobanu Gh., Nica V., Mustaţă Floare, Mărăcine Virginia, Mitruţ D., ***"Cercetări Operaţionale. Optimizări în reţele. Teorie şi aplicaţii economice"***, Editura MatrixRom, Bucureşti, 2002

1.3.1. Arbori de acoperire

Fie $G = (V, E)$ un graf neorientat. Un arbore $H = (V, Y)$ se spune că este un *arbore care acoperă* graful G dacă $Y \subseteq E$. Dacă $|V| = n$ şi $|Y| = n - 1$ arborele de acoperire este numit *maximal*.

Teorema 1.1 *Într-un graf $G = (V, E)$ există un graf parţial H care este un arbore de acoperire dacă şi numai dacă G este conex.*

Să considerăm un graf $G = (V, E)$ conex. O muchie $e \in E$ este *valorizată* dacă i se asociază o valoare nenegativă $v(e) \geq 0$. Aceste valori pot reprezenta costuri, lungimi, profituri ale legăturilor definite prin muchiile grafului. Un graf G este *valorizat* dacă toate muchiile sale sunt valorizate. Graful G fiind un graf conex, conform teoremei 1.1, conţine un arbore de acoperire $H = (V, Y)$, $Y \subseteq E$, ca un graf parţial al său.

Valoarea unui arbore $H = (V, Y)$ conţinut în graful $G = (V, E)$, $Y \subseteq E$ este, prin definiţie, suma valorilor muchiilor sale:

$$v(H) = \sum_{e \in Y} v(e). \qquad (1.1)$$

Problema extremală pe care dorim să o rezolvăm este *problema arborelui de valoare minimă*: ***să se determine un arbore de acoperire maximal $H^* = (V, Y^*)$ conţinut într-un graf $G = (V, E)$, $Y^* \subseteq E$ de valoare minimă*:**

$$v(H^*) = \min_H v(H) \qquad (1.2)$$

unde $v(H)$ este valoarea unui arbore determinată cu relaţia (1.1).

Problema determinării unui arbore de valoare minimă constă în determinarea unui arbore maximal H^* care acoperă graful dat G astfel încât acesta să fie:

(1) *conex* (fiecare nod să fie conectat cu oricare alt nod);
(2) *fără legături redundante* (un singur lanţ este suficient pentru a conecta un nod cu oricare alt nod);
(3) *de valoare minimă* (având în vedere valorile asociate muchiilor).

Similar, se defineşte *problema arborelui de valoare maximă*, care constă în determinarea unui arbore H^* care acoperă graful dat G şi care verifică proprietăţile (1) şi (2), de mai sus, iar proprietatea (3) este înlocuită cu:

$$v(H^*) = \max_H v(H)$$

unde $v(H)$ este valoarea unui arbore determinat de relaţia (1.1).

În activitatea economică, de multe ori apare problema determinării unor arbori de valoare optimă (minimă sau maximă), întrucât există procese economice sau activităţi economice care pot fi reprezentate cu ajutorul unui arbore în condiţii convenabile. Această problemă are importanţă practică în trasarea reţelelor de comunicaţii şi a reţelelor de distribuţie, în general, în studiul reţelelor de transport:

- aprovizionarea cu apă potabilă sau cu energie electrică sau termică a unor puncte de consum de la un punct central;
- evoluţii posibile ale unui sistem pornind de la o stare iniţială;
- construirea unei reţele telefonice radiale, a unei reţele de televiziune sau internet de la un punct central;
- schemele bloc ale programelor pentru calculatoare;
- studiul circuitelor electrice în electrotehnică, etc.

Teorema 1.2 (*Berge*) *Dacă graful $G = (V, E)$ este un graf conex şi complet şi dacă valorile asociate muchiilor sunt toate distincte $v(e_i) \neq v(e_j)$ oricare ar fi $e_i, e_j \in E$ cu $i \neq j$ atunci problema determinării arborelui de acoperire de valoare minimă admite o soluţie şi numai una (arborele este unic determinat).*

Dacă *valorile muchiilor grafului* $G = (V, E)$ *nu sunt distincte*, adică, dacă de exemplu, $v(e_1) = v(e_2) = v(e_3) = \ldots$, considerăm:

$$v'(e_1) = v(e_1) + \varepsilon$$
$$v'(e_2) = v(e_2) + 2\varepsilon$$
$$v'(e_3) = v(e_3) + 3\varepsilon$$

.....

unde ε este o valoare pozitivă foarte mică, astfel încât să nu schimbe ordinea de mărime a muchiilor. Prin aceasta se introduce o ordine strictă între muchii şi putem avea situaţia din teorema 1.2 de formare a arborelui $H = (V, Y)$. În finalul construcţiei (în soluţia optimă) se ia $\varepsilon = 0$.

În *situaţia în care valorile muchiilor grafului nu sunt distincte, problema nu admite, în general, soluţie unică*. Pentru determinarea tuturor soluţiilor, se va alege la fiecare etapă k numai o muchie dintre cele cu valori egale şi se verifică pentru fiecare dintre acestea dacă formează ciclu cu muchiile deja alese în etapele 1,2, …, k-1.

Dacă graful $G = (V, E)$ nu este *complet,* se atribuie muchiilor ce lipsesc valoarea ∞ sau valori foarte mari. Aceste muchii nu vor intra în mulţimea Y a muchiilor arborelui H.

Procedeul de construcţie efectivă a unui arbore care acoperă un graf dat G este prezentat în cele ce urmează:

1.3.2. Algoritmul Kruskal[1)]

Fie $G = (V, E)$ un graf conex cu $n = |V|$ şi cu muchiile valorizate $v(e) \geq 0$, $e \in E$.

Etapa de iniţializare:

Din mulţimea muchiilor E se alege o muchie e_1 cu valoarea $v(e_1)$ cea mai mică. Dacă există mai multe muchii cu aceeaşi valoare se alege una dintre acestea. Fie Y mulţimea muchiilor alese. Iniţial $Y = \{e_1\}$ şi $|Y| = 1$.

Etapa iterativă:

Din mulţimea muchiilor neselectate $E - Y$ se alege o muchie e_i cu valoarea $v(e_i)$ cea mai mică, pentru care $Y \cup \{e_i\}$ nu conţine un ciclu. Se ia $Y \cup \{e_i\} \rightarrow Y$ şi $|Y| + 1 \rightarrow |Y|$.

STOP: Etapa iterativă se opreşte atunci când sunt alese n –1 muchii. $H = (V, Y)$ este arborele de acoperire de valoare minimă.

Arborele de valoare minimă $H^* = (V, Y)$ este determinat de muchiile alese. În cazul în care există mai multe muchii de valori egale, arborele de valoare minimă nu este, în general, unic. Este preferabil, în aceste situaţii, să fie determinaţi toţi arborii de valoare minimă şi dintre aceştia un manager (decident) să îl aleagă pe cel mai convenabil corespunzător unui alt criteriu economic.

Algoritmul Kruskal este foarte simplu. El intră în categoria algoritmilor de tip "greedy" (lacom) deoarece la fiecare pas al algoritmului se alege cea mai bună variantă. Totuşi, acest algoritm din ştiinţa managementului produce întotdeauna soluţia optimă.

Algoritmul Kruskal poate fi utilizat şi pentru determinarea *arborilor de acoperire de valoare maximă*, prin înlocuirea, în algoritm: *"se alege muchia cu valoarea cea mai mică"* cu *"se alege muchia cu valoarea cea mai mare"*.

Exemplul 1.1: ***Problema sistemului de comunicaţii***

Într-un oraş se studiază posibilitatea ca principalele instituţii să fie conectate într-un sistem printr-o reţea de telecomunicaţii. Schema tuturor legăturilor posibile, precum şi costul executării acestor legături între şase dintre instituţiile oraşului sunt indicate în graful din figura 1.10. Valorile indicate pe arce reprezintă costurile (în unităţi monetare) asociate realizării legăturilor dintre instituţii. Să se determine modul în care vor fi interconectate cele şase instituţii astfel încât costul realizării sistemului de telecomunicaţii să fie minim.

[1)] J. B. Kruskal, *On the Shortest Spanning Subtree of a Graph and the Traveling Salesman Problem*, Proceedings of the American Mathematical Society 7, 48-50, 1956.

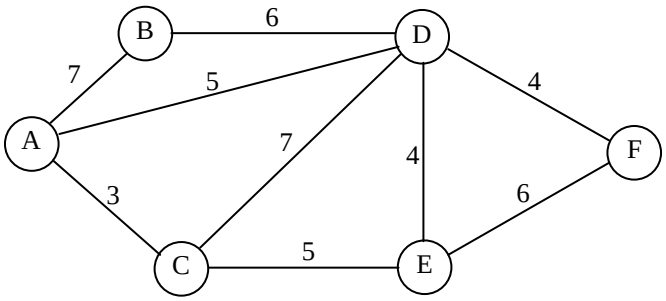
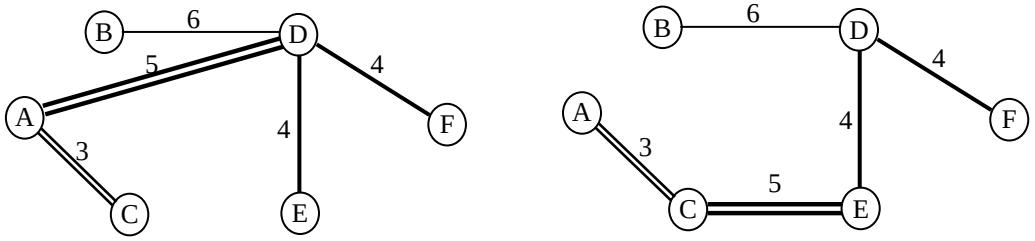


Figura 1.10

Rezolvare:

Pentru rezolvarea acestei probleme se aplică algoritmul Kruskal. Muchia cu cel mai mic cost este [*A,C*] cu *v*[*A, C*] = 3 u.m.. Aceasta este prima alegere. În continuare, pot fi alese muchiile [*D, E*] şi [*D, F*], fiecare având costul de 4 u.m.. Pentru că fiecare din aceste muchii nu formează ciclu cu muchia [*A, C*] şi nici toate cele trei muchii nu formează ciclu, se alege a doua oară şi a treia oară câte una din muchiile [*D, E*] şi [*D,F*]. Muchiile [*A, D*] şi [*C, E*] cu costul *v*[*A, D*] = *v*[*C, E*] = 5 u.m. pot fi alese, în continuare. Se alege una din muchii, cealaltă formează un ciclu. Deci, pot exista două variante conform figurilor 1.11.a) şi 1.11.b) prin alegerea fie a muchiei [*A*, D] fie a muchiei [*C, E*].



a) H_1 b) H_2

Figura 1.11

Următoarea alegere este fie muchia [*E, F*] fie muchia [*B, D*] ambele având costul *v*[*E, F*] = *v*[*B, D*] = 6 u.m.. Dar, muchia [*E, F*] formează ciclu în ambii arbori parţiali împreună cu muchiile [*D, E*] şi [*D, F*]. În acest caz, se alege muchia [*B, D*] în ambii arbori parţiali.

Rezultă două soluţii optime, deci doi arbori de valoare minimă prezentaţi în figura 1.11, cu $v(H_1) = v(H_2) = 22$. Alegerea între aceşti arbori va avea în vedere şi alte considerente economice pe lângă costul instalării sistemului de telecomunicaţii.

Exemplul 1.2: ***Problema modernizării reţelei de drumuri***

Prefectura judeţului *X* şi-a fixat ca obiectiv modernizarea reţelei drumurilor care leagă localităţile judeţului conform grafului din figura 1.12. Pe fiecare muchie este înscrisă valoarea numerică în unităţi monetare [u.m.] a costului modernizării tronsonului respectiv. În prima etapă se caută să se modernizeze numai unele drumuri astfel încât fiecare localitate să fie conectată la cel puţin un drum modernizat şi costul întregii operaţii de modernizare (parţială) să fie minim.

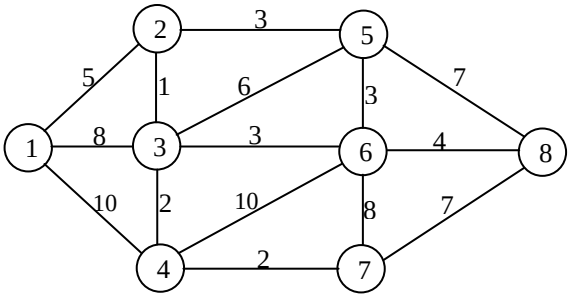


Figura 1.12

Rezolvarea problemei se reduce la determinarea unui arbore maximal de valoare minimă. **TEMĂ:** Aplicaţi algoritmul Kruskal pentru determinarea variantei optime de modernizare a drumurilor.

1.3.3. Arborescenţe

Fie $G = (X, U)$ un graf ***orientat***.

Un nod $x_k \in X$ se numeşte *rădăcină* în graful G dacă pentru oricare $x_i \in X$, $x_i \neq x_k$ există în G un drum de la x_k la x_i.

Arborescenţa este un graf orientat, finit şi fără circuite, în care:

- există un nod şi numai unul numit rădăcină care nu este extremitatea terminală a nici unui arc;
- oricare nod diferit de nodul rădăcină este *extremitatea terminală* a unui singur arc.

Propoziţia 1.2 *Orice arborescenţă este un arbore.*

Dacă n este numărul nodurilor, atunci numărul arcelor unei arborescenţe este $m = n - 1$. Prin urmare, arborescenţa este un arbore cu orientare unică pe muchii, în care se pun în evidenţă drumuri de la nodul rădăcină la toate celelalte noduri. Dacă $G = (X, U)$ este un graf orientat şi conex, printre grafurile parţiale există arborescenţe ale grafului dat.

Arborele genealogic este un exemplu de arborescenţă, nodurile fără descendenţi fiind noduri frunză.

CURSUL 2

1.4. Drumuri de valoare minimă

1.4.1. Problema drumului de valoare minimă

Fie $G = (V, E)$ un graf finit (orientat, neorientat sau parţial orientat).

Dacă $e \in E$ este o muchie cu extremităţile x_i şi x_j atunci această muchie are o orientare permisă prin arcele (x_i, x_j) şi (x_j, x_i). În acest mod, un graf neorientat este tratat ca graf orientat $G = (V, U)$. În continuare, ne referim numai la grafuri *orientate finite*.

Dacă arcul u este dat prin extremităţile sale $u = (x_i, x_j)$, cu $x_i, x_j \in V$, valoarea arcului va fi dată prin notaţiile $v(x_i, x_j)$ sau v_{ij}. Dacă u este o muchie în graful $G = (V, E)$ atunci $v(u) = v(x_i, x_j) = v(x_j, x_i)$, deci *pe ambele sensuri arcele au aceeaşi valoare*. Pe o muchie neorientată $\{i, j\}$ (sau $[i, j]$) putem avea însă şi $v(i, j) \neq v(j, i)$. *Valorile asociate arcelor pot reprezenta în practică distanţele între două localităţi, costurile executării unei lucrări pe tronsoanele respective, costurile trecerii unei linii de fabricaţie de la un produs la altul, timpii de execuţie ai unor activităţi, măsura siguranţei transmisiunii unui semnal*, etc.

Pentru un drum λ între nodul $x_i \in X$ şi nodul $x_j \in X$, precizat prin specificarea arcelor componente, se defineşte *valoarea drumului* λ ca fiind suma valorilor arcelor componente:

$$v(\lambda) = \sum_{u \in \lambda} v(u). \qquad (1.3)$$

Din noţiunile introductive ştim că un drum este o succesiune de arce permise:

$$\lambda = (u_1, u_2, ..., u_q)\ ,\ u_i \in U$$

cu proprietatea că pentru orice $1 \leq i \leq q - 1$, extremitatea finală a unui arc u_i coincide cu extremitatea iniţială a arcului u_{i+1}. Altfel spus, de-a lungul drumului λ arcele componente sunt orientate în acelaşi sens, de la nodul iniţial al arcului u_1 către nodul final al arcului u_q. Numărul arcelor componente defineşte lungimea $l(\lambda)$ a drumului λ. Drumurile de lungime 1 se identifică cu arcele permise ale grafului.

Dacă $u_1 = (x_0, x_1)$, $u_2 = (x_1, x_2)$, ..., $u_q = (x_{q-1}, x_q)$ sunt arcele date prin extremităţile lor, atunci un drum poate fi definit şi prin succesiunea de noduri:

$$\lambda = (x_0, x_1, x_2, ..., x_q)$$

Nodul x_0 este extremitatea de start a drumului λ iar nodul x_q extremitatea finală. Dacă extremităţile sale coincid ($x_0 = x_q$) atunci drumul λ este un *circuit*.

Un drum este *elementar* dacă nodurile sale sunt distincte (drumul trece o singură dată prin fiecare nod al său). Un drum este *simplu* dacă arcele sale sunt distincte (drumul trece o singură dată prin fiecare arc al său). Evident, orice drum elementar este şi simplu. Reciproca nu este, în general, adevărată. Mulţimea drumurilor elementare dintr-un graf finit este finită. Un drum λ dintr-un graf G este inclus în drumul λ' ($\lambda \subseteq \lambda'$) dacă toate arcele drumului λ sunt arce şi ale drumului λ'. Evident, $v(\lambda) \le v(\lambda')$.

O problemă extremală într-un graf orientat $G = (V, U)$ constă în ***determinarea unui drum elementar*** *(sau a drumurilor elementare),* ***între două noduri date, de valoare minimă*** *(sau de valoare maximă).*

Fie s şi t două noduri oarecare ale grafului G. Notăm cu $G^*(s, t)$ un subgraf al grafului G format din mulţimea drumurilor din graful considerat $G = (V, U)$ între nodurile s şi t, în care sunt incluse şi drumurile de lungime 1. Problema drumurilor de valoare minimă între nodurile s şi t se defineşte ca problema determinării drumului λ^* de la nodul s la nodul t în graful $G^*(s, t)$, de valoare minimă:

$$v(\lambda^*) = \min\{v(\lambda) \mid \lambda \in G^*(s, t)\}. \qquad (1.4)$$

Dacă se consideră drumurile de valoare minimă de la un nod s la oricare alt nod al grafului G, mulţimea acestor drumuri formează un graf parţial orientat în G, numit *graful drumurilor de valoare minimă* cu originea în nodul s şi este notat $G^*(s)$ sau simplu G^*. Problema pusă în acest fel cuprinde şi problema determinării drumurilor de valoare minimă de la nodul s la un alt nod oarecare t. Într-adevăr, dacă în $G^*(s)$ figurează şi nodul t, atunci în acest graf vom găsi şi drumurile de valoare minimă de la s la t.

Dacă luăm în considerare numai drumurile elementare de la s la t, deoarece mulţimea acestora este finită, atunci există, întotdeauna, un drum de la s la t de valoare minimă. Aceste drumuri elementare sunt identificate greoi, motiv pentru care drumurile de valoare minimă se determină considerând, în graf, toate drumurile elementare sau neelementare între cele două noduri.

Este posibil ca problema (1.4) să nu admită soluţii optime (de exemplu în cazul existenţei circuitelor de valoare negativă). În acest caz, există o infinitate de drumuri de la s la t a căror mulţime de valori poate fi nemărginită inferior.

Propoziţia 1.3 *Problema drumurilor de valoare minimă cu originea în nodul s admite soluţii în graful $G = (V, U)$ dacă şi numai dacă graful $G^*(s)$ nu conţine nici un circuit cu valoare strict negativă.*

Demonstraţie: Fie s un nod al grafului G. Vom presupune că în graful $G^*(s)$ există un circuit $c = (x = x_0, x_i, x_j, \ldots, x_k, x)$ de valoare strict negativă $v(c) = \rho < 0$. Deoarece $x \in X$ există un drum μ de la s la x şi fie ω valoarea acestuia. Prin prelungirea drumului μ cu circuitul c, noul drum $\mu \circ c$ are valoarea $\omega + \rho$. De asemenea, $\mu \circ c \circ c$ are valoarea $\omega + 2\rho$. Prin recurenţă, prelungirea drumului μ cu de k ori circuitul c, duce la drumul $\mu \circ \underbrace{c \circ c \circ \ldots \circ c}_{k \text{ ori}}$, $k \in \mathbb{N}$, are valoarea $\omega + k\rho$. Prin urmare, $\lim_{k\to\infty}(\omega + k\rho) = -\infty$.

În concluzie, nu există un drum de valoare minimă în mulţimea drumurilor de la s la x şi problema nu are soluţie.

Reciproc, dacă $G^*(s)$ nu are circuite de valoare strict negativă atunci pentru orice $x \in X$ există un drum μ de la s la x şi deci şi un drum elementar μ^E de la s la x, inclus în μ. Întrucât graful nu admite circuite de valoare strict negativă, prin eliminarea circuitelor suplimentare ale drumului μ faţă de μ^E avem $v(\mu^E) \le v(\mu)$, prin urmare minorantul va fi atins într-un drum elementar.

Pentru că mulţimea drumurilor elementare este finită minimul acesteia este efectiv atins într-un element al acesteia şi problema are soluţii, în sensul că există drumuri de valoare minimă.

Un circuit în graful G de valoare strict negativă este numit *circuit absorbant*. Deoarece problemele de determinare a drumurilor de valoare minimă între nodurile date se rezolvă considerând toate drumurile posibile, nu numai cele elementare, *vom presupune verificată următoarea condiţie:* ***în graful G nu există circuite de valoare strict negativă.***

Determinarea drumului de valoare minimă poate avea loc într-un graf în care valorile sau ponderile arcelor sunt nenegative. În grafuri în care există arce cu ponderi negative, sunt necesare eforturi mai mari pentru rezolvare; în acest caz se va presupune îndeplinită condiţia: *nu există circuite de valoare strict negativă* (*absorbante*).

Observaţie. Similar, problema drumurilor de valoare maximă într-un graf G cu nodul de pornire s admite soluţie dacă şi numai dacă graful $G^*(s)$ *nu conţine nici un circuit de valoare strict pozitivă*.

Dacă λ şi λ' sunt drumuri elementare în *G* având aceleaşi extremităţi, atunci drumurile pot fi comparabile: fie $v(\lambda') \le v(\lambda)$, fie $v(\lambda) \le v(\lambda')$.

În cazul în care există drumuri de la *s* la *t*, evaluarea şi compararea acestora, conform afirmaţiei anterioare, se poate limita la drumurile elementare cu aceleaşi extremităţi. Numărul drumurilor elementare fiind finit unul dintre ele va fi drumul de la *s* la *t* de valoare minimă.

Dacă graful *G* nu este conex şi nodurile *s* şi *t* nu fac parte din aceeaşi componentă conexă (figura 1.13.a) sau orientările pe arce nu permit atingerea nodului *t* din nodul *s* (figura 1.13.b), există posibilitatea ca în graful *G* să nu existe drumuri de la *s* la *t*.

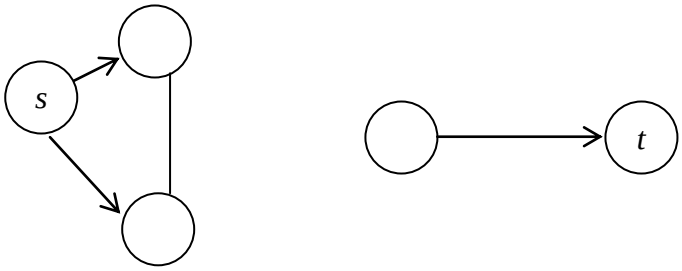


Figura 1.13.a

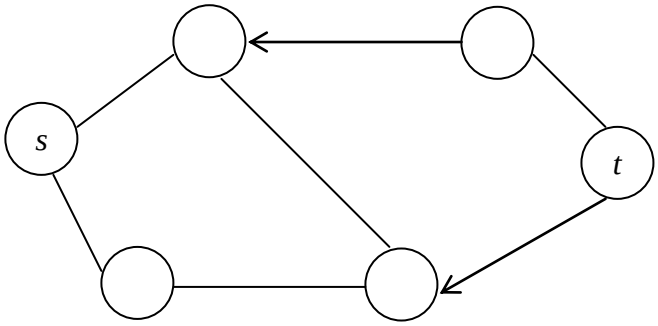


Figura 1.13.b

În acest context vom spune că nodul *i* este atins din nodul *s* dacă există un drum de la *s* la *i* format din *arce permise*. Ne ocupăm de rezolvarea problemei determinării drumului de valoare minimă de la s la toate celelalte noduri ale grafului, $P(s)$, şi a cazului său particular $P(s,t)$ - determinarea drumului de valoare minimă de la *s* la *t*, în ipoteza: *toate valorile arcelor permise sunt* ***nenegative***: $v(u) \ge 0;\ \forall u \in U$.

1.4.2. Metoda Dijkstra[†)]

1. Preliminarii

1) Fiecărui nod $i \in V$ i s-a asociat o variabilă $d(i)$ numită în continuare ***eticheta nodului i***. Prin definiţie $d(s) = 0$. În oricare moment al aplicării algoritmului variabila $d(i)$ reţine valoarea unui drum de la *s* la *i* găsit de algoritm până în acel moment. Dacă algoritmul nu a găsit încă un drum de la *s* la *i* variabila $d(i)$ are valoarea +∞. La finalul aplicării metodei, eticheta indică valoarea celui mai „scurt" drum de la nodul *s* la nodul *i*. Pe parcursul aplicării algoritmului etichetele se corectează în sensul micşorării valorilor lor. În momentul în care o etichetă $d(i)$ atinge valoarea minimă ea devine **permanentă**.

Metodele de rezolvare a problemei determinării drumului de valoare minimă între două noduri ale unui graf sunt procedee de etichetare care se împart în două clase:

- ***metode cu etichetare permanentă*** (la fiecare iteraţie se identifică un nod a cărui etichetă se permanentizează – valoarea ei nu se va mai modifica până la finalul rezolvării); şi

[†)] Dijkstra, E.W. "*A Note on two Problems in Connection with Graphs*", Numerische Math. 1, 269-271, 1959

- ***metode cu etichetare temporară*** (etichetele tuturor nodurilor au valori temporare care devin permanente doar la finalul rezolvării).

Metodele aparţinând primei clase se aplică numai grafurilor cu valori (costuri) nenegative ale muchiilor, în timp ce metodele aparţinând celei de a doua clase se aplică grafurilor cu valori (costuri) negative ale muchiilor. Algoritmul Dijkstra aparţine primei clase.

2) Fiecărui nod $i \in V$ diferit de *s* i se asociază o altă variabilă PRED(i) numită ***indicator de precedenţă*** cu următoarea semnificaţie: în oricare moment al aplicării algoritmului, PRED(i) conţine ultimul nod dinaintea lui *i* pe drumul de la *s* la *i* găsit de algoritm până în acel moment. Atâta timp cât un asemenea drum nu a fost găsit $d(i) = +\infty$, indicatorul PRED(i) nu este definit el fiind o locaţie vidă (PRED(*i*) = ∅).

3) Iniţializăm o listă ***P*** în care vom include toate nodurile $i \in \mathrm{V}$ pentru care algoritmul a găsit un drum de valoare minimă de la *s* la *i* (lista nodurilor cu ***eticheta declarată permanentă***). La start *P* se reduce la nodul de plecare *s*.

4) Iniţializăm o listă ***T*** în care vom include toate nodurile $j \in V$ care sunt vecine cu noduri din lista *P*: un nod *j* este vecin cu *i* dacă arcul (i, j) este **permis.** *T* este lista nodurilor cu ***eticheta declarată temporară***.

Nodurile cu etichetă permanentă din lista *P* se selectează din lista *T* a nodurilor candidate. Corectarea unei etichete se face după următoarea schemă echivalentă (figura 1.14).

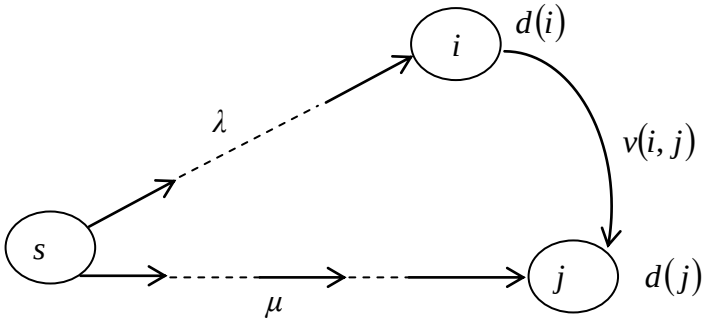


Figura 1.14

Referitor la un arc permis $(i, j) \in U$ cu valoarea $v(i, j) \geq 0$ presupunem că algoritmul a găsit un drum λ de la s la *i* a cărui valoare e reţinută în eticheta $d(i)$ şi un drum μ de la *s* la *j* a cărui valoare se găseşte în locaţia $d(j)$. Figura 1.14 pune în evidenţă cele două drumuri de la *s* la *j*.

- drumul $\lambda \cup (i, j)$ de valoare $d(i) + v(i, j)$;
- „vechiul" drum μ de valoare $d(j)$.

Dacă $d(i) + v(i, j) < d(j)$ atunci primul drum are o valoare mai mică şi ca urmare va fi reţinut de algoritm ca fiind cel mai bun drum de la *s* la *j* găsit până în acest moment. Memorarea acestui nou drum se face prin corectarea etichetei $d(j)$ care ia valoarea $d(j) = d(i) + v(i, j)$ şi actualizarea indicatorului de precedenţă PRED (*j*): PRED (*j*) = *i*.

Cu aceste pregătiri trecem la prezentarea algoritmului Dijkstra.

2. Algoritmul Dijkstra

START: Iniţializăm:

- Lista nodurilor cu eticheta permanentă: $P = \{s\}$;
- Lista nodurilor cu eticheta temporară: $T = \{i \in V$ / există arcul permis $(s, i) \in U\}$;
- Etichetele: $d(s) = 0$; $d(i) = v(s, i)$ *pentru* $i \in T$ *si* $d(i) = +\infty$ în rest;
- Predecesorii: PRED$(i) = s$ *pentru* $i \in T$ şi PRED(j) nedefinit în rest.

ITERAŢIE:

Pas 1: Dacă lista *T* e vidă, STOP: algoritmul a găsit toate nodurile ce pot fi atinse din *s* de-a lungul unor drumuri de valoare minimă, toate nodurile sunt în lista *P* şi valorile minime ale drumurilor de la *s* la aceste noduri sunt indicate de etichetele corespunzătoare. Identificarea drumului de valoare minimă se face cu ajutorul indicatorului de precedenţă, din aproape în aproape, de la *t* (sau de la fiecare nod al grafului) către *s*.

Dacă $T \neq \varnothing$ se trece la pasul 2.

Pas 2: Se selectează nodul $i^* \in T$ cu proprietatea: $d(i^*) = \min\{d(i), i \in T\}$

Se transferă nodul i^* din lista *T* în lista *P* (i^* devine nod cu eticheta permanentă). Drumul de la *s* la i^* găsit până în momentul selectării este un drum de valoare minimă. Se adaugă la lista *T* toate nodurile $j \in V - \{s\}$ care nu figurau în această listă, noduri adiacente lui i^* pentru care există arcul $(i^*, j) \in U$.

Pas 3: Pentru fiecare nod $j \in T$, vecin cu i^* se compară $d(j)$ cu suma $d(i^*) + v(i^*, j)$ (vezi figura 1.14 cu *i* schimbat în i^*).

- Dacă $d(i^*) + v(i^*, j) \geq d(j)$ se trece la examinarea altui nod din *T* vecin cu i^*;
- Dacă $d(i^*) + v(i^*, j) < d(j)$ se fac actualizările:

$$d(j) = d(i^*) + v(i^*, j) \text{ şi}$$
$$\text{PRED}(j) = i^*$$

după care se trece la examinarea altui nod din lista *T* vecin cu i^*.

După examinarea tuturor vecinilor lui i^* din *T* se revine la Pasul 1 în cadrul unei noi iteraţii.

STOP: Algoritmul se termină în următoarele situaţii:

- La Pasul 1: dacă lista *T* e vidă, $T = \varnothing$;
- La Pasul 2: Dacă ne interesează numai drumul de valoare minimă de la *s* la un nod *t* algoritmul descris se termină în momentul în care nodul selectat este *t* : $i^* = t$.

NOTĂ:

1) Dacă algoritmul se termină la Pasul 1 şi $d(t) = +\infty$, în graful *G* **NU există** nici un drum de la *s* la *t* (vezi situaţiile din figura 1.13).
2) Aplicarea Algoritmului Dijkstra nu rezolvă doar problema găsirii drumului de valoare minimă de la *s* la *t* ci a TUTUROR drumurilor de valoare minimă de la *s* la toate celelalte noduri ale grafului *G*.

Exemplul 1.3. Aplicaţi metoda DISKTRA pentru a determina drumurile de valoare minimă de la nodul *s* la nodurile ce pot fi atinse din *s* în graful din figura 1.15.

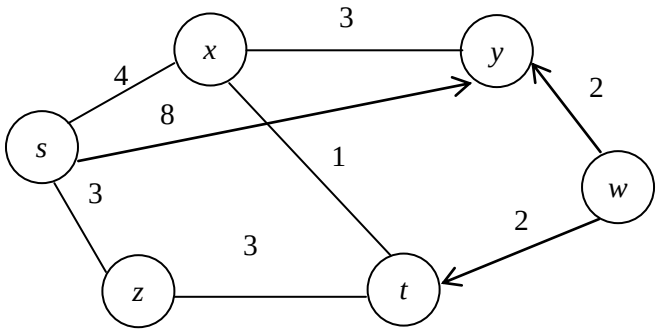


Figura 1.15

Fiecărei muchii îi este asociat un cost valabil în ambele sensuri de parcurgere pe muchiile neorientate.

Rezolvare: Valorile iniţiale şi intermediare ale variabilelor $d(i)$ şi PRED(*i*) sunt date în tabelul 1.

START: Iniţializăm:

- Listele $P = \{s\}$, $T = \{x, y, z\}$

- $d(s)=0;\ d(x)=4, d(y)=8, d(z)=3;\ d(i)=+\infty$ în rest;
- $\text{PRED}(i)=s$ *pentru* $i\in T$ şi $\text{PRED}(j)$ nedefinit în rest.

Iteraţia 1: Se calculează $\min\{d(i), i\in T\}=\min\{4, 8, 3\}=3\Rightarrow i^*=z$

Transferăm *z* din lista *T* în lista *P*: $P=\{s,z\}$ şi includem în lista *T* nodul *t* care este vecin cu $i^*=z: T=\{x,y,t\}$.

Examinăm toţi vecinii lui $i^*=z$ din lista *T*: există un singur vecin, nodul *t*. Deoarece $d(z)+v(z,t)=3+3=6<+\infty=d(t)$ actualizăm $d(t)=6$ şi $\text{PRED}(t)=z$ (valorile actualizate se trec în tabelul 1 în linia ITERAŢIA 1, celelalte valori nu se modifică).

Iteraţia 2: Se calculează $\min\{d(i), i\in T\}=\min\{4, 8, 6\}=4\Rightarrow i^*=x$

Actualizăm listele *P* şi *T*: $P=\{s,z,x\},\ T=\{y,t\}$

Observăm că vecinii lui $i^*=x$ cu etichetă nepermanentă, adică *y* şi *t* sunt deja în lista *T*.

Trecem la corectarea – dacă este cazul – a etichetelor $d(y)$ şi $d(t)$:

- $d(x)+v(x,y)=4+3=7<8=d(y)\Rightarrow$ actualizăm: $d(y)=7$ şi $\text{PRED}(y)=x$
- $d(x)+v(x,t)=4+1=5<6=d(t)\Rightarrow$ actualizăm $d(t)=5$ şi $\text{PRED}(t)=x$.

Celelalte etichete şi indicatori de precedenţă rămân neschimbaţi.

Iteraţia 3: Calculăm $\min\{d(i), i\in T\}=\{7, 5\}=5\Rightarrow i^*=t$.

Actualizăm $P=\{s,z,x,t\}; T=\{y\}$.

$i^*=t$ nu are vecini cu etichetă nepermanentă, nici în *T*, nici în afara lui *T*.

În această iteraţie nu are loc nici o corectare de indicatori.

Iteraţia 4: $i^*=y$

$P=\{s,z,x,t,y\}; T=\Phi$

$i^*=y$ nu are nici un vecin cu etichetă nepermanentă.

STOP, pentru că $T=\Phi$.

Tabelul 1

ITERAŢIA	*d*(*s*)	*d*(*x*)	*PRED*(*x*)	*d*(*y*)	*PRED* (*y*)	*d*(*z*)	*PRED* (*z*)	*d*(*t*)	*PRED* (*t*)	*d*(*w*)	*PRED* (*w*)
START	0*	4	*s*	8	*s*	3	*s*	+∞	-	+∞	-
ITERAŢIA 1	-	4	*s*	8	*s*	3*	*s*	6	*z*	+∞	-
ITERAŢIA 2	-	4*	*s*	7	*x*	-	-	5	*x*	+∞	-
ITERAŢIA 3	-	-	-	7	*x*	-	-	5*	*x*	+∞	-
ITERAŢIA 4	-	-	-	7*	*x*	-	-	-	-	+∞	-
STOP	**0**	4*	*s*	**7**	***x***	**3**	***s***	**5**	***x***	+∞	**-**

Drumurile de valoare minimă se construiesc cu ajutorul indicelui de precedenţă şi sunt vizualizate în figura 1.16.

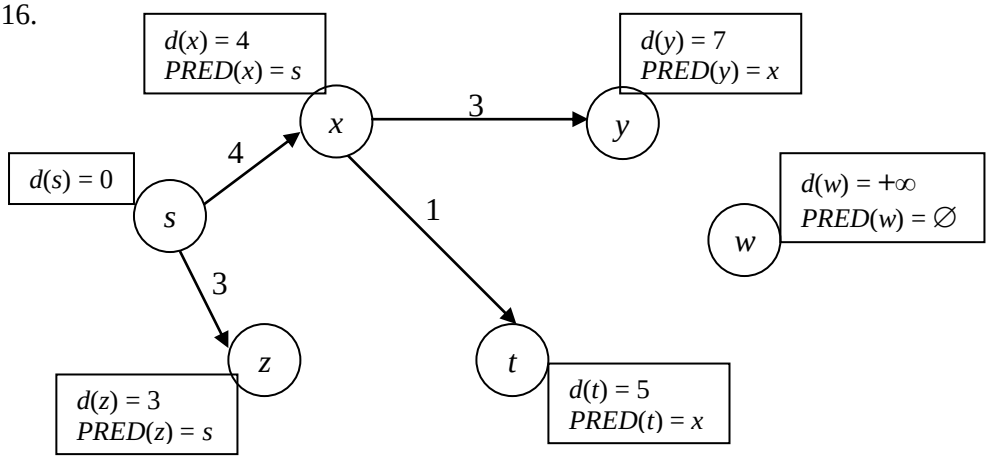


Figura 1.16

Aşa cum se observă, deşi graful din figura 1.15 este conex, datorită orientărilor existente pe muchii, nodul *w* nu poate fi atins din *s* (nu se găseşte în lista *P*). Lungimile drumurilor de valoare minimă de la *s* la toate celelalte noduri ale grafului sunt date de etichetele fiecărui nod (de exemplu, valoarea celui mai scurt drum de la *s* la *y* este 7 şi el trece prin nodul *x* – PRED(*y*) = *x* iar PRED(*x*) = *s*).

Exemplul 1.4. TEMĂ: 1. Aplicaţi algoritmul Kruskal pentru determinarea arborilor maximali de valoare minimă în grafurile din figura 1.17, 1.18 şi 1.19.

2. Aplicaţi metoda DISKTRA pentru a determina drumurile de valoare minimă de la nodul *s* la nodurile ce pot fi atinse de *s* din grafurile din figura 1.17, 1.18 şi 1.19.

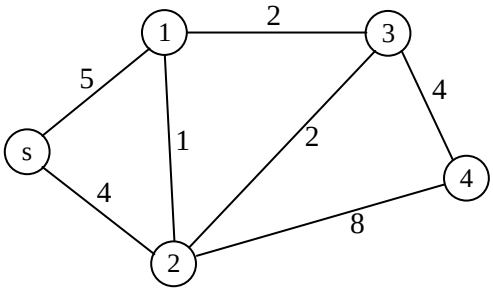


Figura 1.17

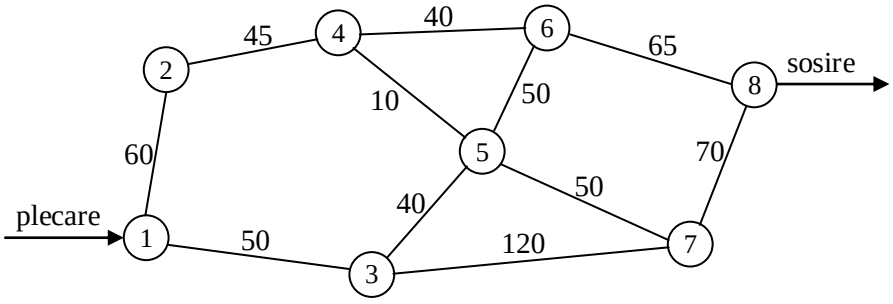


Figura 1.18

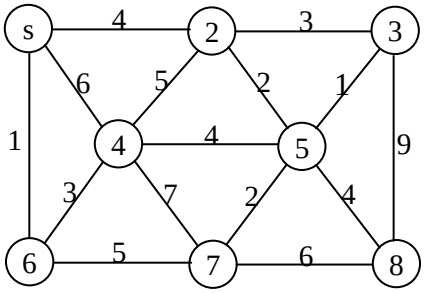


Figura 1.19

Observaţie finală: Una dintre metodele de determinare a drumurilor de valoare minimă în grafuri cu costuri negative asociate muchiilor este Metoda Ford. Ea aparţine clasei metodelor cu etichetare temporară şi poate fi lecturată în: Ciobanu Gh., Nica V., Floare Mustaţă, Virginia Mărăcine, Mitruţ D., ***"Cercetări Operaţionale. Optimizări în reţele. Teorie şi aplicaţii economice"***, Editura MatrixRom, Bucureşti, 2002.

CURSUL 3

CAPITOLUL 2
MANAGEMENTUL PROIECTELOR COMPLEXE
(Analiza Drumului Critic)

NOTĂ: Suportul de curs al Capitolelor 1, 2 şi 3 are la bază lucrările:
- Ciobanu, Gh., Nica, V., Mustaţă, Fl., Mărăcine, V. Mitrut, D. (2002), ***Cercetări operaţionale. Optimizări în reţele. Teorie şi aplicatii economice***, Editura MATRIX ROM, Bucureşti;
- Ciobanu, Gh., Nica, V., Mustaţă Fl., Mărăcine, V. (1996), ***Cercetări operaţionale cu aplicaţii în economie. Teoria grafurilor şi Analiza drumului critic***, Editura MATRIX-ROM, Bucureşti

2.1. Proiect. Activităţi. Precedenţe. Conducerea unui proiect.

În multe situaţii managerii trebuie să-şi asume responsabilitatea planificării, programării şi urmăririi unei acţiuni complexe constând din numeroase activităţi şi la care participă diverse unităţi productive, instituţii sau persoane fizice. Aceste activităţi sunt atât de mari încât un singur om nu poate stăpâni toate informaţiile aferente pentru a decide în cunoştinţă de cauză. Ca exemple de PROIECTE am putea lua:

- construcţia unui obiectiv civil (un hotel, o autostradă), o fabrică, un avion, o navă, o rachetă;
- cercetarea şi dezvoltarea de noi produse şi servicii;
- proiecte de îmbunătăţiri funciare;
- revizia şi reparaţia unor noi instalaţii.

În proiectele de tipul celor amintite, managerii trebuie să coordoneze proiectul a.î. acţiunea să se termine la termen. Un factor care complică enorm această sarcină este interdependenţa dintre activităţi (unele încep numai după terminarea altora). Un proiect poate avea sute sau mii de activităţi, rezultând că managerii caută metode care să-i ajute să răspundă la întrebări precum:

- care este durata de execuţie (minimă) a unei activităţi?
- când trebuie să înceapă şi să se termine o anumită activitate?
- care sunt activităţile CRITICE (activităţi care trebuie să respecte exact termenul de programare pentru a menţine întreg proiectul sub control)?
- cât de mult pot fi întârziate activităţile NECRITICE înainte de a cauza o întârziere a întregului proiect?

Terminologie specifică:

PROIECT	= **acţiune complexă compusă** dintr-un număr mare de activităţi **intercorelate**. Acţiunea are drept scop realizarea unui obiectiv şi necesită resurse materiale, timp, bani.
ACTIVITATE	În general, o activitate = o **porţiune bine determinată** a unui proiect care **necesită timp şi eventual resurse**. De ex.: Într-un proiect de construire a unui imobil, activităţi specifice sunt: - excavarea fundaţiilor; - turnarea betonului în fundaţii; - întărirea betonului. Primele 2 activităţi necesită personal calificat, ultima are nevoie doar de timp. Gradul de detaliere al unui proiect în activităţi este un element important. În faza de concepţie proiectul este divizat în număr mic de activităţi de talie mare. Pe aceste structuri foarte simple managerii îşi fac o idee despre posibilitatea de realizare a proiectului şi pot analiza costurile. Abia după aprobarea proiectului, fiecare activitate „mare" este privită de sine stătător şi descompusă în părţi mai mici. De exemplu: Turnarea betonului în fundaţie presupune: $A \equiv$ aducerea în şantier a materialelor lemnoase şi feroase pentru cofraje şi armături. $B \equiv$ confecţionarea cofrajelor $C \equiv$ confecţionarea armăturilor $D \equiv$ montarea cofrajelor $E \equiv$ montarea armăturilor $F \equiv$ prepararea betonului $G \equiv$ turnarea betonului în cofraje

	Un grad mic de detaliere nu este de folos pentru conducerea generală a proiectului, pe când un grad mare de detaliere transformă proiectul într-un colos. Detalierea unui proiect în activităţi se face PROGRESIV în funcţie de etapa de studiu a proiectului sau de necesităţi.
DEPENDENŢE	Faptul că între activităţile unui proiect există DEPENDENŢE nu înseamnă decât că: unele activităţi se pot realiza simultan, sunt independente, altele nu se pot desfăşura decât una după cealaltă. Dacă *A* şi *B* sunt activităţi ale unui proiect *P* vom spune că: - *A* precede *B* sau că *B* este precedat de *A* dacă *B* nu poate începe decât după terminarea activităţii *A*. Notăm $A \prec B$. - dacă *A* precede *B* şi *B* poate începe **imediat** după terminarea lui *A* vom spune că *A* precede **direct** *B* sau că *B* e precedat direct de *A*. Notăm $A \overset{d}{\prec} B$. **Observaţii:** • relaţia de precedenţă este o relaţie de ORDINE în mulţimea activităţilor unui proiect: Dacă *A* precede *B* şi *B* precede *C* atunci *A* precede *C*. • relaţia de precedenţă este, în general o relaţie de ordine PARIŢIALĂ în sensul că pot exista activităţi INDEPENDENTE. • relaţia de precedenţă directă **NU este TRANZITIVĂ**: Dacă *A* precede direct *B* şi *B* precede direct *C* atunci *A* precede *C* dar nu direct între ele aflându-se activitatea *B* • precedenţele directe determină toate relaţiile de precedenţă dintre activităţile unui proiect în sensul că: $A \prec B \Leftrightarrow$ ..... activităţile $C_1, C_2, \ldots, C_p$ astfel încât $A \overset{d}{\prec} C_1 \overset{d}{\prec} C_2 \overset{d}{\prec} \ldots \overset{d}{\prec} C_p \overset{d}{\prec} B$. **Exemplu:** Să considerăm acţiunea complexă (proiectul) *P* de turnare a betonului în fundaţie din exemplul precedent compusă din activităţile *A, B, . . .F, G*. Precedenţele directe sunt: $A \overset{d}{\prec} B, C;\ B \overset{d}{\prec} D;\ C \overset{d}{\prec} E;\ D, E, F \overset{d}{\prec} G$
DURATELE ACTIVITĂ-ŢILOR	După identificarea activităţilor şi a relaţiilor de precedenţă directă se estimează DURATELE activităţilor = numărul de unităţi de timp care trece de la începerea activităţii şi până la finalizarea ei.

La finalul definirii elementelor anterioare se întocmeşte un document numit LISTĂ DE ACTIVITĂŢI care conţine cel puţin următoarele elemente:

Nr. crt.	Denumirea activităţii	Codul activităţii	Activităţi DIRECT precedente	Durata

Observaţie finală. Reluând exemplul precedent vom recunoaşte că simpla listare a precedenţelor (directe) dintre activităţi, altfel spus cunoaşterea listei de activităţi a proiectului nu ne prea ajută la înţelegerea structurii acestuia!

În schimb, următorul ,,desen" este mai sugestiv.

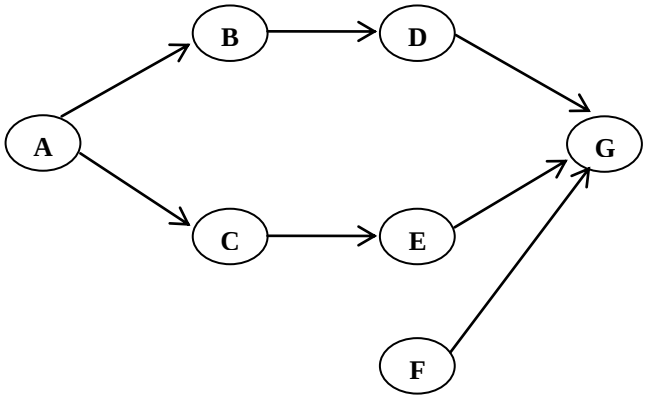


Figura 2.1

Ajungem astfel la problema REPREZENTĂRII STRUCTURII unui proiect, **structură** însemnând aici cuplul (**ACTIVITĂŢI, PRECEDENŢE DIRECTE**).

Această reprezentare se face printr-o REŢEA COORDONATOARE care, în mare, este un graf compus din PUNCTE (numite „NODURI") şi MUCHII ORIENTATE (numite şi ARCE).

Există două moduri de a pune în corespondenţă activităţile şi precedenţele directe dintr-un proiect cu nodurile şi arcele reţelei coordonatoare asociate:

A) Primul mod: $(A \circ A)$ **- Activitate ↔ ARC:**

- ARCELE corespund ACTIVITĂŢILOR ;
- NODURILE reprezintă MOMENTE DE TIMP (numite şi EVENIMENTE): de începere şi/sau de terminare ale unei activităţi sau ale mai multora.

B) Al doilea mod: $(A \circ N)$ **- Activitate ↔ NOD**

- NODURILE corespund ACTIVITĂŢILOR;
- ARCELE corespund PRECEDENŢELOR DIRECTE dintre activităţi.

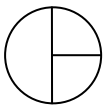
Exemplul 2.1: Pentru cele ce urmează vom lua ca exemplu de referinţă proiectul ridicării unei hale industriale, proiect dat prin următoarea listă de activităţi:

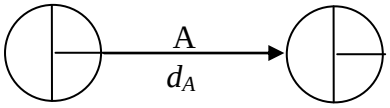
Tabelul 2.1

Nr. crt.	Denumirea activităţii	Cod	Activităţi directe precedente	Durata (luni)
1	Proiectarea halei	P	-	7
2	Obţinerea avizelor de execuţie	A	P	2
3	Comenzi utilaje	CU	P	3
4	Organizare de şantier	OS	P	3
5	Livrări utilaje	LU	CU	6
6	Execuţie reţele tehnice	R	A, OS	10
7	Execuţie drumuri interioare	D	R	5
8	Lucrări construcţii montaj	C	LU, R	14
9	Instruirea personalului	E	P	10

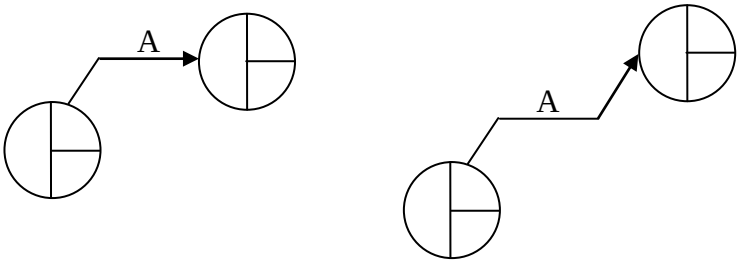
2.2. Construcţia reţelei coordonatoare în reprezentarea (A°A)

2.2.1. Reguli de reprezentare a activităţilor

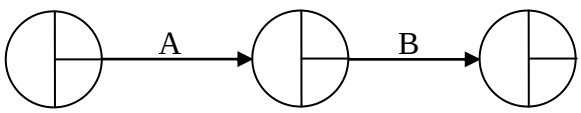
I) Un nod va fi reprezentat prin sigla: 

II) O activitate *A* cu durata d_A se va reprezenta prin sigla: 

Se recomandă ca sigla să fie ORIZONTALĂ sau să conţină o porţiune orientată SV-NE:

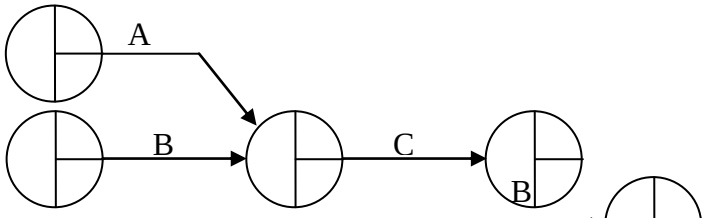


III) Regula fundamentală. Precedenţa directă: *A* precede direct *B* se va reprezenta prin:

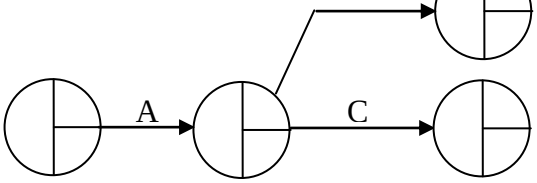


Variante:

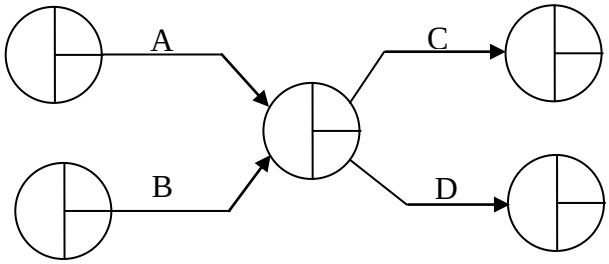
- *A, B* preced direct *C:*



- *A* precede direct pe *B* şi pe *C:*

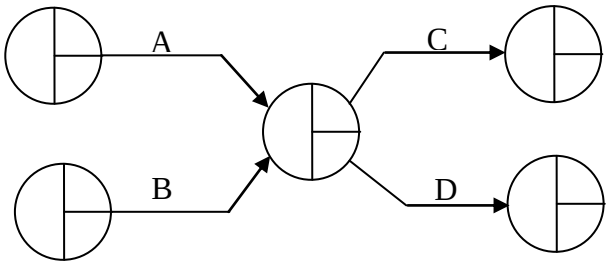


- *A* şi *B* preced direct *C* şi *D:*

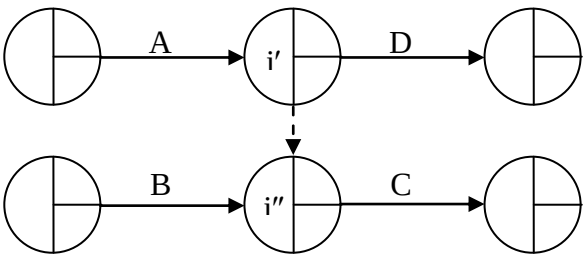


IV) Regula de evitare a falsei precedenţe. Considerăm situaţia $\begin{cases} A,B \ \ preced \ \ direct \ \ C \\ A \quad\ \ precede \ \ direct \ \ D \end{cases}$.

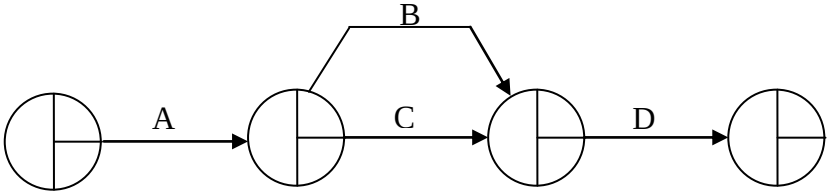
Reprezentând mai întâi primul „rând” de precedenţe (*A* şi *B* preced direct *C*) şi apoi al doilea (*A* precede direct *D*) obţinem desenul din figura alăturată. Reprezentarea este **INCORECTĂ** deoarece activitatea *D* nu este condiţionată de *B*.



Vom evita acest lucru introducând o ACTIVITATE FICTIVĂ cu durata ZERO (întotdeauna reprezentată prin linie punctată). Reprezentarea corectă se va obţine *desenând mai întâi precedenţa simplă* (*A* precede direct *D*) *şi apoi dubla precedenţă* (*A* şi *B* preced direct *C*).

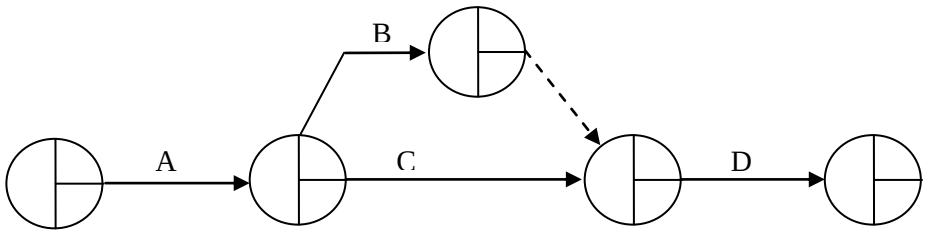


V) Regula de reprezentate a activităţilor paralele. Considerăm situaţia $\begin{cases} A \ \ precede \ \ direct \ \ B,C \\ B,C \ \ preced \ \ direct \ \ D \end{cases}$ care se poate reprezenta prin desenul:



Reprezentarea este CORECTĂ din punct de vedere logic dar NESATISFĂCĂTOARE pentru că nu permite identificarea unui arc prin perechea de noduri extremităţi (activităţile *B* şi *C* au acelaşi nod iniţial şi acelaşi nod final).

Pentru evitarea paralelismului dintre *B* şi *C* utilizăm din nou o ACTIVITATE FICTIVĂ cu durata ZERO. Se obţine în acest fel reprezentarea:



Această regulă poate fi enunţată astfel: două activităţi ale unui proiect pot avea în comun cel mult un o extremitate (nod).

2.2.2. Cum se desenează o reţea coordonatoare în reprezentarea A°A

1) Se începe prin fixarea pe hârtie a unui nod care semnifică ÎNCEPEREA EXECUŢIEI PROIECTULUI. Din acest nod vor pleca toate arcele corespunzătoare activităţilor INIŢIALE (sau activităţi INDEPENDENTE – realizarea lor NU este condiţionată de finalizarea altei-altor activităţi) ale proiectului.

2) Se vor reprezenta mai întâi activităţile SIMPLU condiţionate (condiţionate de către o singură activitate) de activităţile deja trasate, apoi cele dublu condiţionate, triplu condiţionate ş.a.m.d. până la epuizarea întregii liste de activităţi.

3) La introducerea unui nou arc în desen se va avea în vedere reprezentarea CORECTĂ a dependenţelor dintre activitatea introdusă şi activităţile DEJA reprezentate.

4) Se recomandă ca în momentul inserării unei noi activităţi în reţea SĂ NU SE DESENEZE ŞI NODUL FINAL al arcului corespunzător! Acesta poate fi nod final şi pentru alte activităţi ÎNCĂ NEREPREZENTATE !!

5) La epuizarea LISTEI DE ACTIVITĂŢI se poate constata că unul sau mai multe arce din desen NU AU NOD FINAL!! Aceste arce corespund activităţilor FINALE din proiect adică acelor activităţi care nu preced direct nici o altă activitate. Tuturor acestor arce li se va asigura un unic nod final cu semnificaţia de moment de TERMINARE A PROIECTULUI.

6) După desenarea reţelei se recomandă ca nodurile ei să fie NUMEROTATE pentru a putea identifica activităţile cu ajutorul lor. Nodul care semnifică începerea proiectului se numerotează de obicei cu *O*. **Numerotarea trebuie astfel făcută astfel încât nodul iniţial al unui arc să aibă un număr de ordine MAI MIC decât numărul de ordine al nodului final.** Numerotarea nu este în general unică.

7) CELE MAI BUNE INSTRUMENTE PENTRU TRASAREA UNEI REŢELE COORDONATOARE SUNT UN CREION ŞI O GUMĂ.

În figura 2.2 este dată reţeaua corespunzătoare proiectului de referinţă din tabelul 2.1, în prezentarea **A°A**:

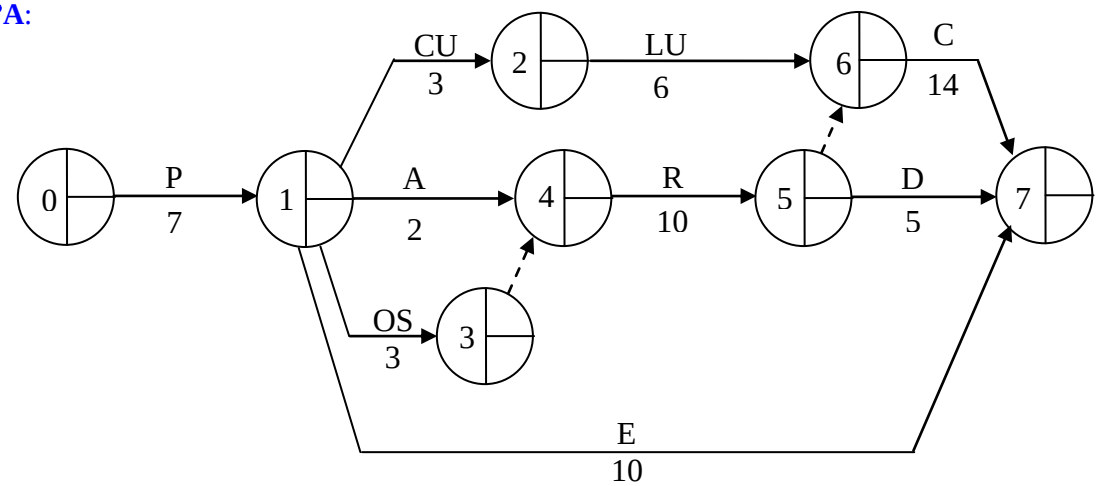


Figura 2.2

TEMĂ: Refaceţi reţeaua din figura 2.2 reprezentând activităţile proiectului secvenţial, utilizând regulile enunţate anterior.

NOTĂ: Reprezentarea **A°A** nu va fi discutată la curs/seminar, ea constituind o lectură opţională, destinată studenţilor care doresc să citească mai mult pe această temă. Conceptele introduse în această secţiune şi care vor fi utilizate şi la reprezentarea **A°N** sunt însă OBLIGATORII (reţea coordonatoare, termene minime/maxime, durata de realizare a proiectului, drum critic, activităţi critice, rezerva de timp, …).

2.2.3. Analiza reţelei coordonatoare în reprezentarea A°A

ANALIZA reţelei coordonatoare este un PROCES SIMPLU DE CALCUL prin care se obţin următoarele rezultate:

- **DURATA MINIMĂ DE EXECUŢIE A PROIECTULUI** ţinând cont NUMAI de DURATELE activităţilor şi de PRECEDENŢELE DIRECTE dintre acestea şi **fără** a se ţine seama de **resurse** sau **costuri.**
- **Activităţile CRITICE** şi **termenele de începere şi terminare ale acestora**. Activităţile critice sunt acele activităţi care condiţionează NEMIJLOCIT realizarea proiectului la termenul minim stabilit.
- **TERMENE EXTREME** şi **REZERVA TOTALĂ** pentru fiecare activitate NECRITICĂ. Termenele extreme sunt termenul CEL MAI DEVREME DE ÎNCEPERE (EET) şi termenul CEL MAI TÂRZIU DE TERMINARE (LET). **Rezerva totală** este **DIFERENŢA** dintre cele două termene extreme, din care **SE SCADE DURATA activităţii**.

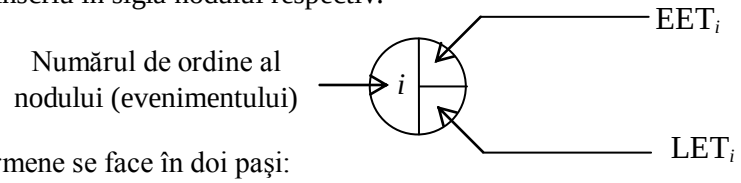
Reamintim că un nod al reţelei coordonatoare în reprezentarea A°A semnifică un MOMENT DE TIMP (EVENIMENT) în derularea proiectului care poate însemna ÎNCEPEREA şi/sau TERMINAREA unei activităţi sau a mai multora.

Fiecărui nod *i* se ataşează două TERMENE:

EET_i ≡ termenul cel mai devreme de producere (EARLIEST EVENT TIME) ≡ termenul minim la care pot începe activităţile care au ca nod INIŢIAL nodul *i*;

LET_i ≡ termenul cel mai târziu de producere (LATEST EVENT TIME) ≡ termenul maxim la care se pot termina activităţile care au ca nod FINAL nodul *i*.

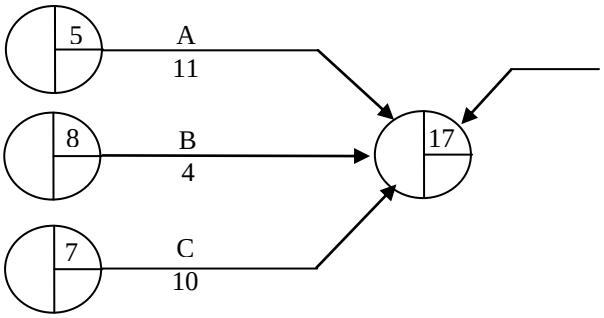
Aceste termene se înscriu în sigla nodului respectiv.



Calculul acestor termene se face în doi paşi:

I) În PASUL ÎNAINTE (FORWARD STEP) – de la nodul INIŢIAL către cel FINAL al reţelei – **se calculează EET** pentru toate nodurile reţelei după următoarea schemă ITERATIVĂ:

- pentru nodul INIŢIAL (care semnifică ÎNCEPEREA PROIECTULUI) se ia EET = 0;
- pentru celelalte noduri se aplică judecata rezultată din exemplul:



$$\max\{5+11,\ 8+4,\ 7+10\}$$

LOGIC, NU?

Un eveniment nu se poate produce MAI DEVREME de terminarea unei activităţi căreia îi serveşte drept moment de închidere!

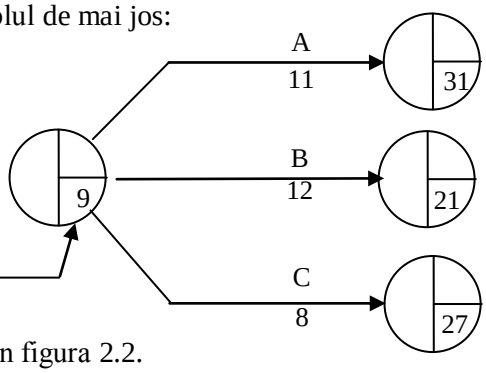
IMPORTANT: După executarea pasului înainte, **termenul timpuriu** de producere al nodului care semnifică terminarea proiectului arată **DURATA MINIMĂ DE EXECUŢIE A PROIECTULUI** = EET.

II) În PASUL ÎNAPOI (BACKWARD STEP) – de la nodul FINAL către cel INIŢIAL al reţelei – **se calculează LET** pentru toate nodurile reţelei după următoarea schemă ITERATIVĂ:

- pentru nodul FINAL (care semnifică TERMINAREA PROIECTULUI) se ia **LET = EET**
- pentru celelalte noduri se aplică judecata din exemplul de mai jos:

LOGIC, NU?
Un eveniment nu se poate produce MAI TÂRZIU de începerea unei activităţi căreia îi serveşte drept moment de start!

$$\min\{31-11,\ 21-12,\ 27-8\}$$



Exemplu: Vom efectua cei doi paşi pentru reţeaua din figura 2.2.

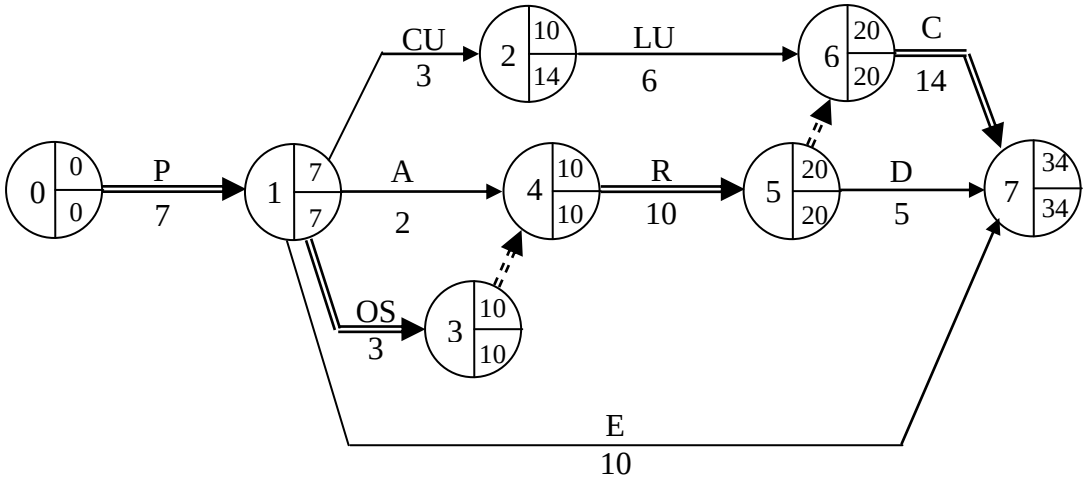


Figura 2.3

Concluzii:

1) Durata minimă de execuţie a proiectului este de 34 luni.

2) Nodurile reţelei se împart în două categorii:

- NODURILE CRITICE caracterizate de egalitatea celor două termene de producere timpuriu şi târziu. În cazul de faţă nodurile: 0, 1, 3, 4, 5, 6, 7.
- NODURI NECRITICE pentru care EFT ≠ LET. Exemplu: nodul 2.

3) Se numeşte **ACTIVITATE CRITICĂ** o activitate cu proprietăţile:

- **arcul reprezentativ este plasat între două noduri critice;**
- **diferenţa dintre termenele de producere ale evenimentelor extremităţi este egală cu durata activităţii.**

În cazul nostru activităţile critice sunt: P, OS, R şi C.

4) Suma duratelor activităţilor critice este egală cu **durata minimă de execuţie a proiectului.**

5) (Dacă inducem şi activităţile fictive) activităţile critice constituie un DRUM de la nodul INIŢIAL ≡ ÎNCEPEREA PROIECTULUI la cel FINAL ≡ TERMINAREA PROIECTULUI. Acest drum se numeşte **DRUM CRITIC** şi este mulţimea de activităţi SUCCESIVE cu CEA MAI MARE durată de execuţie. În concluzie:

Durata MINIMĂ de execuţie a unui proiect este egală cu MAXIMUL duratelor de execuţie ale tuturor mulţimilor de activităţi succesive ale proiectului:

$$d_c = d_P + d_{OS} + d_R + d_C = 34\,luni$$

Într-o reţea coordonatoare pot exista mai multe drumuri critice, toate având însă aceeaşi durată.

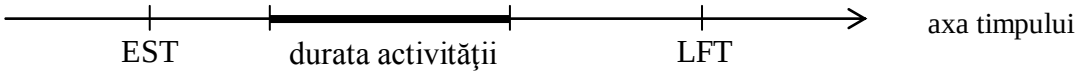
6) Din reţea rezultă – pentru fiecare ACTIVITATE – două termene EXTREME importante:

a) TERMENUL CEL MAI DEVREME DE ÎNCEPERE (abreviat EST ≡ EARLIEST STARTING TIME). EST al unei activităţi = EET al evenimentului (nodului) INIŢIAL al activităţii;
b) TERMENUL CEL MAI TÂRZIU DE TERMINARE (LFT ≡ LATEST FINISHING TIME). LFT al unei activităţi = LET al evenimentului (nodului) FINAL al activităţii.

Pe baza acestor termene, pentru orice activitate putem calcula:

- TERMENUL CEL MAI DEVREME DE TERMINARE (abreviat EFT ≡ EARLIEST FINISHING TIME): **EFT = EST + DURATA activităţii**;
- TERMENUL CEL MAI TÂRZIU DE ÎNCEPERE (abreviat LST = LATEST STARTING TIME): **LST = LFT – DURATA activităţii**.

7) Fiecărei activităţi a proiectului i s-a precizat un INTERVAL DE TIMP în care se poate realiza fără a încălca precedenţele sau termenul final al proiectului.

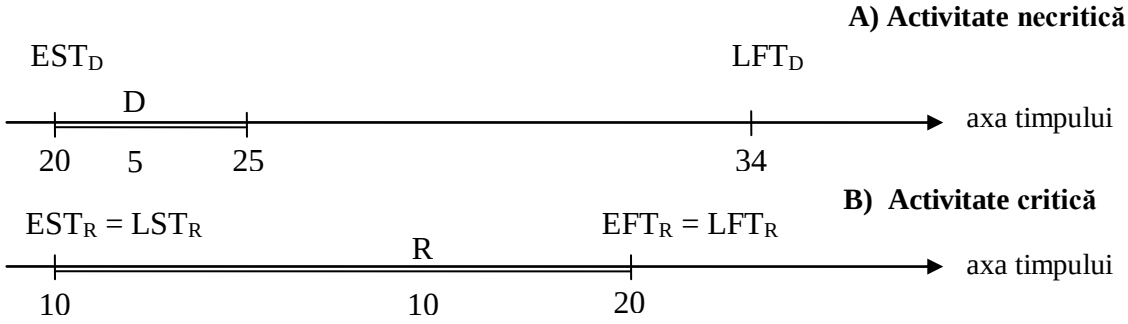


Definim:

REZERVA TOTALĂ a unei activităţi $\underset{def}{=}$ **LFT – EST – DURATA activităţii** = LFT- EFT **SAU** LST - EST

Rezultă imediat că rezerva totală a unei activităţi critice este ZERO.

Rezerva totală este INTERVALUL MAXIM cu care poate fi amânată o activitate de la termenul cel mai devreme de începere fără ca amânarea să afecteze termenul final al proiectului (durata minimă de execuţie a întregului proiect).



8) Pentru conducerea operativă, diferitele termene calculate mai sus se trec într-un document cu formatul:

Tabelul 2.2

ACTIVITATE	DURATA	TERMEN de ÎNCEPERE		TERMEN de TERMINARE		REZERVA TOTALĂ
		MINIM (EST)	MAXIM (LST)	MINIM (EFT)	MAXIM (LFT)	
P	7	0	0	7	7	**0**
A	2	7	8	9	10	1
CU	2	7	11	10	14	4
OS	3	7	7	10	10	**0**
LU	6	10	14	16	20	4
R	10	10	10	20	20	**0**
D	5	20	29	25	34	9
C	14	20	20	34	34	**0**
E	10	7	24	17	34	17

9) Datorită posibilităţii de introducere a unor noduri şi respectiv activităţi fictive reprezentarea prin grafic coordonator a unui proiect complex NU este UNICĂ. Unic în această reprezentare este doar termenul de FINALIZARE al întregului proiect (valoarea înscrisă prin cele două etape de marcaj în modul final al proiectului).

2.3. Construcţia reţelei coordonatoare în reprezentarea activitate – nod (A°N)

În reprezentarea **A°N nodurile** reţelei coordonatoare corespunde **activităţilor**, în timp ce **arcele** pun în evidenţă **precedentele directe** dintre activităţi.

Fie *P* = un proiect compus din activităţile *A, B, C.*

Reamintim că activitatea *A* precede direct activitatea *B* dacă *B* poate începe imediat după terminarea lui *A*.

Presupunem cunoscute toate relaţiile de precedenţă directă dintre activităţile proiectului *P*, precum şi duratele acestor activităţi. Pentru activitatea *A* notăm *d*(*A*) = durata activităţii *A*. Activităţile proiectului, dependenţele şi duratele activităţilor se regăsesc în **lista de activităţi**. Fiecărei activităţi i-au fost asociate următoarele termene:

- termenul cel mai devreme (timpuriu) de începere notat EST;
- termenul cel mai devreme de terminare, EFT;
- termenul cel mai târziu de începere, LST;
- termenul cel mai târziu de terminare, LFT.

În reprezentarea **A°N** fiecare activitate este reprezentată printr-un dreptunghi compartimentat ca în figura 2.4.

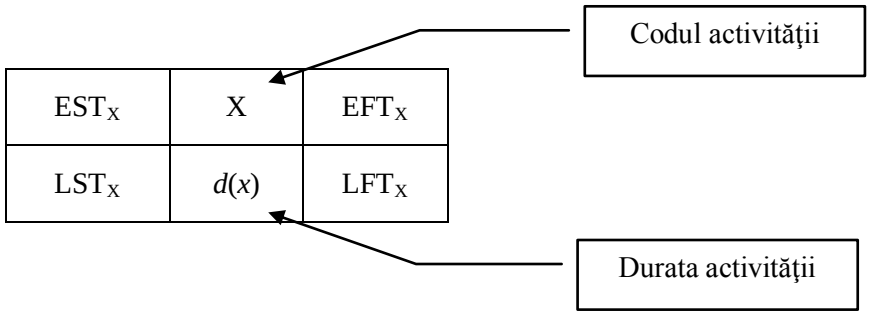


Figura 2.4

Faptul că activitatea *A* precede direct activitatea *B* va fi reprezentat precum în figura 2.5.

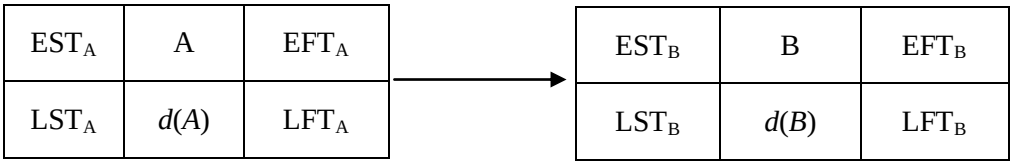


Figura 2.5

Asemănător cazului **A°A** , reţeaua **A°N** trebuie să aibă:

- un singur nod iniţial;
- un singur nod FINAL;

În vederea îndeplinirii acestei cerinţe:

- dacă **proiectul *P* are mai multe activităţi iniţiale** el se va completa cu **o activitate fictivă de START** cu durata 0 şi care precede direct toate activităţile iniţiale ale proiectului;
- dacă *P* are mai multe activităţi finale el se va completa cu o **activitate fictivă de FINISH** cu **durata 0** şi care este direct precedată de toate activităţile finale din proiect.

Exemplul 2.2: Să trasăm reţeaua **A°N** corespunzătoare proiectului dat prin lista din Tabelul 2.3.

Tabelul 2.3

Nr. crt.	Cod activitate	Activităţi direct precedente	Durata (luni)
1	P	-	5
2	A	P	3
3	CLU	P	9

4	OS	P	4
5	R	A, OS	10
6	D	R	5
7	CM	CLU, R	14
8	E	P	10

Aplicarea regulilor de reprezentare introduse anterior ne conduc la reţeaua coordonatoare din figura 2.6. La epuizarea listei de activităţi se constată că există 3 activităţi finale: CM, D şi E. Ca urmare, reţeaua a fost completată cu o activitate fictivă pe post de unică activitate finală a grafului coordonator.

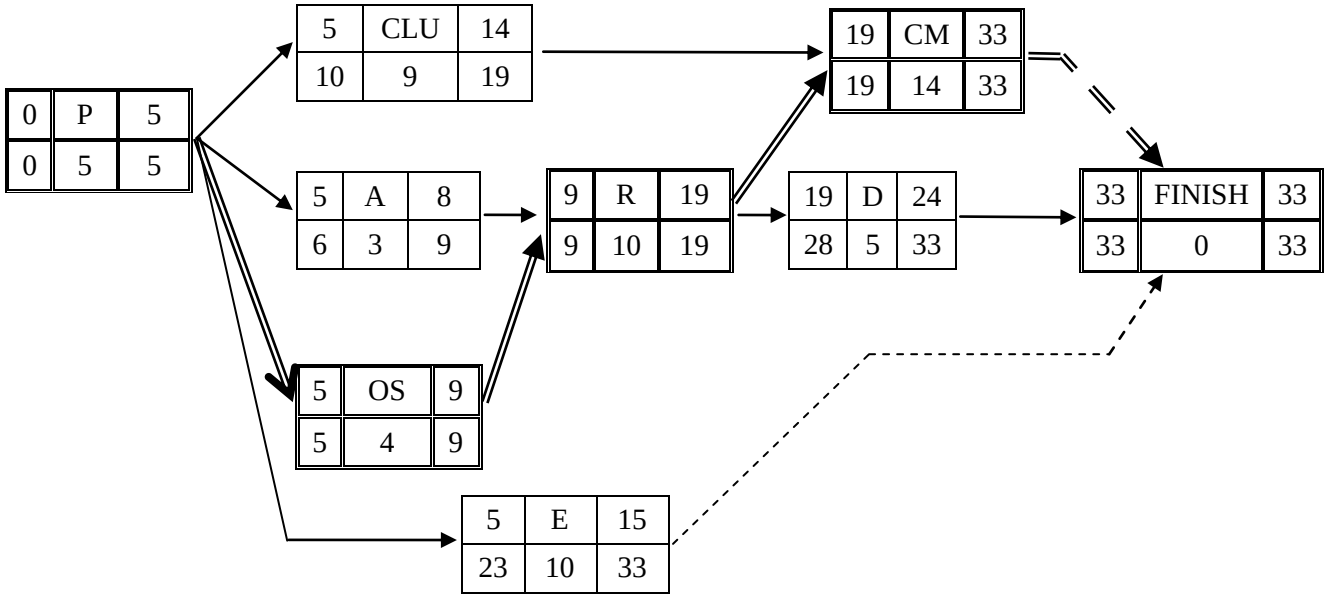


Figura 2.6

Calculul celor 4 termene s-a făcut la fel ca în reprezentarea **A°A** în doi paşi:

I. Pasul înainte, dinspre nodul iniţial către nodul final. La acest pas se calculează termenele MINIME de începere şi terminare a activităţilor:

- Pentru activitatea iniţială *I*: *EST* (I) = 0 şi *EFT* (I) = *EST* (I) + *d*(I) = *d*(I);
- Pentru o activitate diferită de activitatea iniţială:

$$EST(Y) = \max\{EFT(X)/\ X\ \ precede\ direct\ Y\}\ \text{şi}\ EFT(Y) = EST(Y) + d(Y)$$

II. Pasul înapoi, de la nodul final către cel iniţial. La acest pas se calculează termenele MAXIME (târzii) de începere şi terminare a activităţii.

- Pentru activitatea finală *F* (reală sau fictivă): *LFT*(*F*) = *EFT(F)*; *LST*(*F*) = *LFT*(*F*) – *d*(*F*);
- Pentru o activitate oarecare *X* diferită de activitatea finală *F*:

$$LFT(X) = \min\{LST(Y)/\ X\ \ precede\ direct\ Y\} si\ LST(X) = LFT(X) - d(X)$$

Justificarea formulelor a fost deja dată în cazul reprezentării **A°A**.

Concluzii:

1) Prin definiţie activitatea finală a proiectului are un singur termen de terminare EFT = LFT; aceasta va da termenul minim de execuţie al proiectului (în exemplul anterior 33 luni);

2) O activitate se va numi critică dacă termenele minime şi maxime de începere coincid, ceea ce este echivalent cu faptul că termenele de terminare minime şi maxime coincid.

În cazul de faţă, activităţile critice sunt: P, OS, R, CM.

Activităţile critice constituie o succesiune de activităţi care indică drumul critic. Suma duratelor activităţilor de pe drumul critic = **durata minimă** de execuţie a proiectului. Pe de altă parte **drumul critic este succesiunea de activităţi cu cea mai mare durată de execuţie**.

Reamintim că REZERVA totală a unei activităţi *X* este egală cu diferenţa dintre termenele de începere MAXIME şi MINIME ceea ce este tot una cu diferenţa dintre termenele de terminare MAXIME şi MINIME.

Rezerva unei activităţi critice este ZERO. **Rezerva totală este intervalul de timp cu care poate fi prelungită o activitate de la termenul ei cel mai devreme de începere fără a se perturba termenul final de terminare al proiectului**.

De exemplu: Activitatea CLU are o rezervă totală de 10 – 5 = 19 – 14 = 5 luni; aceasta înseamnă că de la momentul ei minim de începere 5, activitatea CLU poate fi prelungită cu maximum 5 luni fără perturbarea termenului final al întregului proiect.

Dacă o activitate a fost prelungită prin consumarea întregii sale rezerve totale, toate activităţile care o succed devin critice.

TEMĂ: Construiţi reţeaua coordonatoare în reprezentarea **A°N pentru proiectul dat în tabelul 2.1 şi reţeaua în reprezentarea A°A pentru proiectul dat prin lista de activităţi din tabelul 2.3.**

CURSUL 4

2.4. Alocarea resurselor în managementul proiectelor

2.4.1 Ce este alocarea resurselor?

De regulă, realizarea activităţilor unui proiect implică utilizarea unor resurse. Printre acestea **forţa de muncă**, de diferite calificări, ocupă un loc important. În general resursele sunt disponibile la un moment sau altul în cantităţi limitate. În aceste condiţii, în planificarea activităţilor trebuie să se ţină seama nu numai de precedenţele dintre activităţi şi durata acestora, dar şi de **necesitatea încadrării în disponibilul de resurse pe tot parcursul realizării proiectului**.

2.4.2 Ipoteze şi notaţii

Problema planificării activităţilor unui proiect în funcţie şi de resurse este extrem de complexă şi de aceea vom adopta următoarele ipoteze simplificatoare:

1. În oricare moment al derulării proiectului, **disponibilul dintr-o anumită resursă este acelaşi**;
2. Pe tot parcursul realizării unei activităţi **necesarul dintr-o resursă pentru această activitate este acelaşi**.

Notaţii:

$H \equiv$ mulţimea resurselor considerate a avea disponibil limitat;

b_h = disponibilul din resursa $h \in H$ asigurat şi constant pe toată durata de realizare a proiectului;

r_{ih} = necesarul de resurse h pentru activitatea i presupus constant pe toată durata de execuţie a activităţii i.

Presupunem că $r_{ih} \le b_h$.

Exemplul 2.3: Vom considera proiectul definit prin datele din tabelul 2.4:

Tabelul 2.4

Activitate	Activităţi direct precedente	Durata (zile)	Necesar resurse	
			R_1	R_2
A	-	1	2	2
B	-	3	2	3
C	B	5	1	3
D	A	4	1	2
E	A	5	1	1
F	C, D, E	3	3	3
G	C	1	1	2
			$b_1 = 3$	$b_2 = 5$

La realizarea acestui proiect se utilizează 2 categorii de forţă de muncă R_1 şi R_2. Pentru executarea activităţii F sunt necesari în fiecare din cele 3 zile ale duratei de execuţie 3 muncitori cu calificarea R_1 şi alţi 3 muncitori având calificarea R_2. În fiecare zi sunt disponibili: b_1 = 3 muncitori R_1 şi b_2 = 5 muncitori R_2 .

Alocarea resurselor înseamnă în acest caz:

Etapa 1: Determinarea duratei minime de realizare a proiectului fără a ţine seama de consumul de resurse;

Etapa 2: Planificarea în timp a activităţilor proiectului a.î. necesarul de resurse să se încadreze în disponibilul dat.

Etapa 1: Aşa cum am procedat în secţiunile anterioare, vom planifica activităţile proiectului fără a ţine seama de resurse. Există posibilitatea ca rezultatul obţinut să fie satisfăcător şi din punct de vedere al utilizării resurselor limitate.

Reprezentarea **A°A** conduce la reţeaua dată în figura 2.7.

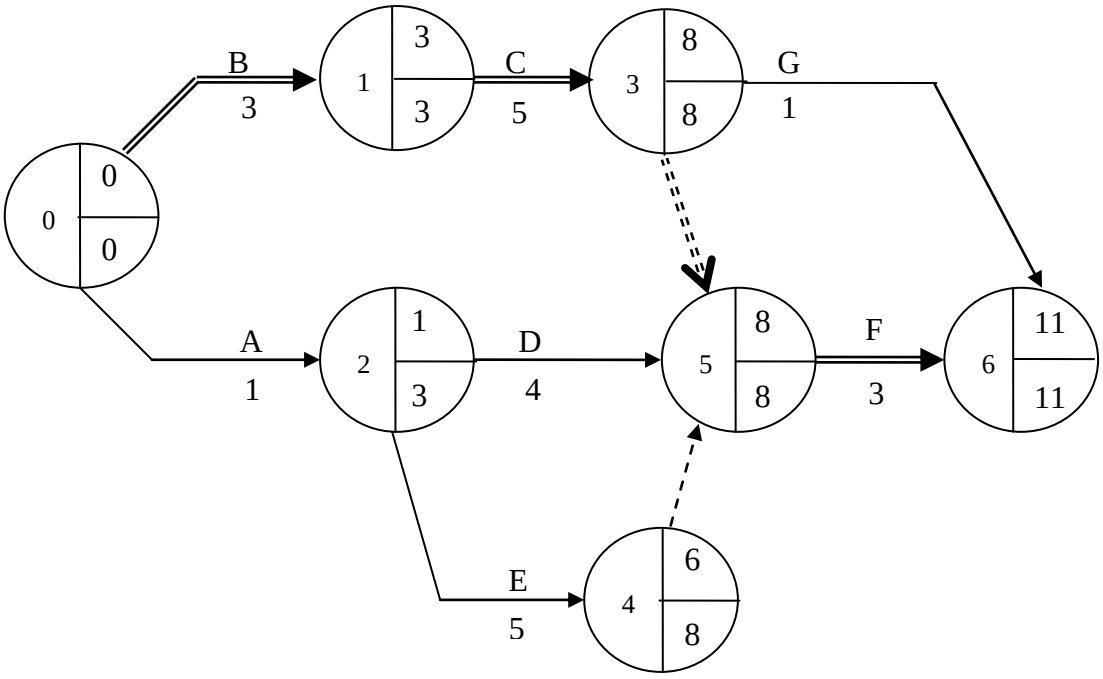


Figura 2.7

TEMĂ: Calculaţi durata maximă de execuţie a proiectului, termenele minime şi maxime de începere şi terminare a tuturor activităţilor, precum şi rezervele de timp ale activităţilor fără a ţine seama de resurse.

Etapa 2: Sunt suficiente resursele R_1 şi R_2 pentru a termina proiectul în 11 zile?

Vom răspunde la această întrebare trasând histogramele consumurilor de resurse necesare planificării tuturor activităţilor proiectului la termenele minime de începere (EST).

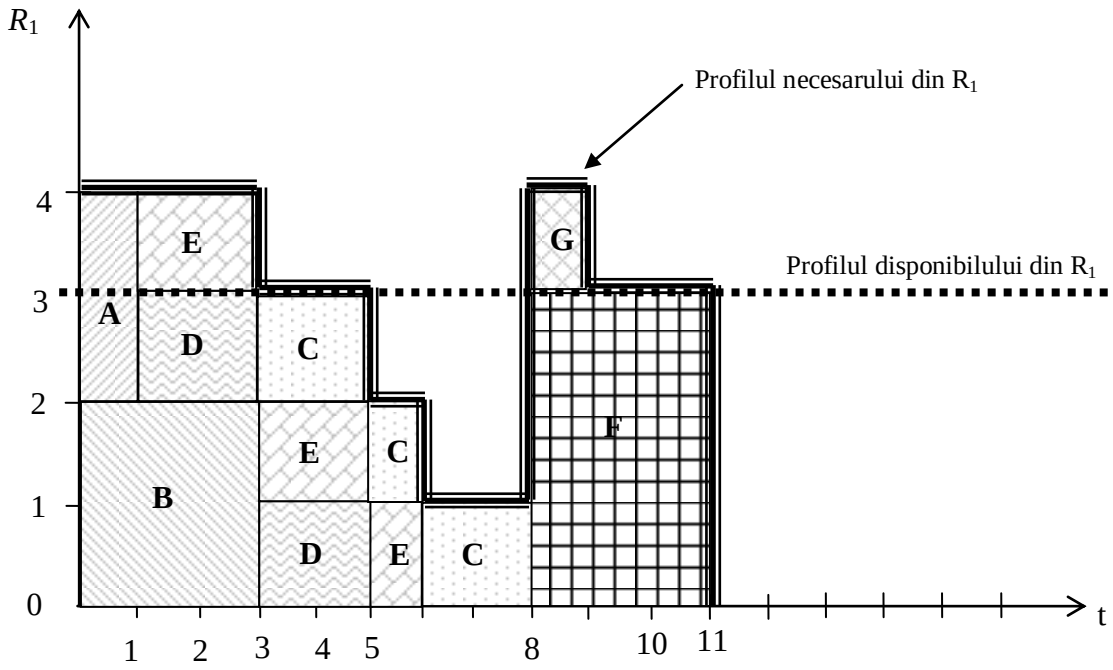


Figura 2.8

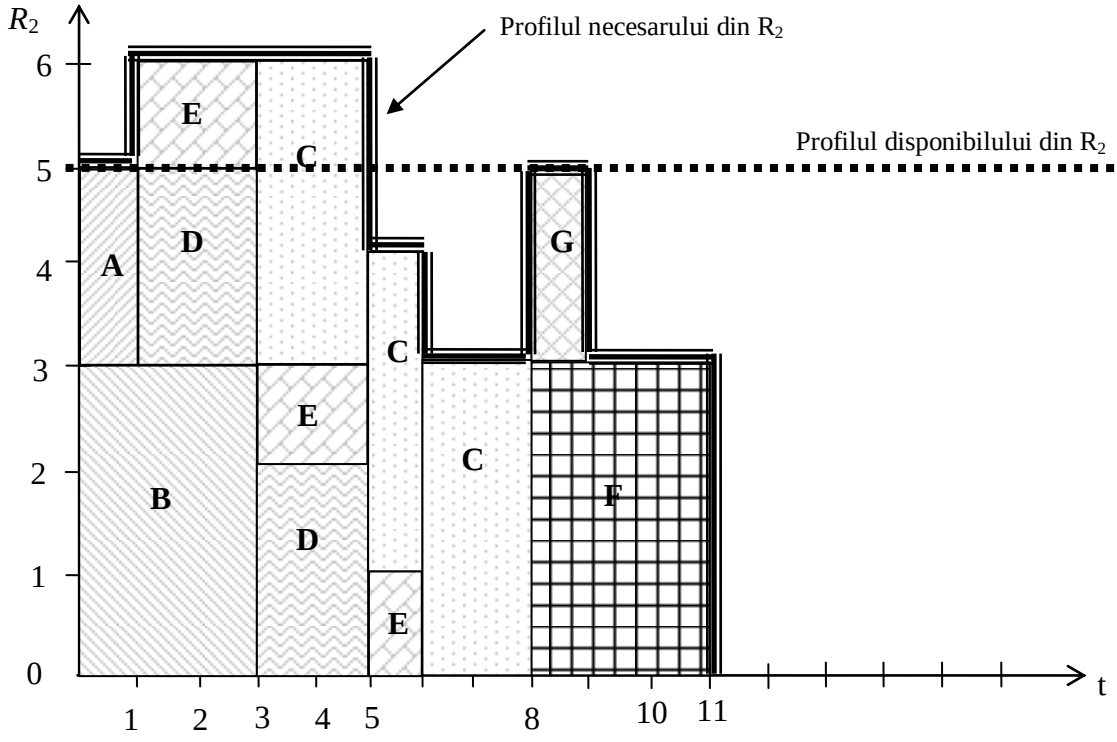


Figura 2.9

Aşa cum se observă din figurile 2.8 şi 2.9, atât pentru resursa R_1 cât şi pentru resursa R_2 disponibilele sunt insuficiente pentru a programa activităţile proiectului la termenele lor minime de începere. În cazul resursei R_1 în intervalele 0-3 şi 8-9 de timp avem o depăşire de o unitate de resursă (un lucrător cu calificarea 1 mai puţin), iar în cazul R_2 în intervalul 1-5 există o depăşire de o unitate de resursă (un lucrător cu calificarea 2 mai puţin).

O depăşire a disponibilului de resursă se va numi în continuare **conflict de resursă**.

2.4.3 Rezolvarea unui conflict de resurse

În cazul în care la un moment dat apare depăşirea necesarului faţă de disponibil la una sau mai multe din resursele utilizate de către proiect, singura cale de rezolvare a conflictului de resurse constă în amânarea unor activităţi care pot începe la momentul respectiv.

Operaţia de amânare este delicată din mai multe motive:

- amânarea unor activităţi poate antrena amânarea altor activităţi care depind de prima;
- rezolvarea unui conflict de resursă la un anumit moment, poate modifica profilul necesarului pentru un moment viitor determinând apariţia unor conflicte noi.

Pentru a găsi soluţia corectă (optimă) ar trebui examinate toate posibilităţile de amânare care pot rezolva un conflict de resurse dat. Fiecare modalitate de amânare are propriile ei conflicte de resursă şi rezolvarea fiecăreia dintre ele generează noi alternative care trebuie studiate. Rezultă de aici un proces stufos şi greu de controlat.

În această situaţie ne vom mulţumi cu determinarea unor soluţii nu neapărat optime, dar care sunt, subiectiv vorbind, satisfăcătoare (acceptabile) şi care au avantajul că se obţin uşor şi în timp util.

Un procedeu care conduce la o soluţie bună, nu neapărat optimă, poartă numele de **euristică**. În esenţă, o euristică pentru rezolvarea unei probleme de optimizare *P* este un procedeu iterativ de determinare a unei soluţii a problemei *P, procedeu care la fiecare pas încearcă să facă cea mai bună alegere.*

În general, optimizarea locală nu conduce decât în anumite cazuri la soluţia optimă globală, astfel că rezultatul aplicării unei euristici este, de regulă, o soluţie **suboptimală** destul de apropiată de soluţia optimă căutată.

2.4.4 O euristică pentru problema alocării

Concret, rezultatul euristicii de alocare a resurselor constă în faptul că în rezolvarea unui conflict de resurse nu se examinează toate alternativele de amânare, ci numai una dintre ele construită pe baza unui criteriu de prioritate la amânare cum ar fi de exemplu:

- **are prioritate la amânare activitatea cu cea mai mare rezervă de timp** sau **activitatea cu durata cea mai mică**.
- **activităţile amânate se aleg în aşa fel încât cele alese pentru programare să asigure o utilizare cât mai bună a resurselor.**

Aceste criterii se utilizează în practică fie separat, fie în combinaţie. Pentru descrierea schemei generale a acestei euristici avem nevoie de următoarele notaţii:

a) Fiecare activitate va avea ataşat un **termen potenţial de începere (TPI)** care se poate modifica pe parcurs. La **START** pentru fiecare activitate se va lua **TPI = EST**.

b) Pe parcursul derulării proiectului activităţile de împart în:

- activităţi deja programate să se desfăşoare în funcţie de consumul de resurse;
- activităţi încă neprogramate.

În momentul în care o activitate este declarată PROGRAMATĂ termenul ei potenţial de începere devine **termen definitiv** de începere şi nu va mai putea fi modificat în etapele ulterioare ale algoritmului.

c) Programarea activităţilor ţinând cont de resurse se face la anumite momente de timp. Momentul curent notat *t* este **minimul** termenelor potenţiale de începere a activităţilor încă neprogramate. La start t_{min}=0.

În raport cu momentul curent de programare *t* , activităţile deja PROGRAMATE se împart în:

- terminate (≡ termenul definitiv de terminare ≤ *t*) şi
- activităţi în curs de desfăşurare (≡ termenul definitiv de terminare > *t*).

Printre activităţile încă neprogramate vom distinge activităţi ***candidate la programare*** (activităţile al căror termen potenţial de începere este egal cu *t*).

d) Notăm cu $\mathcal{A}$ = mulţimea de *activităţi în curs de desfăşurare* ***sau*** *candidate la programare* la momentul curent de programare *t*.

La fiecare moment de programare *t* vom compara disponibilul de resurse cu necesarul pentru susţinerea activităţilor din lista $\mathcal{A}$. Diferenţa $\Delta_h = b_h - \sum_{i \in A} r_{ih},\ h \in H$ se numeşte **indicator de amânare relativ la resursa *h***.

Dacă $\Delta_h \geq 0\ (\forall)\ h \in H$ avem suficiente resurse pentru susţinerea tuturor activităţilor din $\mathcal{A}$, în caz contrar, unele activităţi vor fi amânate.

Prezentăm în continuare etapele euristicii problemei de alocare a resurselor.

START: Trasăm reţeaua coordonatoare a proiectului şi determinăm termenele minime de începere *EST* ale tuturor activităţilor. Trasăm histograma necesarului din fiecare resursă considerând că toate activităţile încep la termenele minime rezultate din reţeaua coordonatoare. **Dacă profilul necesarului din fiecare resursă nu depăşeşte disponibilul STOP**. În caz contrar iniţializăm:

- pentru fiecare activitate termenul potenţial de începere *TPI = EST;*
- momentul curent de programare $t = 0$;
- toate activităţile se declară neprogramate.

Conţinutul unei ITERAŢII:

Pas 1: Se identifică mulţimea $\mathcal{A}$ a activităţilor în curs de desfăşurare la momentul t şi candidate la programare la momentul t.

Pas 2: Pentru activităţile din lista $\mathcal{A}$ se calculează indicatorii de amânare:

$$\Delta_h = b_h - \sum_{i \in A} r_{ih},\ h \in H$$

- **Dacă** $\Delta_h \geq 0, \forall\, h \in H$ se actualizează $\mathcal{A}' = \mathcal{A}$, şi $\mathcal{B} = \varnothing$ ($\mathcal{B}$ = lista activităţilor amânate la momentul t) şi se trece la pasul 4, unde $\mathcal{A}'$ = lista activităţilor programate sa înceapă la momentul t.
- **Dacă** pentru o resursă $h \in H$ avem $\Delta_h < 0$ se trece la pasul 3.

Pas 3: Se identifică mulţimea $\mathcal{B}$ a activităţilor ce trebuie amânate pe baza criteriilor de prioritate la amânare avute în vedere. Se actualizează $\mathcal{A}' = \mathcal{A} - \mathcal{B}$.

Pas 4: Se programează activităţile candidate din $\mathcal{A}'$ să înceapă **definitiv** la momentul t (în cazul în care există asemenea activităţi).

Pas 5: Tuturor activităţilor amânate din $\mathcal{B}$ în caz că $\mathcal{B} \neq \varnothing$ li se fixează un nou *TPI* egal cu minimul termenelor definitive de terminare ale activităţilor din $\mathcal{A}'$. Dacă este cazul se va modifica şi *TPI* al activităţilor care depind direct sau indirect de activităţile amânate. Se actualizează $t \equiv$ MINIMUL termenelor potenţiale de începere al activităţilor încă neprogramate după care se revine la Pasul 1 în cadrul unei noi iteraţii.

Ilustrare numerică

Vom aplica euristica prezentată anterior proiectului dat în lista din tabelul 2.4.

În operaţia de amânare vom utiliza ca şi criteriu principal *criteriul rezervei totale de timp* combinat cu *modul de încadrare în disponibilul de resursă*.

Rezultatele programării sunt date în tabelul 2.5.

Tabelul 2.5

Activităţi	Durata	Rezerva de timp condiţională	Termen definitiv de		Termene potenţiale de începere la ITERAŢIA				
			Începere	Terminare	IT. 1 $t_{min} = 0$	2 $t_{min} = 3$	3 $t_{min} = 4$	4 $t_{min} = 8$	5 $t_{min} = 12$
A	1	2, -1	3	4	0	3*	-	-	-
B	3	0	0	3	0*	-	-	-	-
C	5	0	3	8	3	3*	-	-	-
D	4	3, 0, -4	8	12	1	4	4	8*	-
E	5	2, -1	4	9	1	4	4*	-	-
F	3	0, -1, -4	12	15	8	9	9	12	12*
G	1	2	8	9	8	8	8	8*	-

IT. 1: $t_{min} = 0$

Pas 1: $\mathcal{A}_1 =$ {A, B}. $\Delta_1 = 3 - (2+2) < 0$, $\Delta_2 = 5 - (2+3) = 0$

Pas 2: Avem $\Delta_1 < 0 \Rightarrow$ se trece la pasul 3.

Pas 3: Se observă uşor că pentru rezolvarea conflictului de resurse este suficientă amânarea uneia din cele 2 activităţi. Conform celor convenite din START va avea prioritate la amânare rezerva de timp cea mai mare: $R_{(A)} = 3 - (0+1) = 2$, $R_{(B)} = 3 - (0+3) = 0$.

Se amână activitatea A (în notaţiile generale $\mathcal{B}_1 = \{A\}$ şi $\mathcal{A}_1' = \mathcal{A}_1 - \mathcal{B}_1 = \{B\}$.

Pas 4: Se va programa *B* să înceapă la momentul 0. Trecem acest termen în tabel în coloana „termene definitive de începere" după care calculăm şi termenul definitiv de terminare. În coloana IT. 1 trecem * în dreptul activităţii programate B (din acest moment cu activitatea B nu se mai lucrează).

Pas 5: Activităţii A i se fixează un nou *TPI* egal cu minimul termenelor definitive de terminare ale activităţilor din $\mathcal{A}_1'$. TPI_A = termenul la care se termină activitatea programată *B* = 3; trecem termenul în coloana Iteraţia 1.

Reţeaua coordonatoare arată faptul că activităţile *D* şi *E* (condiţionate de către A) nu pot începe acum mai devreme de termenul de terminare al lui A, $3 + 1 = 4 \Rightarrow$ completăm coloana iteraţia 2 cu $TPI_D = TPI_E = 4$.

Amânarea activităţii *E* implică amânarea activităţii *F* la termenul $TPI_F = 4 + 5 = 9$.

Activităţile neprogramate *C* şi *G* îşi menţin *TPI* ele nefiind condiţionate de activităţile amânate anterior.

Coloana Iteraţia 1 este acum completă.

Iteraţia 2: $t = 3$

.
.
.

TEMĂ: Parcurgeţi iteraţiile 2, 3, 4 şi 5 ale rezolvării exemplului anterior regăsind datele din tabelul 2.5. Trasaţi apoi histogramele pentru R_1 şi R_2 şi comparaţi-le cu cele din figurile 2.8 şi 2.9.

CAPITOLUL 3
FLUX ÎN REŢELE DE TRANSPORT

NOTĂ: Suportul de curs al Capitolelor 1, 2 şi 3 are la bază lucrările:

- Ciobanu, Gh., Nica, V., Mustaţă, Fl., Mărăcine, V. Mitrut, D. (2002), ***Cercetări operaţionale. Optimizări în reţele. Teorie şi aplicatii economice***, Editura MATRIX ROM, Bucureşti;
- Ciobanu, Gh., Nica, V., Mustaţă Fl., Mărăcine, V. (1996), ***Cercetări operaţionale cu aplicaţii în economie. Teoria grafurilor şi Analiza drumului critic***, Editura MATRIX-ROM, Bucureşti

3.1 Reţea de transport. Flux într-o reţea

Într-o mare varietate de situaţii se pune problema deplasării unei cantităţi dintr-o resursă, care poate fi materie primă, produse finite, energie, informaţie etc., din unele locuri numite *surse* în alte locuri numite *destinaţii*. Această deplasare se efectuează pe anumite *rute de legătură*. Rutele de transport pot avea şi alte puncte comune în afara surselor şi destinaţiilor care vor fi numite *puncte intermediare*. Pot exista şi legături directe între surse şi/sau destinaţii. Ansamblul acestor rute, surse şi destinaţii precum şi a altor puncte comune rutelor se poate formaliza utilizând noţiunea de graf finit, fără bucle şi conex.

Fie $G = (X, U)$ un graf finit, conex şi fără bucle.

Graful G este numit ***reţea de transport*** dacă:

a) există un nod şi numai unul $s \in X$, numit *nod de intrare în reţea* sau *nod sursă*;
b) există un nod şi numai unul $t \in X$, numit *nod de ieşire din reţea* sau *nod destinaţie* (ţintă); orice alt nod $i \neq s$ şi $i \neq t$ este un *nod intermediar*;
c) fiecare arc $u = (i, j)$ este valorizat cu o valoare nenegativă, notată $c(u)$ sau c_{ij}, numită *capacitatea arcului u.*

Fie U_x^- mulţimea arcelor incidente spre interior în nodul x şi U_x^+ mulţimea arcelor incidente spre exterior în nodul x. Arcele $u \in U_s^+$ sunt *arce de intrare* în reţea iar $u \in U_t^-$ sunt *arce de ieşire* din reţea.

Definim prin *flux într-o reţea de transport* $G = (X, U)$ de la s la t o aplicaţie $\varphi : U \to \rceil$ indexată de toate rutele orientate ale reţelei cu următoarele condiţii:

P_1) $0 \le \varphi(u) \le c(u)$ oricare $u \in U$;

P_2) $\sum_{u \in U_x^-} \varphi(u) = \sum_{u \in U_x^+} \varphi(u)$ oricare $x \in X - \{s, t\}$, ceea ce înseamnă că în nodurile intermediare ale reţelei nu există consum sau pierdere de unităţi de flux.

Valoarea $\varphi(u)$ este fluxul propagat pe ruta $u = (i, j)$ sau fluxul pe arcul u şi reprezintă cantitatea de resursă care parcurge arcul u de la i la j, acesta fiind sensul orientării permise a arcului. Cantităţile de resursă care se propagă în reţeaua G intră prin nodul s şi ies din reţea prin nodul t. Unităţile indivizibile din resursă care sunt propagate de-a lungul rutelor între surse şi destinaţii sunt numite *unităţi de flux*. Capacităţile asociate arcelor sunt limitări ale volumului de flux care este propagat în reţea. În cazul de faţă, capacităţile sunt *limitări superioare*.

Formal, conform proprietăţii P_1 se exprimă cerinţa ca, pe fiecare rută, fluxul să fie pozitiv şi să nu depăşească capacitatea arcului. *Valorile capacităţilor $c(u) \ge 0$, $u \in U$ sunt presupuse, în continuare, numere întregi, acesta fiind cazul frecvent în aplicaţiile practice*. Proprietatea P_2 defineşte legea conservării fluxului (prima lege a lui Kirckhoff) prin care, exceptând sursa şi destinaţia, cantitatea de materie care intră într-un nod x prin arcele incidente spre interior este egală cu cantitatea care iese prin arcele incidente spre exterior din acelaşi nod x, lucru care face ca, valoarea fluxului propagat într-o reţea să poată fi măsurat fie la intrarea s, fie la ieşirea t a reţelei şi atunci, din proprietatea P_2, rezultă:

$$\sum_{u \in U_t^-} \varphi(u) = \sum_{u \in U_s^+} \varphi(u) = v(\varphi) . \qquad (3.1)$$

Conform relaţiei (3.1) cantitatea de resursă care iese din nodul sursă se regăseşte, în întregime, în nodul destinaţie t. Valoarea comună $v(\varphi)$ se numeşte *valoarea fluxului* φ în reţeaua de transport.

Să considerăm cazul în care reţeaua de transport este un graf oarecare $G = (X, U)$, U fiind mulţimea muchiilor. Definiţia fluxului permite existenţa simultană a unor fluxuri pe ambele sensuri de parcurgere a unei muchii, cu condiţia ca respectivele sensuri să fie permise. Putem introduce, astfel, conceptul de *flux net* ca fiind fluxul φ care satisface condiţia suplimentară:

P_3) dacă pe o muchie $[i, j]$ ambele sensuri de parcurgere (i, j) şi (j, i) sunt permise, atunci cel mult unul dintre fluxurile φ_{ij} şi φ_{ji} este nenul ($\varphi_{ij} \cdot \varphi_{ji} = 0$).

În general, două fluxuri φ şi φ' sunt *echivalente* dacă au aceleaşi valori: $v(\varphi) = v(\varphi')$. Plecând de la un flux oarecare φ definim în G un nou flux φ' de la s la t prin următoarea formulă:

$$\text{pentru orice } [i, j] \in U,\ \varphi'_{ij} = \begin{cases} \max(0, \varphi_{ij} - \varphi_{ji}) & \text{dacă } (i, j) \text{ şi } (j, i) \text{ sunt rute permise} \\ \varphi_{ij} & \text{dacă doar } (i, j) \text{ este rută permisă} \end{cases} \quad (3.2)$$

Aplicaţia $\varphi' = (\varphi'_{ij})$ definită de (3.2) este un flux deoarece satisface condiţiile P_1), P_2) şi P_3) şi este un *flul net* echivalent cu fluxul iniţial φ, având aceeaşi valoare cu acesta. În concluzie, pentru orice flux există un flux net echivalent.

Un arc $u \in U$ pentru care $\varphi(u) = c(u)$ este numit *arc saturat* şi, respectiv, un drum de la *s* la *t* care conţine cel puţin un arc saturat este un *drum saturat*. Un flux în reţeaua de transport cu toate drumurile saturate este numit *flux complet*.

Fie μ un drum nesaturat de un flux φ între *s* şi *t* şi

$$\theta = \min_{u \in \mu} (c(u) - \varphi(u)). \quad (3.3)$$

atunci

$$\varphi_1(u) = \begin{cases} \varphi(u) + \theta & \text{dacă } u \in \mu \\ \varphi(u) & \text{dacă } u \notin \mu \end{cases} \quad (3.4)$$

este noul flux φ_1 care saturează cel puţin un arc pe drumul μ.

Dacă se procedează la fel cu toate drumurile nesaturate de la *s* la *t* se obţine un flux complet. Într-o reţea de transport convenim să notăm capacităţile pe fiecare arc, de obicei, între paranteze iar alăturat fluxul (cumulat) ce străbate arcul respectiv.

În reţeaua de transport din figura 3.1 se pun în evidenţă următoarele drumuri de la *s* la *t* şi corespunzător cantitatea de flux propagată calculată cu formulele (3.3). Iniţial, fluxul este nul.

$\mu_1 = (s, 1, 2, t)$, $\theta_1 = \min(7, 4, 8) = 4$, $v(\varphi) = 4$;

$\mu_2 = (s, 1, 3, 2, t)$, $\theta_2 = \min(3, 2, 5, 4) = 2$, $v(\varphi) = 4+2 = 6$;

$\mu_3 = (s, 3, 2, t)$, $\theta_3 = \min(6, 3, 2) = 2$, $v(\varphi) = 6+2 = 8$;

$\mu_4 = (s, 3, t)$, $\theta_4 = \min(4, 6) = 4$, $v(\varphi) = 8+4 = 12$.

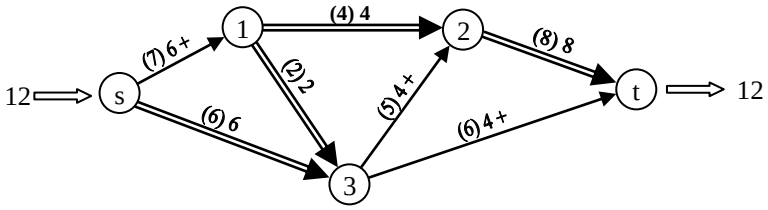


Figura 3.1

Pentru această alegere, pe primul drum μ_1 s-au propagat 4 unităţi de flux, pe al doilea drum μ_2 s-au propagat 2 unităţi de flux, pe drumul μ_3 s-au propagat 2 unităţi de flux şi pe drumul μ_4 s-au propagat 4 unităţi de flux. În total, fluxul propagat în reţea este $v(\varphi) = \theta_1 + \theta_2 + \theta_3 + \theta_4 = 12$ unităţi. Fluxurile propagate pe fiecare arc sunt cumulate şi scrise după capacităţi. Arcele saturate sunt: (1, 2) prin drumul μ_1, (1, 3) prin drumul μ_2, (2, *t*) prin drumul μ_3 şi (*s*, 3) prin drumul μ_4. Celelalte arce sunt nesaturate. Opţional, pe arcele nesaturate se scrie şi semnul (+).

Dacă reţeaua de transport conţine şi muchii neorientate (muchii), în identificarea drumurilor între *s* şi *t* vom avea grijă ca o rută neorientată ***să fie folosită doar într-un singur sens***. Astfel, odată cu identificarea unui drum, se orientează şi rutele.

În legătură cu propagarea unui flux într-o reţea de transport se pune problema dacă reţeaua de transport este capabilă să permită acoperirea integrală a cererilor în punctele de destinaţie. Aceasta va implica rezolvarea problemei *determinării volumului maxim de unităţi de flux ce pot fi deplasate de la surse la destinaţii*. În acest sens, se pune următoarea problemă numită ***problema fluxului de valoare maximă***:

Fiind dată o reţea de transport $G = (X, U)$ *şi capacităţile arcelor* $c(u)$, $u \in U$, *să determinăm un flux* φ *astfel încât*:

1°) *să se verifice condiţiile* P_1) *şi* P_2);

2°) $v(\varphi)$ *să fie o valoare maximă*.

Problema fluxului de valoare maximă într-o rețea de transport cu intrarea *s* şi ieşirea *t* se poate formula şi ca o problemă de programare liniară (vezi Capitolul 4):

$$\begin{aligned} &(\max) v \\ &\sum_j \varphi_{ji} - \sum_j \varphi_{ij} = 0 \quad \text{dacă } i \neq s \text{ şi } i \neq t \\ &\sum_j \varphi_{js} - \sum_j \varphi_{sj} = -v \\ &\sum_j \varphi_{jt} - \sum_j \varphi_{tj} = v \\ &0 \leq \varphi_{ij} \leq c_{ij} \text{ oricare } i \text{ şi } j \\ &\varphi_{ij} \text{ întreg} \end{aligned}$$

unde prin φ_{ij} s-a notat fluxul pe arcul (i, j).

Particularităţile combinatorii ale problemei, numărul mare de restricţii şi variabile, dificultăţile legate de integritatea variabilelor fac ineficientă rezolvarea cu tehnicile programării liniare, fiind necesară dezvoltarea unor metode specifice de rezolvare.

5.2 Tăietură într-o reţea

Fie $G = (X, U)$ o reţea de transport, cu *s* nodul–sursă şi *t* nodul–destinaţie, prin care se propagă un flux φ. Definim în *G* o partiţie a nodurilor, notată (S, T), unde $S \cup T = X$ şi $S \cap T = \varnothing$ şi astfel încât $s \in S$ şi $t \in T$.

Prin *tăietură* în reţeaua *G* vom înţelege mulţimea:

$$(S, T) = \{(i, j) \in U / i \in S \text{ şi } j \in T\} \qquad (3.5)$$

adică mulţimea rutelor orientate cu originea în *S* şi terminaţia în *T*. O tăietură are *capacitatea*, notată cu $c(S, T)$ dată de suma capacităţilor arcelor sale:

$$c(S, T) = \sum_{u \in (S,T)} c(u) \qquad (3.6)$$

De exemplu, în reţeaua din figura 3.2 tăietura care separă submulţimea $S = \{s, 1, 3\}$ de submulţimea $T = \{2, t\}$ este definită de linia punctată, deci:

$$(S, T) = \{(s, 2) , (1, 2) , (3, 2) , (3, t)\}$$

cu capacitatea:

$$c(S, T) = c_{s2} + c_{12} + c_{32} + c_{3t} = 26$$

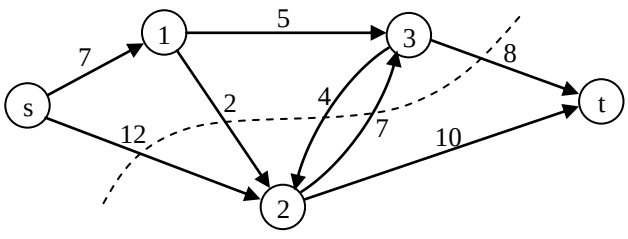


Figura 3.2

În general, într-o reţea de transport se poate presupune că există arce care intră în sursă, respectiv care ies din destinaţie, dar în acelaşi timp, nu este recomandabilă utilizarea acestor arce. În acest sens putem considera două tăieturi prticulare (S, T) în care $S = \{s\}$ şi $T = \{\text{noduri} \neq s\}$ sau $S = \{\text{noduri} \neq t\}$ şi $T = \{t\}$.

Teorema 3.1 *Valoarea oricărui flux într-o reţea este mai mică sau egală decât capacitatea oricărei tăieturi în reţea*.

Demonstraţie: Fie φ un flux şi (S, T) o tăietură în reţea. Pentru simplificarea notaţiilor un nod x_k este precizat prin indicele k. Pentru sursa s avem:

$$\sum_{i\in U_s^+}\varphi_{si} - \sum_{j\in U_s^-}\varphi_{js} = \sum_{i\in U_s^+}\varphi_{si} = v(\varphi). \qquad (3.7)$$

Pentru un nod $k \neq s$, cu $k \in S$, avem, conform proprietăţii P_2):

$$\sum_{i\in U_k^+}\varphi_{ki} - \sum_{j\in U_k^-}\varphi_{jk} = 0. \qquad (3.8)$$

Din (3.7) şi (3.8) obţinem valoarea fluxului:

$$v(\varphi) = \sum_{k\in S}\left(\sum_{i\in U_k^+}\varphi_{ki} - \sum_{j\in U_k^-}\varphi_{jk}\right) = \sum_{k\in S, i\in U_k^+}\varphi_{ki} - \sum_{k\in S, j\in U_k^-}\varphi_{jk} =$$

$$= \left[\sum_{k\in S, i\in S}\varphi_{ki} + \sum_{k\in S, i\in T}\varphi_{ki}\right] - \left[\sum_{k\in S, j\in S}\varphi_{jk} + \sum_{k\in S, j\in T}\varphi_{jk}\right].$$

Deoarece:

$$\sum_{k\in S, i\in S}\varphi_{ki} = \sum_{k\in S, j\in S}\varphi_{jk}$$

avem:

$$v(\varphi) = \sum_{k\in S, i\in T}\varphi_{ki} - \sum_{k\in S, j\in T}\varphi_{jk}. \qquad (3.9)$$

Întrucât $\sum_{k\in S, j\in T}\varphi_{jk} \geq 0$ avem inegalităţile:

$$v(\varphi) \leq \sum_{k\in S, i\in T}\varphi_{ki} \leq \sum_{k\in S, i\in T}c_{ki} = c(S, T),$$

deci, $v(\varphi) \leq c(S, T)$.

Relaţia (3.9) poate avea următoarea interpretate: *Valoarea unui flux într-o reţea de transport este egală cu suma fluxurilor de pe arcele de la S spre T minus suma fluxurilor de pe arcele de la T spre S, oricare ar fi tăietura* (S, T).

Teorema 3.2 *Dacă valoarea fluxului* φ *într-o reţea de transport este egală cu capacitatea unei tăieturi* (S, T) $(v(\varphi) = c(S, T))$, *atunci* ***fluxul*** φ ***are valoare maximă*** *iar* ***tăietura*** (S, T) ***are capacitate minimă***.

Demonstraţie: Dacă, prin absurd, fluxul φ nu are valoarea maximă, atunci ar exista un alt flux φ', de la s la t, de valoare strict mai mare: $v(\varphi') > v(\varphi)$. Cum $v(\varphi) = c(S, T)$ ar rezulta că fluxul φ' ar avea o valoare strict mai mare decât tăietura (S, T), în contradicţie cu teorema 3.1.

Analog, dacă tăietura (S, T) nu ar avea capacitate minimă, ar exista o tăietură (S', T') de capacitate strict mai mică $(c(S', T') < c(S, T)$. Cum $c(S, T) = v(\varphi)$ rezultă că fluxul φ are valoarea strict mai mare decât capacitatea tăieturii (S', T'), de asemenea, în contradicţie cu teorema 3.1.

Observaţie: Un flux care este propagat în reţeaua G între s şi t traversează toate tăieturile din reţea. Pentru o tăietură dată, arcele acesteia pot fi saturate sau nu. Din teoremele de mai sus rezultă că, ***dacă toate arcele unei tăieturi sunt saturate, atunci aceasta este o tăietură de capacitate minimă iar fluxul propagat este de valoare maximă***, egală cu capacitatea tăieturii.

3.3 Capacitate reziduală. Lanţ de augmentare

Fie $G = (X, E)$ un graf conex finit şi fără bucle cu nodurile s şi t fixe. Fiecare muchie $e \in E$ poate fi înlocuită cu două rute orientate, dar nu active în acelaşi timp. Fie $c(e)$, $e \in E$ capacitatea muchiilor grafului G şi considerăm că în graful G s-a propagat un flux φ. În raport cu fluxul φ, dacă $\varphi_{ij} = c_{ij}$, ruta orientată (i, j) este saturată. Vom conveni, ca toate rutele să fie orientate şi permise. În cazul unei rute blocate (nepermise), capacitatea va fi considerată nulă (şi implicit valoarea oricărui flux este nulă: $\varphi_{ij} = c_{ij} = 0$). Rutele blocate nu trebuie considerate ca fiind saturate.

Pentru început, o *reţea reziduală* diferă de reţeaua originară numai prin faptul că fiecare rută orientată ($i \to j$) face să lipsească ruta orientată în direcţia opusă ($j \to i$). În consecinţă, capacităţile arcelor în reţeaua reziduală (numite *capacităţi reziduale*) sunt definite astfel: *În fiecare moment o cantitate de flux φ este adăugată rutei ($i \to j$) în reţeaua originară, capacitatea reziduală a rutei ($i \to j$) descreşte cu mărimea φ iar capacitatea reziduală a rutei ($j \to i$) creşte cu mărimea φ. Astfel, capacitatea reziduală reprezintă capacitatea nefolosită a muchiei în reţeaua originară sau cantitatea de flux în direcţia opusă în această reţea care poate fi redusă sau o combinaţie a ambelor în reţeaua originară cu rute în ambele direcţii.*

Pentru fiecare arc (rută orientată permisă) $u = (i, j)$ definim *capacitatea reziduală* $\bar{c}_{ij}$ sau $\bar{c}(u)$, în raport cu fluxul φ, prin formula:

$$\bar{c}_{ij} = c_{ij} - \varphi_{ij} + \varphi_{ji} . \qquad (3.10)$$

Întrucât toate rutele sunt orientate în sensul intrare – ieşire ($s \to t$) arcele (i, j) au capacitate reziduală pozitivă: $\varphi_{ij} \le c_{ij}$, $\varphi_{ij} \ge 0$, de unde rezultă că $\bar{c}_{ij} \ge 0$.

Un *lanţ de augmentare* relativ la un flux φ este un lanţ orientat de la sursă la destinaţie în reţeaua reziduală astfel încât fiecare rută pe acest lanţ are capacitatea reziduală strict pozitivă. Fiecare rută componentă a unui lanţ de augmentare verifică una din următoarele proprietăţi:

- fie ruta (i, j) este permisă şi nesaturată ($\varphi_{ij} < c_{ij}$)
- fie ruta inversă (j, i) este permisă şi străbătută de un flux nenul $\varphi_{ji} > 0$.

În identificarea lanţurilor de augmentare de la s la t trebuie avut în vedere condiţia P_3 ca o rută neorientată să nu fie folosită decât într-un singur sens. Deci, odată cu identificarea unui lanţ se orientează şi rutele corespunzătoare.

Dacă λ este un lanţ de augmentare, $\lambda = [s, i_1, i_2, \dots, i_r, t]$, atunci capacitatea reziduală minimă pe rutele lanţului:

$$\theta = \min_{u \in \lambda} \bar{c}(u) \qquad (3.11)$$

este numită *capacitatea reziduală a lanţului de augmentare* şi reprezintă cantitatea de flux care poate fi propagată de-a lungul lanţului λ şi care se adaugă la valoarea fluxului total.

Teorema 3.3 *Dacă relativ la fluxul φ există un lanţ de augmentare de la s la t, atunci fluxul φ nu are valoare maximă.*

Demonstraţie: Fie θ minimul capacităţilor reziduale pe rutele unui lanţ λ orientat de la sursă la destinaţie. Conform formulei (3.11) avem:

$$\theta = \min_{k} \bar{c}_{i_k i_{k+1}} > 0 , \; k = 0, 1, \dots, r \qquad (3.12)$$

unde $i_0 = s$ şi $i_{r+1} = t$.

Construim un nou flux φ' în reţeaua G, după cum urmează:

1) pentru orice rută (i, j) care nu face parte din lanţul de augmentare λ, se menţine fluxul:

$$\varphi'_{ij} = \varphi_{ij}$$

2) dacă o muchie $[i_k, i_{k+1}] \in \lambda$ aparţine lanţului sunt posibile două situaţii:
   a) $\varphi_{i_k i_{k+1}} > 0$, atunci $\varphi_{i_k i_{k+1}} < c_{i_k i_{k+1}}$ şi $\varphi_{i_{k+1} i_k} = 0$. În această situaţie, fluxul pe ruta (i_k, i_{k+1}) creşte cu valoarea θ, iar fluxul pe ruta inversă rămâne nul:

$$\varphi'_{i_k i_{k+1}} = \varphi_{i_k i_{k+1}} + \theta \quad \text{şi} \quad \varphi'_{i_{k+1} i_k} = \varphi_{i_{k+1} i_k} = 0$$

   b) $\varphi_{i_{k+1} i_k} > 0$, atunci $\varphi_{i_k i_{k+1}} = 0$. În această situaţie apar două cazuri:
      i) $\varphi_{i_{k+1} i_k} \ge \theta$. În acest caz, diminuăm fluxul pe ruta (i_{k+1}, i_k) cu valoarea θ şi fluxul pe ruta inversă rămâne nul:

$$\varphi'_{i_{k+1} i_k} = \varphi_{i_{k+1} i_k} - \theta \quad \text{şi} \quad \varphi'_{i_k i_{k+1}} = \varphi_{i_k i_{k+1}} = 0$$

      ii) $\varphi_{i_{k+1} i_k} < \theta$. În acest caz micşorăm la zero fluxul pe ruta (i_{k+1}, i_k) şi creştem fluxul pe ruta inversă (i_k, i_{k+1}) cu $\theta - \varphi_{i_{k+1} i_k}$:

$$\varphi'_{i_{k+1}i_k} = 0 \quad \text{şi} \quad \varphi'_{i_k i_{k+1}} = 0 + (\theta - \varphi_{i_{k+1}i_k}) = \theta - \varphi_{i_{k+1}i_k}$$

În final, se observă că fluxul pe arcele (i_k, i_{k+1}) creşte iar pe arcele (i_{k+1}, i_k) scade. Din felul cum a fost construit noul flux φ'_{ij} rezultă că:

$$0 \le \varphi'_{i_{k+1}i_k} \le \varphi_{i_{k+1}i_k} \le c \quad \text{şi} \quad 0 \le \varphi_{i_k i_{k+1}} \le \varphi'_{i_k i_{k+1}} \le \varphi_{i_k i_{k+1}} + \theta - \varphi_{i_{k+1}i_k} \le c_{i_k i_{k+1}}$$

deci noile valori φ'_{ij} verifică proprietatea P_1).

De asemenea, pentru orice nod i avem două variante:

1. Nodul i nu se află pe lanţul λ. Fluxurile tuturor arcelor adiacente rămân aceleaşi şi deci se respectă proprietatea P_2).
2. Nodul $i = i_k \in \lambda$. În acest caz, la arcele adiacente spre interior nodul i_k s-a adăugat fluxul θ corespunzător arcelor (i_{k-1}, i_k) şi (i_{k+1}, i_k) iar la arcele adiacente spre exterior nodului i_k s-a adăugat, de asemenea, fluxul θ corespunzător arcelor (i_k, i_{k-1}) şi (i_k, i_{k+1}), deci se respectă, de asemenea, proprietatea P_2).

În concluzie, noul ansamblu de valori $\varphi' = \{\varphi'_{ij}\}$ este un flux în reţea şi $v(\varphi') = v(\varphi) + \theta > v(\varphi)$. Cum $\theta > 0$ avem $v(\varphi') > v(\varphi)$. Existenţa unui lanţ de augmentare permite creşterea valorii fluxului curent φ şi, implicit, existenţa sa arată că φ nu are valoare maximă. QED

3.4 Teorema Ford - Fulkerson[1)]. Algoritmul Ford – Fulkerson

3.4.1. Flux de valoare maximă şi tăietură de capacitate minimă

Teorema 3.4 *Într-o reţea de transport, dacă φ^* este un flux de valoare maximă atunci există o tăietură (S^*, T^*) cu capacitatea egală cu valoarea fluxului φ^*, adică*

$$v(\varphi^*) = c(S^*, T^*). \qquad (3.12)$$

Cu alte cuvinte, *într-o reţea de transport, valoarea maximă a fluxului este egală cu valoarea minimă a capacităţilor tăieturilor în reţea*.

Demonstraţie: Conform teoremei 3.3, valoarea oricărui flux într-o reţea este mai mică sau egală cu capacitatea oricărei tăieturi din reţea. De aici rezultă că, pentru a demonstra teorema este suficient să se arate că există un flux φ având valoarea egală cu capacitatea unei tăieturi (S, T). În aceste condiţii, fluxul φ trebuie să fie de valoare maximă. Dacă ar exista un flux de valoare mai mare, atunci valoarea lui ar depăşi capacitatea tăieturii (S, T). Capacitatea tăieturii (S, T) trebuie să fie minimală; dacă ar exista o tăietură de capacitate mai mică, atunci valoarea fluxului φ ar depăşi această capacitate.

Propagarea unui flux de valoare maximală se poate realiza printr-un procedeu cunoscut sub numele de *"procedeul marcării"* nodurilor reţelei. Anterior aplicării procedului de marcare, în reţea trebuie indus un flux iniţial φ. Prin procedeul de marcare, ce va fi prezentat în continuare, există posibilitatea de creştere a valorii fluxului din reţea, utilizând lanţurile de augmentare care leagă sursa s cu destinaţia t.

Procedură de marcare M **(etichetarea Ford – Fulkerson)**

a) Se marchează sursa (intrarea) s cu [+];
b) Dacă (i, j) este o rută permisă în direcţia $s \to t$ şi ***nesaturată*** ($\varphi_{ij} < c_{ij}$) cu i nod marcat şi j nemarcat, atunci se marchează nodul j cu $[+i]$;
c) Dacă (i, j) este o rută permisă în direcţia $t \to s$ cu ***flux nenul*** ($\varphi_{ij} > 0$), *indiferent de valoarea acestuia*, cu j nod marcat şi i nod nemarcat, atunci se marchează nodul i cu $[-j]$;
d) Dacă i este un nod marcat şi (i, j) este un arc saturat atunci nodul j nu se marchează.

Fie φ^* un flux de valoare maximă de la nodul de intrare s la nodul de ieşire t. Fie S^* mulţimea nodurilor marcate şi T^* mulţimea nodurilor nemarcate. Prin definiţie, $s \in S^*$. Să probăm că $t \notin S^*$ (este un nod nemarcat).

Dacă, prin absurd, nodul t ar fi marcat, atunci, plecând de la t şi folosind marcajul găsim un lanţ λ de la s la t, cu proprietatea că fiecare nod al lanţului este marcat dinspre nodul precedent. Deci λ este un lanţ de

[1)] L. R. Ford, D. R. Fulkerson, *Maximal flow through a network*, Canadian Journal of Mathematic, 8, 1956, p. 399

augmentare de-a lungul căruia valoarea fluxului φ^* poate creşte, contrazicând ipoteza privind maximalitatea lui. Deci t este un nod nemarcat, $t \in T^*$ şi (S^*, T^*) este o tăietură.

Dacă fluxul existent în reţea nu are valoare maximă, atunci de-a lungul lanţului de augmentare λ pus în evidenţă cu ajutorul procedeului de marcaj, valoarea sa se modifică în felul următor: pe fiecare arc din lanţ orientat spre ieşire, adăugăm o cantitate θ de flux, iar pe fiecare arc al lanţului orientat spre intrarea s, scădem cantitatea θ de flux. Valoarea lui θ va fi determinată astfel încât să se respecte condiţiile din definiţia fluxului. Valoarea noului flux va fi: $v(\varphi') = v(\varphi) + \theta$.

În cazul existenţei unui flux de valoare maximă φ^*, pentru a proba egalitatea $v(\varphi^*) = c(S^*, T^*)$ vom examina situaţia tuturor rutelor orientate (i, j) având o extremitate în S^* şi alta în T^*.

Având în vedere regula de marcare b) şi figura 3.3, putem spune că ***ruta (i, j) este saturată***. Mai mult,

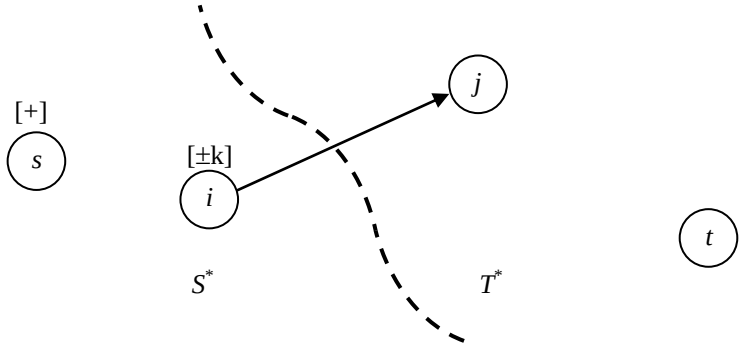


Figura **3**.3

toate arcele tăieturii (S^*, T^*) sunt *saturate şi ele traversează tăietura de la intrare spre ieşire* $(s \to t)$.

Dacă se presupune ruta (i, j) cu extremitatea iniţială $i \in T^*$ şi extremitatea finală $j \in S^*$, având în vedere regula de marcare c) şi figura 3.4, pe ruta (i, j) nu circulă flux ($\varphi_{ij} = 0$). Mai mult, pe orice arc orientat de la ieşire către intrare $(t \to s)$ nu circulă flux.

Prin urmare, în situaţia în care nodul de ieşire t nu poate fi marcat, fluxul total are valoare maximă.

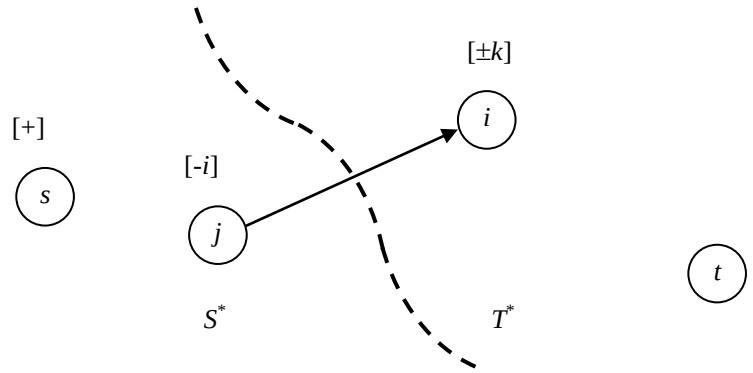


Figura 3.4

Considerând tăietura determinată de partiţia (S^*, T^*), faptul că nu s-a putut ajunge cu marcarea la nodul de ieşire t înseamnă că:

- toate arcele din mulţimea $\{(i, j) / i \in S^*, j \in T^*\}$ sunt saturate, adică $\varphi_{ij} = c_{ij}$;
- toate arcele din mulţimea $\{(i, j) / i \in T^*, j \in S^*\}$ au fluxul egal cu zero ($\varphi_{ij} = 0$).

În acest fel, toate unităţile de flux care formează fluxul de valoare maximă φ^* traversează tăietura (S^*, T^*) saturând toate arcele acesteia şi, odată ajunse în T^* nu se mai întorc înapoi. Prin urmare, $v(\varphi^*) = c(S^*, T^*)$.

Din cele prezentate anterior, aplicând procedeul de marcare putem ajunge la una dintre următoarele situaţii:

- ieşirea t a reţelei nu este marcată, în acest caz *fluxul din reţea are valoare maximă*;
- ieşirea t a reţelei este marcată, în acest caz *fluxul existent în reţea nu are valoare maximă*, valoarea sa urmând a fi majorată cu θ unităţi de flux de-a lungul unui lanţ de augmentare λ.

Procedeul de marcare se repetă până când ieşirea t a reţelei nu mai poate fi marcată. În acest moment fluxul obţinut are valoare maximă.

Teorema 3.5 *Odată cu valoarea maximă $v(\varphi^*)$ a fluxului în reţea se determină şi o tăietură (S^*, T^*) cu capacitate minimă.*

Demonstraţie: Conform observaţiilor de mai sus, când fluxul are valoare maximă avem $\varphi_{ij} = c_{ij}$ pe arcele (i, j) cu $i \in S^*$ şi $j \in T^*$ şi $\varphi_{ij} = 0$ pe arcele (i, j) cu $i \in T^*$ şi $j \in S^*$. Din relaţia 3.9 avem:

$$v(\varphi^*) = \sum_{i\in S^*,\, j\in T^*} \varphi_{ij} - \sum_{i\in T^*,\, j\in S^*} \varphi_{ij} = \sum_{i\in S^*,\, j\in T^*} \varphi_{ij} = \sum_{i\in S^*,\, j\in T^*} c_{ij} = c(S^*, T^*).$$

Prin urmare, dacă fluxul φ^* are valoarea maximă, atunci şi tăietura (S^*, T^*) are capacitatea minimă.

Dacă se suprimă arcele saturate, într-o reţea în care a fost propagat un flux de valoare maximă, se obţine un graf parţial care nu mai este conex. Acest lucru se justifică prin faptul că tăietura de capacitate minimă este formată numai din arce saturate şi acestea, odată eliminate, despart graful în două părţi disjuncte.

Observaţii:

1. Conceptul de "tăietură de valoare minimă" poate fi privit plastic ca "cel mai îngust loc al reţelei" în traversarea reţelei de către flux de la intrare spre ieşire;
2. Teorema 3.4 poate fi citită astfel: "cantitatea maximă de unităţi de flux care poate traversa toată reţeaua de la intrare la ieşire este egală cu lărgimea celui mai îngust loc al reţelei". Fluxul trebuie privit ca o bilă care traversează un tub cu secţiune variabilă.
3. Procedeul de marcare se poate folosi pentru verificarea maximalităţii unui flux, un flux are valoare maximă dacă nodul terminal t al reţelei nu poate fi marcat.
4. În reţea mai pot fi şi alte tăieturi de capacitate minimă în afară de (S^*,T^*) găsită prin procedeul de marcare; tăietura (S^*, T^*) este însă "prima" de la intrare spre ieşire.
5. Marcarea unui nod ne spune că, până la acel nod, mai există, încă, cel puţin un traseu pe care ar mai putea fi aduse unităţi de flux de la intrarea s iar marcajul ne va arăta şi care este acel traseu.

Prin marcarea nodului final se precizează pe ce traseu putem transporta mai multe unităţi de flux de la s la t, iar θ care este cantitatea maximă ce poate fi transportată pe acest traseu.

Ţinând seama de faptul că în orice reţea există întotdeauna un flux (de exemplu, fluxul nul), observaţiile de mai sus sugerează, de fapt, un algoritm prin care poate fi găsit un flux de valoare maximă într-o reţea. Acest algoritm va fi prezentat în cele ce urmează.

3.4.2. Algoritmul Ford – Fulkerson

Fie G o reţea de transport cu intrarea în nodul s şi ieşirea în nodul t. Determinarea unui flux de valoare maximă ce trece prin reţeaua de transport G de la s la t parcurge următoarele etape:

Etapa 1. *Construirea de drumuri de la s la t*

Iniţial, se consideră că în reţeaua G este propagat fluxul nul ($\varphi_{ij} = 0$ pentru orice arc (i, j)). Se aplică procedeul de marcare M constând doar din regulile a), b) şi d) şi se identifică drumuri în direcţia $s \to t$ între intrarea s şi ieşirea t. Prin aplicarea, în special, a regulei b) spunem că se face marcarea în sensul orientării rutei de la s la t ($s \to t$)

Fie μ un drum nesaturat de la s la t şi:

$$\theta = \min_{u\in\mu} \bar{c}(u) \quad , \quad \bar{c}(u) = c(u) - \varphi(u). \qquad (3.13)$$

Fluxul propagat de-a lungul drumului μ de valoare θ saturează cel puţin un arc (de fapt, toate acele arce care au $\bar{c}(u) = \theta$). Pe fiecare arc al reţelei va circula fluxul:

$$\varphi'(u) = \begin{cases} \varphi(u) + \theta & \text{dacă } u \in \mu \\ \varphi(u) & \text{dacă } u \notin \mu \end{cases}. \qquad (3.14)$$

Valoarea fluxului în reţeaua G va fi:

$$v(\varphi') = v(\varphi) + \theta. \qquad (3.15)$$

În identificarea acestor drumuri vom avea grijă ca *o muchie să nu fie folosită decât ca o rută într-un singur sens*. Odată cu definirea unui drum de la *s* la *t* se orientează şi muchiile reţelei. Pe fiecare arc se însumează fluxurile propagate succesiv.

Construirea unui flux continuă atâta timp cât este posibil să identificăm drumuri nesaturate de la *s* la *t*. În general, pentru o organizare eficientă a drumurilor folosite, la fiecare pas se poate considera drumul definit de rutele permise "*cele mai de sus*" (metoda drumului superior nesaturat). Aceasta permite să identificăm cât mai multe drumuri şi să obţinem în prima etapă un flux cu o valoare cât mai mare şi să mărim convergenţa algoritmului.

Fluxul propagat la finalul primei etape are valoarea egală cu suma fluxurilor propagate pe fiecare drum.

Etapa 2. *Identificarea lanţurilor de augmentare de la s la t*

Se aplică procedeul de marcare M şi, cu ajutorul etichetelor, se identifică două situaţii:

1. Ieşirea *t* ***nu este marcată***. În această situaţie, fluxul curent are valoarea maximă. Valoarea fluxului prin reţea este suma fluxurilor propagate pe toate drumurile de la etapa 1 şi pe toate lanţurile de augmentare de la etapa 2. Algoritmul continuă cu etapa 3.

2. Ieşirea *t* ***este marcată***. În această situaţie, s-a reuşit marcarea unui lanţ de augmentare λ care uneşte intrarea *s* cu ieşirea *t*. Se calculează valoarea:

$$\theta = \min\{\min_{u\in B}[c(u)-\varphi(u)], \min_{u\in C}\varphi(u)\} \qquad (3.16)$$

unde B este mulţimea arcelor din lanţul λ pentru care s-a efectuat marcarea după regula b) iar C este mulţimea arcelor din lanţul λ pentru care s-a efectuat marcarea după regula c). Ţinând cont de regulile de marcare este clar că $\theta > 0$ şi noul flux φ' propagat de-a lungul lanţului λ măreşte valoarea fluxului φ cu valoarea θ iar noile fluxuri pe arcele reţelei se calculează cu relaţia:

$$\varphi'(u) = \begin{cases} \varphi(u)+\theta & dacă\ u\in B \\ \varphi(u)-\theta & dacă\ u\in C \\ \varphi(u) & \text{în rest} \end{cases} \qquad (3.17)$$

şi

$$v(\varphi') = v(\varphi) + \theta \qquad (3.18)$$

Se reia etapa 2 în cadrul unei noi iteraţii.

Etapa 3. *Determinarea unei tăieturi de capacitate minimă*

În ultima marcare, efectuată în etapa 2, fie T^* mulţimea nodurilor nemarcate, conform aplicării regulilor a), b) şi c) de marcare în vederea identificării unui lanţ de augmentare de la *s* la *t*, şi S^* mulţimea nodurilor marcate. Evident $s \in S^*$, $t \in T^*$ şi $S^* \cup T^* = X$, unde *X* este mulţimea nodurilor reţelei *G*.

Tăietura (S^*, T^*) de capacitate minimă este:

$$(S^*, T^*) = \{(i, j) / i \in S^*, j \in T^*\} \qquad (3.19)$$

cu capacitatea

$$c(S^*, T^*) = \sum_{u\in(S^*,T^*)} c(u) \qquad (3.20)$$

În această etapă, se verifică (teorema Ford-Fulkerson) dacă, într-adevăr, fluxul are valoare maximă. Valoarea maximă a fluxului este egală cu valoarea minimă a capacităţilor tăieturilor în reţea:

$$v(\varphi^*) = \max_{\varphi} v(\varphi) = \min_{(S,T)} c(S,T) = c(S^*, T^*).$$

Dacă valoarea fluxulului determinată în etapa 2 nu este egală cu capacitatea tăieturii determinată în etapa 3, atunci algoritmul se reia cu etapa 2 şi se identifică noi lanţuri de augmentare.

Aplicaţie: Flux de valoare maximă într-o reţea de transport

Se consideră reţeaua de transport $G = (X, E)$ din figura 3.5 şi se cere construirea unui flux de valoare maximă de la s la t. Capacităţile arcelor şi muchiilor sunt înscrise în paranteze. Pentru muchiile neorientate, capacitățile sunt valabile în ambele sensuri de parcurgere.

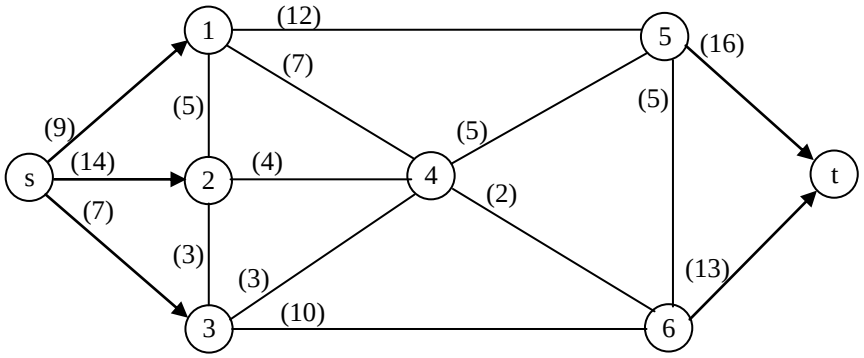


Figura 3.5

Etapa 1. *Construirea de drumuri de la s la t*

Se pun în evidenţă toate drumurile complete μ_i de la s la t şi, pe fiecare dintre acestea, se transportă o cantitate de flux θ_i propagată conform formulelor (3.3). Iniţial, fluxul în reţea este nul.

Nr. crt.	Drum		Unităţi de flux transportate	Rute saturate
1	$\mu_1 = (s, 1, 5, t)$	$\rightarrow$	$\theta_1 = \min(9, 12, 16) = 9$	(s, 1)
2	$\mu_2 = (s, 2, 1, 4, 5, t)$	$\rightarrow$	$\theta_2 = \min(14, 5, 7, 5, 7) = 5$	(4, 5), (2, 1)
3	$\mu_3 = (s, 2, 4, 6, 5, t)$	$\rightarrow$	$\theta_3 = \min(9, 4, 2, 5, 2) = 2$	(4, 6), (5, t)
4	$\mu_4 = (s, 2, 3, 6, t)$	$\rightarrow$	$\theta_4 = \min(7, 3, 10, 13) = 3$	(2, 3)
5	$\mu_5 = (s, 3, 6, t)$	$\rightarrow$	$\theta_5 = \min(7, 7, 10) = 7$	(s, 3), (3, 6)

Deoarece reţeaua de transport conţine şi muchii neorientate, în identificarea drumurilor între s şi t vom avea grijă ca o rută neorientată să nu fie folosită decât într-un singur sens. Astfel, odată cu identificarea unui drum, se orientează şi rutele în sensul dat de prima unitate de flux transportată pe drumul respectiv.

Pentru construirea fluxului din figura 3.6 pe primul drum μ_1 s-au propagat 9 unităţi de flux, pe al doilea drum μ_2 s-au propagat 5 unităţi de flux, pe drumul μ_3 s-au propagat 2 unităţi de flux, pe drumul μ_4 s-au propagat 3 unităţi de flux şi pe drumul μ_5 s-au propagat 7 unităţi de flux. Fluxul propagat în reţea are valoarea $v(\varphi) = \theta_1 + \theta_2 + \theta_3 + \theta_4 + \theta_5 = 26$ unităţi.

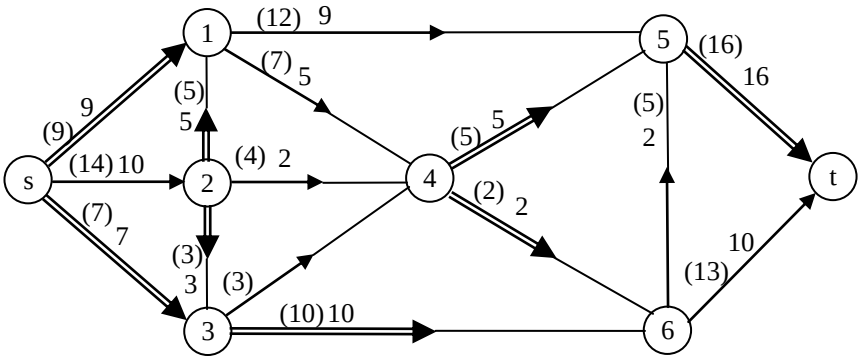


Figura 3.6

În figura 3.6 fluxurile propagate pe fiecare arc sunt cumulate şi înscrise pe arce după capacităţi. Arcele saturate sunt reprezentate prin linie dublă orientată în sensul de parcurgere a muchiei de către unităţile de flux.

Etapa 2. *Identificarea lanţurilor de augmentare de la s la t*

Prin aplicarea procedeului de marcaj se pune în evidenţă un lanţ de augmentare, conform marcajului din figura 3.7:

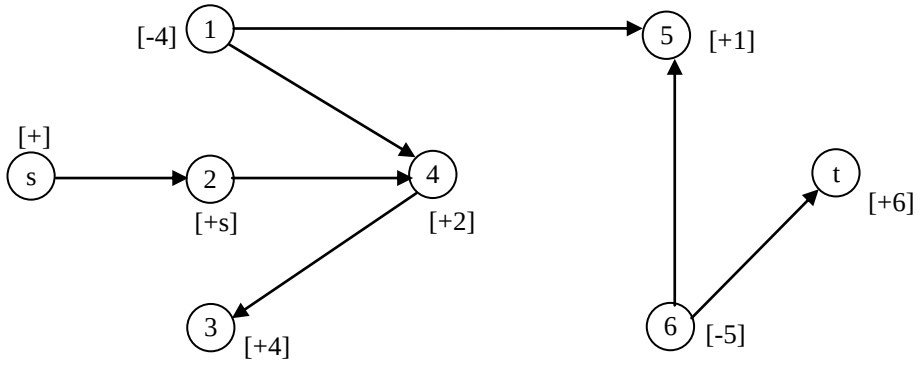


Figura 3.7

Avem astfel lanţul de augmentare:

$$\lambda : s \to 2 \to 4 \leftarrow 1 \to 5 \leftarrow 6 \to t$$

care uneşte intrarea *s* cu ieşirea *t*. Se observă că

$B = \{(s, 2),\ (2, 4), (1, 5), (6, t)\}$, $C = \{(1, 4),\ (6, 5)\}$

sunt mulţimile arcelor din lanţ marcate cu regula b), respectiv cu regula c). De-a lungul acestui lanţ se propagă un flux de valoare:

$$\theta = min\{min\{c_{s2} - \varphi_{s2}, c_{24} - \varphi_{24}, c_{15} - \varphi_{15}, c_{6t} - \varphi_{6t}\}, min\{\varphi_{14}, \varphi_{65}\}\} =$$
$$= min\{min\{4, 2, 3, 3\}, min\{5, 2\}\} = min\{2, 2\} = 2$$

Fluxurile pe arcele (*s*, 2), (2, 4), (1, 5) şi (6, *t*) se măresc cu 2 unităţi iar fluxurile pe arcele (1, 4) şi (6, 5) se micşorează cu 2 unităţi. Arcul (2, 4) devine saturat. În acest fel, în rețeaua de transport s-a obținut un nou flux, φ', de valoare: $v(\varphi') = v(\varphi) + \theta = 26 + 2 = 28$ unităţi. Fluxul φ' este reprezentat în Figura 3.8.

Aplicarea procedeului de marcaj în raport cu noul flux, φ', pune în evidenţă următoarea situaţie:

Nodul marcat şi marcajul	s [+]	2 [+s]

Nodul *t* nu poate fi marcat deci fluxul construit în figura 3.8 are valoare maximă, în rețea neexistând lanţuri de augmentare.

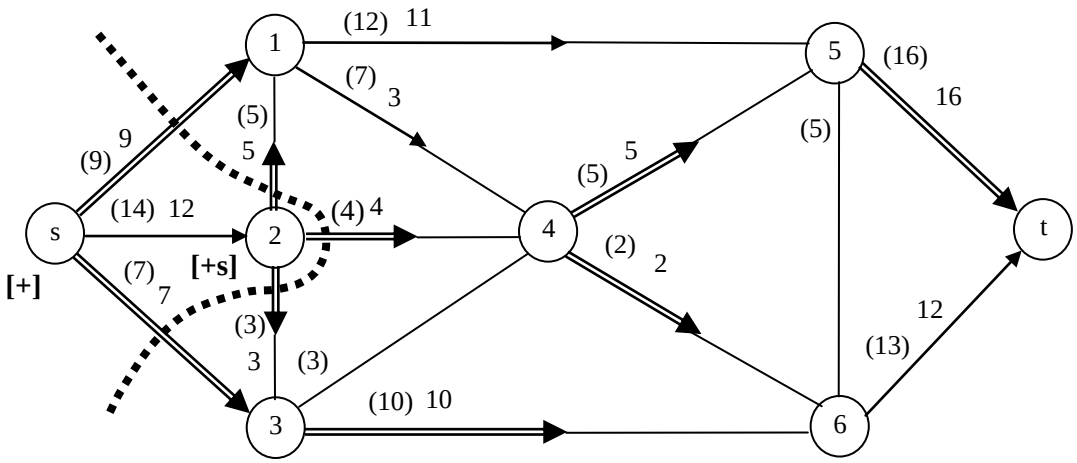


Figura 3.8

Etapa 3. *Determinarea unei tăieturi de capacitate minimă*

Mulţimea nodurilor reţelei de transport se poate scrie ca o reuniune a două submulţimi disjuncte de noduri *S* şi *T*, în submulţimea *S* fiind incluse nodurile marcate iar în *T* cele nemarcate:

$$X = S \cup T,\ S \cap T = \phi,\ S = \{s, 2\},\ T = \{1, 3, 4, 5, 6, t\}.$$

Existenţa fluxului de valoare maximă în reţea şi identificarea submulţimilor *S* şi *T* permite evidenţierea tăieturii de capacitate minimă. Tăietura este reprezentată în figura 3.8 prin linia punctată.

Capacitatea tăieturii este dată de suma capacităţilor arcelor tăieturii (arce saturate care intersectează tăietura orientate de la *s* la *t*):

$$c(S, T) = c_{(s, 1)} + c_{(2, 1)} + c_{(2, 4)} + c_{(2, 3)} + c_{(s, 3)} = 9 + 5 + 4 + 3 + 7 = 28.$$

Deci, conform teoremei Ford-Fulkerson fluxul maxim în reţea are valoarea de 28 de unităţi. Analizând reţeaua din figura 3.8 se observă că din nodul *s* mai poate fi trimisă în reţea încă o unitate de flux pe ruta (s, 2) care poate ajunge în nodul *t* în situaţia în care capacitatea arcului (2, 1), de exemplu, ar fi suplimentată.

3.5. Problema de cerere şi ofertă în reţea de transport cu capacităţi limitate

Fie *G* o reţea în care rutele sunt neorientate, orientate sau parţial orientate cu *S* mulţimea de surse şi *T* mulţimea de destinaţii. O anumită marfă se află disponibilă în locurile sursă $i \in S$ şi este solicitată în locurile destinaţie $j \in T$. Presupunem că a_i este oferta sursei *i* şi b_j este cererea destinaţiei *j*. Dacă *Q* este cantitatea totală de marfă, conform regulei în care cererea totală este egală cu oferta totală, avem:

$$Q = \sum_i a_i = \sum_j b_j$$

Între surse şi destinaţii există o reţea de legături (directe sau indirecte) prin intermediul unor puncte intermediare (de tranzit). Pe unele rute se poate circula în ambele sensuri, pe altele numai într-un sens şi pot exista legături între surse, respectiv între destinaţii. În acelaşi timp, pe fiecare rută pot exista limitări superioare privind cantităţile ce sunt transportate. În aceste condiţii, se pune problema dacă reţeaua permite satisfacerea integrală a cererilor în nodurile de destinaţie.

Această problemă se reduce la o problemă de flux de valoare maximă, în sensul definit anterior, în felul următor:

a) Se completează reţeaua cu un nod suplimentar *s* şi cu rutele orientate (*s*,*i*), $i \in S$, iar capacităţile acestor rute sunt egale cu ofertele a_i ($c_{si}=a_i$);
b) Se completează reţeaua cu un nod suplimentar *t* şi cu rutele *orientate* (*j*, *t*), $j \in T$, iar capacităţile acestor rute vor fi egale cu cererile b_j ($c_{jt} = b_j$).

Se observă uşor că un flux, în noua reţea formată, numită ***reţea standard***, se poate identifica cu deplasarea unei cantităţi de marfă din locurile în care se află disponibilă spre locurile în care este cerută, valoarea fluxului fiind egală cu cantitatea totală de marfă deplasată. Valoarea oricărui flux nu poate depăşi valoarea *Q*.

Din punct de vedere economic şi organizatoric, tăietura (S^*, T^*) de capacitate minimă creează o "ştrangulare" a reţelei. Fluxul curent de valoare maximă propagat în reţea nu satisface, de multe ori, cererea în punctele de destinaţie. Pentru deblocarea situaţiei, în sensul îndeplinirii cererilor, este necesar să se mărească capacitatea (sau capacităţile) *unor rute incluse în tăietura de capacitate minimă*.

Tăietura de capacitate minimă reprezintă o graniţă între o parte a reţelei, unde poate să intre un flux, şi o altă parte, unde poate să treacă un alt flux. Avem chiar următorul rezultat general datorat lui Gale (1957): *"Dacă G este o reţea neorientată cu S mulţimea de surse şi T mulţimea de destinaţii iar muchiile au capacităţi nenegative, atunci condiţia necesară şi suficientă pentru ca cererile* b_j, $j \in T$, *să fie satisfăcute din ofertele* a_i, $i \in S$, *este ca, pentru orice tăietură (U, V) în G (*$s \in U$, $t \in V$*), cererea netă în V să nu depăşească capacitatea tăieturii"*, adică:

$$\sum_{j \in T \cap V} b_j - \sum_{i \in S \cap V} a_i \le c(U, V)$$

Aplicaţie: Problema de cerere şi ofertă

Un produs este disponibil în punctele 1, 2, 3 în cantităţile $a_1 = 16$, $a_2 = 16$, $a_3 = 18$ şi solicitat în punctele 7 şi 8 în cantităţile $b_7 = 20$ şi $b_8 = 30$. Între surse şi destinaţii există rute de legătură conform reţelei din figura 3.9. Toate rutele sunt neorientate. Capacităţile rutelor sunt notate între paranteze şi sunt valabile, acolo unde este cazul, în ambele sensuri de propagare. Să se stabilească în ce măsură cererile pot fi satisfăcute.

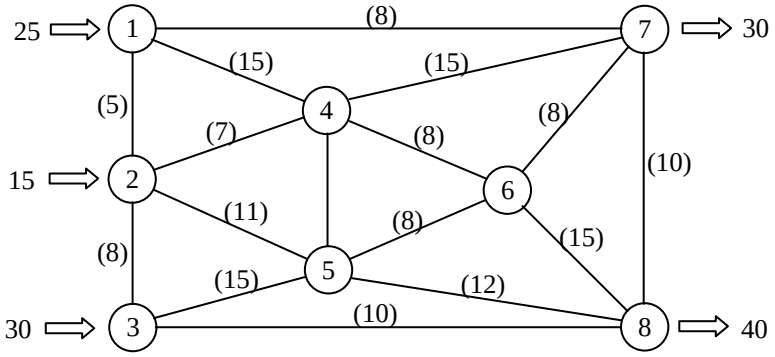


Figura 3.9

Rezolvare:

Se completează reţeaua cu nodurile *s* şi *t* şi cu rutele orientate (*s*, *1*), (*s*, *2*), (*s*, *3*) de capacităţi 25, 15 şi 30 şi rutele orientate (*7*, *t*), (*8*, *t*), de capacităţi 30 şi 40. Se propagă în reţea un flux de la *s* la *t*. Dacă valoarea acestui flux este egală cu 70 (= 30 + 40) unităţi de flux, atunci cererile sunt acoperite în întregime.

Utilizând varianta uzuală a algoritmului Ford-Fulkerson, în prima etapă se propagă un flux nenul de valoare cât mai mare posibilă şi compatibil. Pentru aceasta vom identifica drumuri de la *s* la *t* şi, corespunzător, orientăm arcele:

$\mu_1 = (s, 1, 7, t) \rightarrow \theta_1 = \min (25, 8, 30) = 8$
$\mu_2 = (s, 1, 4, 7, t) \rightarrow \theta_2 = \min (17, 15, 15, 22) = 15$
$\mu_3 = (s, 1, 2, 4, 6, 7, t) \rightarrow \theta_3 = \min (2, 5, 7, 8, 8, 7) = 2$
$\mu_4 = (s, 2, 4, 6, 7, t) \rightarrow \theta_4 = \min (15, 5, 6, 6, 5) = 5$
$\mu_5 = (s, 2, 5, 4, 6, 8, t) \rightarrow \theta_5 = \min (10, 9, 5, 1, 15, 40) = 1$
$\mu_6 = (s, 2, 5, 6, 8, t) \rightarrow \theta_6 = \min (9, 10, 8, 14, 39) = 8$
$\mu_7 = (s, 2, 5, 8, t) \rightarrow \theta_7 = \min (1, 2, 12, 31) = 1$
$\mu_8 = (s, 3, 2, 5, 8, t) \rightarrow \theta_8 = \min (30, 8, 1, 11, 30) = 1$
$\mu_9 = (s, 3, 5, 8, t) \rightarrow \theta_9 = \min (29, 15, 10, 29) = 10$
$\mu_{10} = (s, 3, 8, t) \rightarrow \theta_{10} = \min (19, 10, 19) = 10$

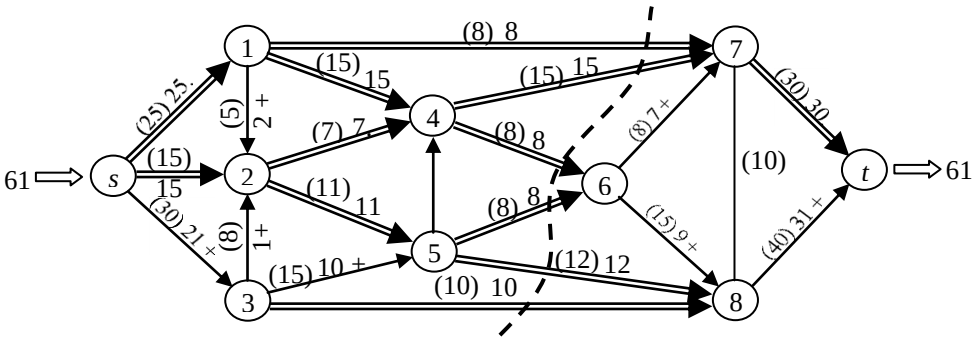


Figura 3.10

Orice drum care uneşte intrarea *s* cu ieşirea *t* conţine cel puţin o rută saturată. În acest fel, s-a propagat în reţea un flux complet de-a lungul drumurilor identificate în reţeaua din figura 3.10. Fluxul complet are valoarea:

$$v(\varphi) = \sum_{i=1}^{10} \theta_i = 61 \text{ unităţi.}$$

În etapa următoare, pentru determinarea fluxului de valoare maximă, vom aplica procedeul de marcare, În reţeaua din figura 3.10 se pot efectua următoarele marcaje (figura 3.11):

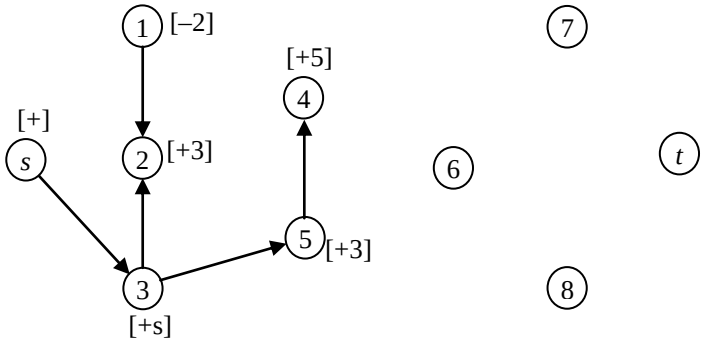


Figura 3.11

Fluxul curent, a cărui valoare este de 61 unităţi, este fluxul de valoare maximă pentru că nu poate fi marcat nodul final *t*. Valoarea sa fiind mai mică decât 70 înseamnă că nu toate cererile sunt acoperite integral; destinaţia 8 primeşte numai 31 unităţi din cele 40 solicitate. Satisfacerea cererii restante se poate face numai prin mărirea capacităţilor unor rute de legătură. Aceste rute le identificăm *în tăietura de capacitate minimă*.

În etapa finală se pune în evidenţă tăietura de capacitate minimă, separând nodurile marcate de cele nemarcate. Având în vedere că $S^* = \{s, 1, 2, 3, 4, 5\}$ şi $T^* = \{6, 7, 8, t\}$ tăietura cuprinde arcele:

$$(S^*, T^*) = \{(1, 7) , (4, 7) , (4, 6) , (5, 6) , (5, 8) , (3, 8)\}$$

Suma capacităţilor celor şase rute din tăietură exact 61, ceea ce confirmă faptul că fluxul curent este de valoare maximă.

Orice cantitate suplimentară de flux trebuie să treacă prin una sau mai multe rute din cele şase ale tăieturii minime. Să presupunem că cele 9 unităţi de flux necesare pentru a satisface cererea destinaţiei 8 le repartizăm pe muchii, crescând capacitatea muchiei [1, 7] cu 2 unităţi, capacitatea muchiei [5, 6] cu 6 unităţi şi capacitatea muchiei [4, 7] cu 1 unitate.

Algoritmul Ford – Fulkerson se reia prin marcarea nodurilor. Se identifică succesiv lanţurile μ_{11}, μ_{12} şi μ_{13} conform figurilor 3.12, 3.13 şi 3.14.

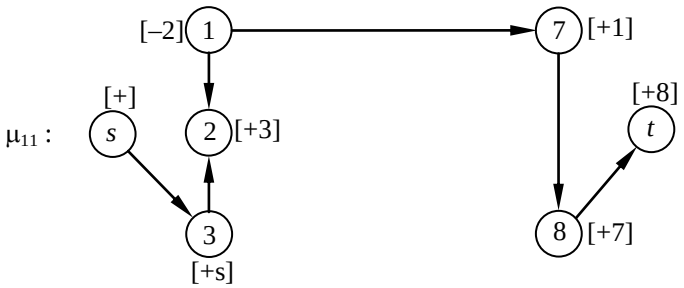


Figura 3.12

Avem:

$$\bar{c}_{s3} = 30 - 21 + 0 = 9, \quad \bar{c}_{32} = 8 - 1 + 0 = 7, \quad \bar{c}_{21} = 5 - 0 + 2 = 7,$$

$$\bar{c}_{17} = 10 - 8 + 0 = 2, \quad \bar{c}_{78} = 10, \quad \bar{c}_{8t} = 40 - 31 + 0 = 9$$

de unde:

$$\theta_{11} = \min (9, 7, 7, 2, 10, 9) = 2.$$

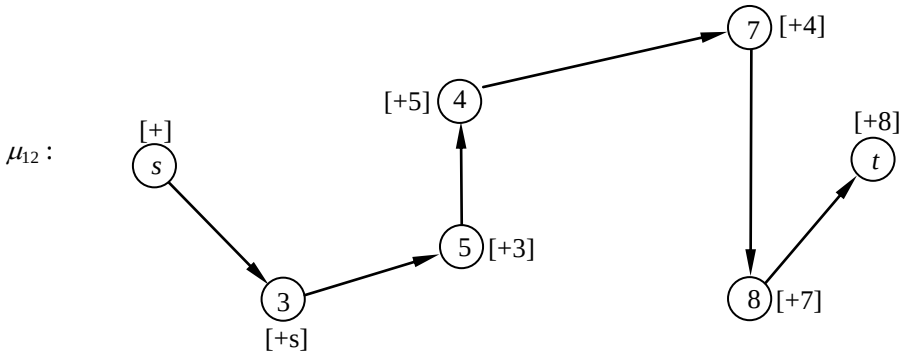


Figura 3.13

Avem:

$$\bar{c}_{s3} = 30 - 23 + 0 = 7, \quad \bar{c}_{35} = 15 - 10 + 0 = 5, \quad \bar{c}_{54} = 5 - 1 + 0 = 4,$$

$$\bar{c}_{47} = 8 - 7 + 0 = 1, \quad \bar{c}_{78} = 10 - 2 + 0 = 8, \quad \bar{c}_{8t} = 40 - 33 + 0 = 7$$

de unde:

$$\theta_{12} = \min (7, 5, 4, 1, 8, 7) = 1$$

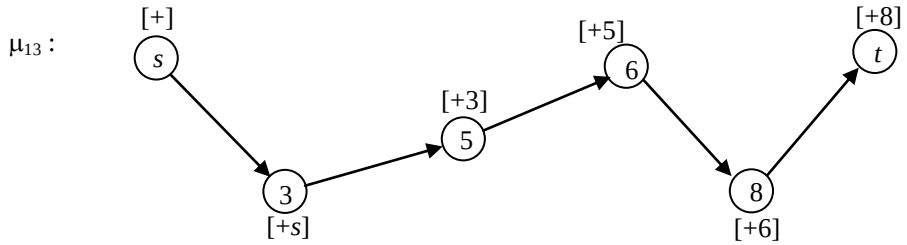


Figura 3.14

Avem:

$$\bar{c}_{s3} = 30 - 24 + 0 = 6, \quad \bar{c}_{35} = 15 - 11 + 0 = 4, \quad \bar{c}_{56} = 14 - 8 + 0 = 6,$$

$$\bar{c}_{68} = 15 - 9 + 0 = 6, \quad \bar{c}_{8t} = 40 - 34 + 0 = 6$$

de unde:

$$\theta_{13} = \min (6, 4, 6, 5, 6) = 4.$$

Transportând în rețea cele $\theta_{11} + \theta_{12} + \theta_{13} = 2 + 1 + 4 = 7$ unități de produs se obține situația din figura 3. 15. Valoarea noului flux propagat în rețea este:

$$v(\varphi) = 61 + \theta_{11} + \theta_{12} + \theta_{13} = 68 \leq 70.$$

Noua tăietură de capacitate minimă are în componență arcele:

$$(S^*, T^*) = \{(1, 7) , (1, 4) , (2, 4) , (2, 5) , (3, 5) , (3, 8)\}$$

cu valoarea $c(S^*, T^*) = 68$.

Repetarea marcării duce la situaţia în care nodul final *t* nu mai poate fi marcat:

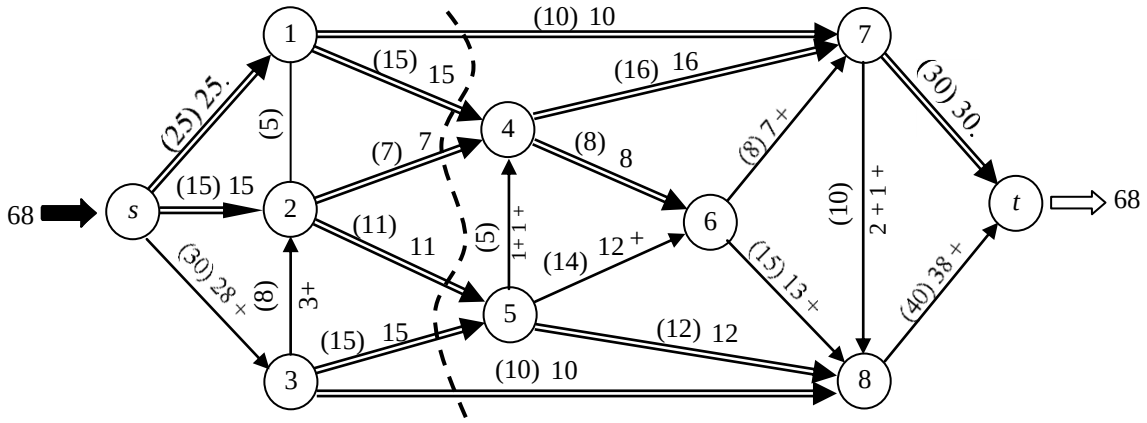


Figura 3.15

Pentru ca cererile să fie satisfăcute, se măreşte capacitatea muchiei [3, 5] cu 2 unităţi. Reluând algoritmul se identifică lanţurile de augmentare μ_{14} şi μ_{15} conform figurilor 3.16 şi 3.17:

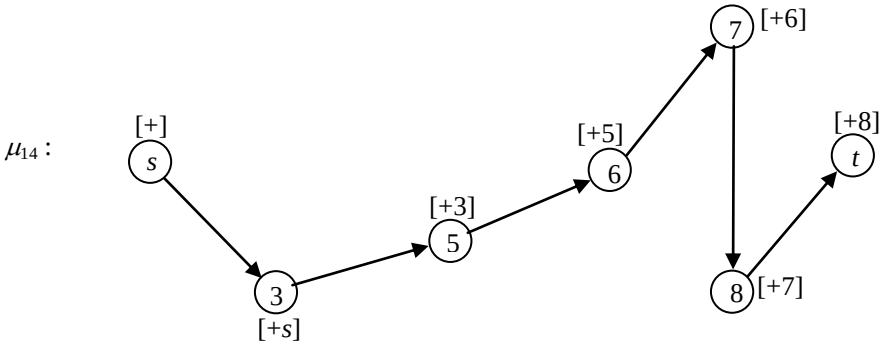


Figura 3.16

Avem:

$$\bar{c}_{s3} = 30 - 28 + 0 = 2,\ \ \bar{c}_{35} = 17 - 15 + 0 = 2,\ \ \bar{c}_{56} = 14 - 12 + 0 = 2,$$

$$\bar{c}_{67} = 8 - 7 + 0 = 1,\ \bar{c}_{78} = 10 - 3 + 0 = 7,\ \ \bar{c}_{8t} = 40 - 38 + 0 = 2,$$

de unde:

$$\theta_{14} = \min (2, 2, 2, 1, 7, 2) = 1$$

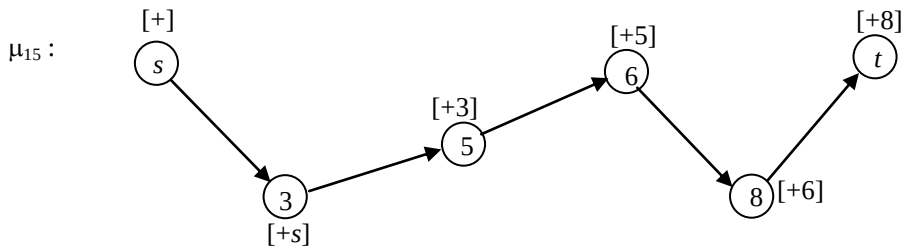


Figura 3.17

Avem:

$$\bar{c}_{s3} = 30 - 29 + 0 = 1,\ \ \bar{c}_{35} = 17 - 16 + 0 = 1,\ \ \bar{c}_{56} = 14 - 13 + 0 = 1,$$

$$\bar{c}_{68} = 15 - 13 + 0 = 2,\ \bar{c}_{8t} = 40 - 39 + 0 = 1$$

de unde: $\theta_{15} = \min (1, 1, 1, 2, 1) = 1.$

În acest moment valoarea fluxului propagat este $v(\varphi) = 68 + \theta_{14} + \theta_{15} = 70$. Cererea este acoperită integral, iar în figura 3.18 sunt date valorile fluxurilor pe fiecare arc.

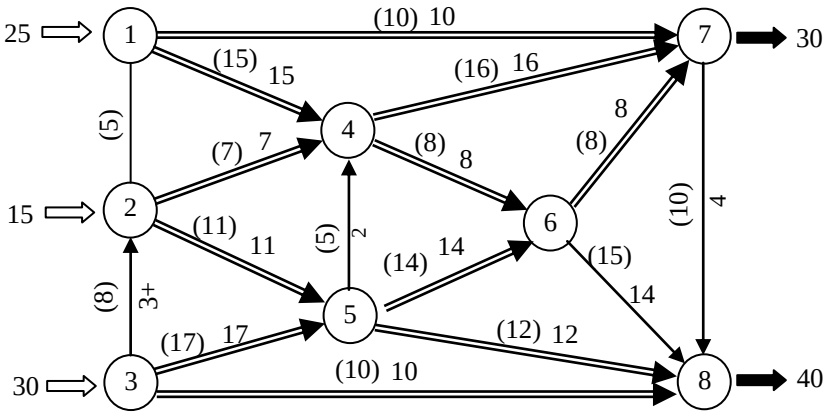


Figura 3.18

3.6. Flux de valoare maximă şi cost minim

Fie o reţea $G = (X, E)$ cu două noduri distincte s şi t, unde s este nodul de intrare iar t este nodul de ieşire. Muchiile pot fi orientate sau nu. Notăm cu U mulţimea rutelor orientate sau a arcelor permise. Pentru fiecare $u \in U$ presupunem două valori numerice:

$c(u)$ = capacitatea arcului u, $c(u) > 0$;
$a(u)$ = costul unitar de transport pe ruta u, $a(u) \geq 0$.

În cazul în care $u = (i, j)$ atunci putem scrie $c(u) = c_{ij}$ şi $a(u) = a_{ij}$. Pentru simplitatea expunerii extindem mulţimea U la toate rutele orientate, generate de graful G, punând $c(u) = 0$ şi $a(u) = +\infty$ în cazul în care ruta u este în realitate blocată.

Pentru un flux $\varphi = (\varphi(u))$, propagat de la s la t, definim costul său prin formula:

$$a(\varphi) = \sum_{u \in U} \varphi(u) \cdot a(u) .$$

Fie $Q > 0$. Se cere să se determine în reţeaua G un flux φ^* de la s la t, a cărui valoare să fie Q şi al cărui cost să fie minim:

$$v(\varphi^*) = Q \ , \ a(\varphi^*) = \min.$$

În legătură cu această cerinţă se ridică problema compatibilităţii, adică trebuie să verificăm dacă există în reţeaua G fluxuri de la s la t de valoare Q. Problema se rezolvă astfel:

- Se adaugă în reţeaua G un nod s_0 şi un arc (s_0, s) de capacitate Q (sau se adaugă un nod t_0 şi un arc (t, t_0) de valoare Q.
- În reţeaua extinsă, valoarea oricărui flux de la s_0 la t (sau de la s la t_0) este mai mică sau cel mult egală cu Q.

În concluzie:

- Dacă în reţeaua extinsă există un flux φ de valoare Q, atunci şi în reţeaua originară există un flux de valoare Q;
- Dacă fluxul maxim în reţeaua extinsă are o valoare inferioară lui Q, atunci în reţeaua originară nu există fluxuri de valoare Q.

Să considerăm în G un flux oarecare φ de la s la t. Atunci, în raport cu fluxul φ, se modifică costurile rutelor orientate, astfel:

$$\overline{a}_{ij} = \begin{cases} a_{ij} & \text{dacă } 0 \leq \varphi_{ij} \leq c_{ij} \text{ şi } \varphi_{ij} = 0 \\ +\infty & \text{dacă } \varphi_{ij} = c_{ij} \\ -a_{ij} & \text{dacă } \varphi_{ij} > 0 \end{cases} \qquad (3.24)$$

***Teorema 3.7**: Fluxul φ are costul minim printre toate fluxurile de aceeaşi valoare cu a sa, dacă şi numai dacă în reţeaua cu costurile modificate nu există circuite cu cost total negativ.*

Pe baza acestei teoreme, prezentăm doi algoritmi de rezolvare a problemelor de flux de valoare maximă şi cost minim.

Algoritmul 1

Etapa de iniţializare: În graful G se consideră un flux de valoare nulă.

Etapa iterativă: Fie $v(\varphi)$ valoarea fluxului curent.
Pasul 1. În raport cu φ, modificăm costurile rutelor, cu relaţiile (3.24).
Pasul 2. Se determină drumul de cost minim de la s la t, folosind costurile modificate:
- Dacă un asemenea drum nu există, atunci algoritmul se opreşte.
- În caz contrar, se propagă pe acest drum un flux de la s la t egal cu minimul capacităţilor reziduale ale rutelor ce compun drumul (saturăm drumul) şi se reia etapa iterativă în cadrul unei noi iteraţii

În general, algoritmul 1 se recomandă să fie folosit în reţele de mici dimensiuni. Pentru că unele costuri modificate pot fi negative, în rezolvarea problemelor de cost minim din etapa iterativă (pasul 2) se poate aplica algoritmul Dijkstra. Se poate aplica, de asemenea, orice procedură de marcaj temporar (algoritmul Ford, de exemplu).

Aplicaţie*: *Flux de valoare maximă şi cost minim

Să considerăm reţeaua de transport din figura 3.19:

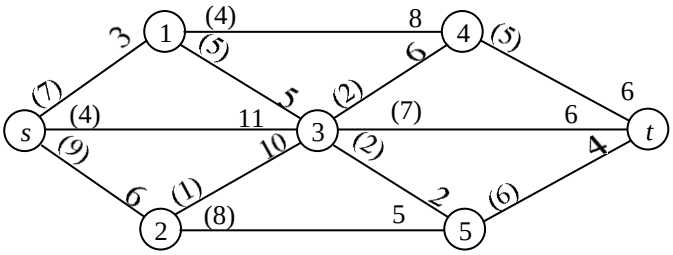


Figura 3.19

Pe fiecare muchie sunt ataşate atât capacităţile (valorile din paranteze), care induc o primă restricţie şi se consideră ca valori maxime, cât şi costurile unitare de transport care reprezintă unităţile monetare care se plătesc pentru transportul unei unităţi de produs între două noduri ale reţelei.
Să se determine fluxul maxim de cost minim de la s la t.

Se consideră că fluxul propagat în graf este fluxul nul de la s la t ($v(\varphi_0) = 0$).

Iteraţia 1

Pasul 1. În raport cu acest flux modificăm costurile ataşate muchiilor (figura 3.20). La fiecare nod i se ataşează valoarea drumului minim de la s la i astfel: p = valoare.

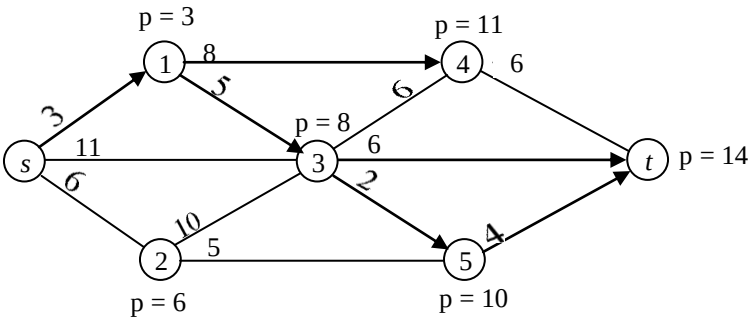


Figura 3.20

Pasul 2. Vom determina drumurile de cost minim de la s la t în reţeaua din figura 3.20. Drumurile de cost minim sunt:

$$s \xrightarrow{7} 1 \xrightarrow{4} 4 \xrightarrow{5} t \qquad \theta_1 = 4$$

$s \xrightarrow{3} 1 \xrightarrow{5} 3 \xrightarrow{7} t$ $\qquad \theta_2 = 3$

$s \xrightarrow{0} 1 \to 3 \to 5 \to t$ $\qquad \theta_3 = 0$

Valoarea fluxului este $v(\varphi_1) = 7$ unităţi de flux (figura 3.21).

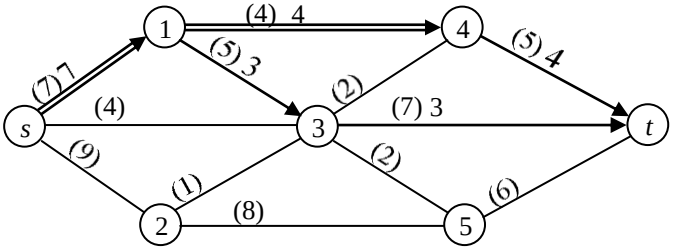


Figura 3.21

Costul este $a(\varphi_1) = \sum a_{ij} \cdot \varphi_{ij} = v(\varphi_1) \cdot p(t) = 98$ unităţi monetare.

Iteraţia 2

Pasul 1. Se modifică costurile în graf în raport cu fluxul φ_1 şi obţinem situaţia din figura 3.22:

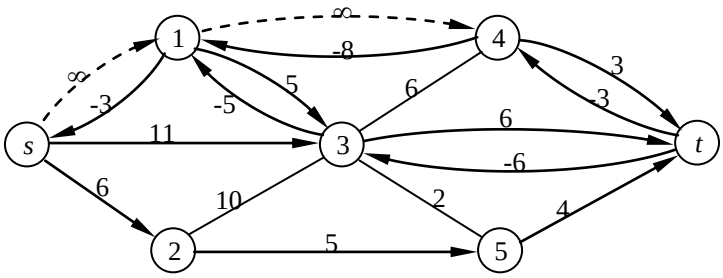


Figura 3.22

Pasul 2. Se determină drumurile de valoare minimă de la s la t.

$s \xrightarrow{9} 2 \xrightarrow{8} 5 \xrightarrow{6} t$ $\qquad \theta_4 = 6$

Se propagă un fluxul φ_2, adăugând la φ_1 6 unităţi de flux pe drumul $s \to 2 \to 5 \to t$ (figura 3.23). Valoarea fluxului este acum $v(\varphi_2) = v(\varphi_1) + 6 = 13$ unităţi de flux.

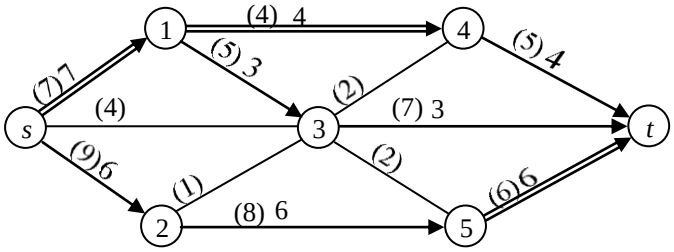


Figura 3.23

Costul va fi $a(\varphi_2) = a(\varphi_1) + 6 \cdot 15 = 188$ unităţi monetare.

Iteraţia 3

Pasul 1. Se modifică costurile în graf în raport cu fluxul φ_2 şi obţinem situaţia din figura 3.24:

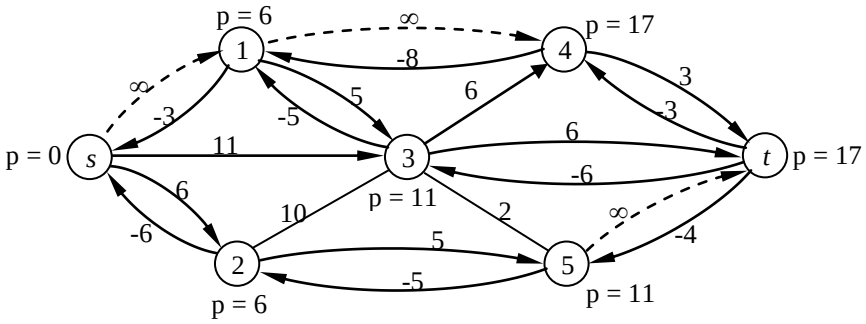


Figura 3.24

Pasul 2 Se determină drumurile de valoare minimă de la *s* la *t*.

$s \xrightarrow{4} 3 \xrightarrow{4} t$ $\qquad\qquad \theta_5 = 4$

Se propagă fluxul φ_3, adăugând la φ_2 un flux de 4 unităţi pe drumul $s \to 3 \to t$.

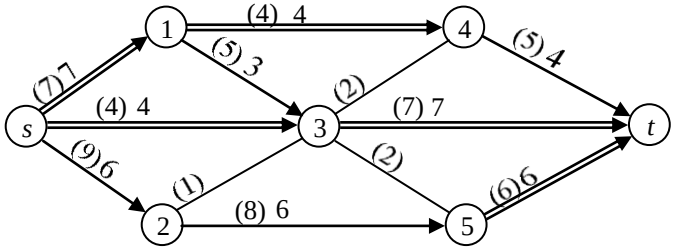


Figura 3.25

Valoarea fluxului este acum $v(\varphi_3) = v(\varphi_2) + 4 = 17$ unităţi de flux (figura 3.25).
Costul va fi $a(\varphi_3) = a(\varphi_2) + 4 \cdot 17 = 256$ unităţi monetare.

Iteraţia 4

Pasul 1. Se modifică costurile în graf în raport cu fluxul φ_3 şi obţinem situaţia din figura 3.26:

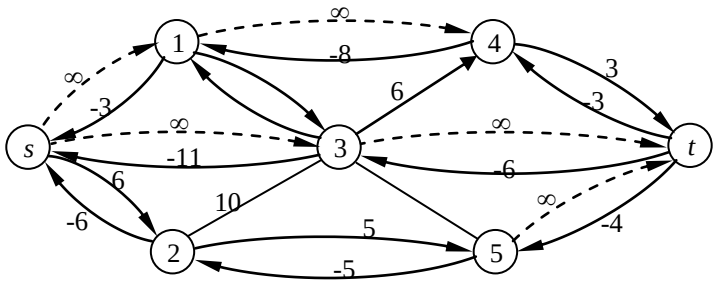


Figura 3.26

Pasul 2. Se determină drumurile de valoare minimă de la *s* la *t*.

$s \xrightarrow{2} 2 \xrightarrow{2} 5 \xrightarrow{1} 3 \to 4 \to t$ $\qquad\qquad \theta_6 = 1$

După propagarea unei unităţi de flux obţinem situaţia din figura 3.27:

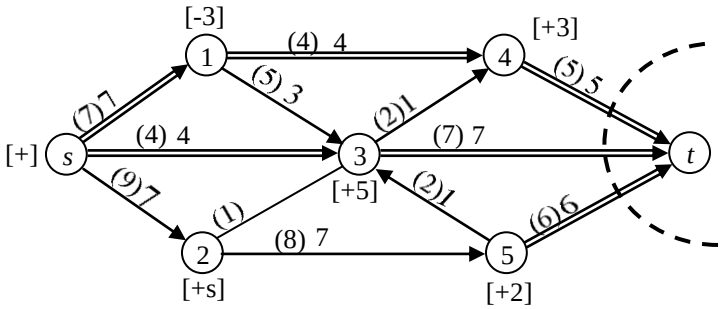


Figura 3.27

Valoarea fluxului este acum $v(\varphi_4) = v(\varphi_3) + 1 = 18$ unităţi de flux.
Costul va fi $a(\varphi_4) = a(\varphi_3) + 1 \cdot 22 = 278$ unităţi monetare.

Iteraţia 5

Pasul 1. Se modifică costurile în graf în raport cu fluxul φ_4 şi obţinem situaţia din figura 3.28:
Pasul 2. Se identifică drumurile de cost minim de la *s* la *t*. Deoarece nu mai există drumuri de cost

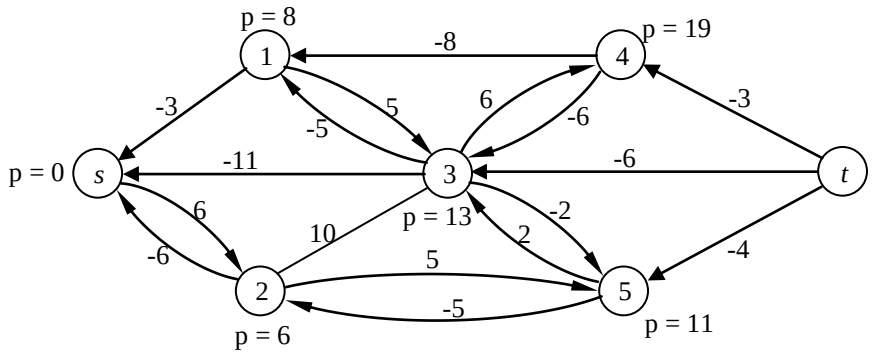


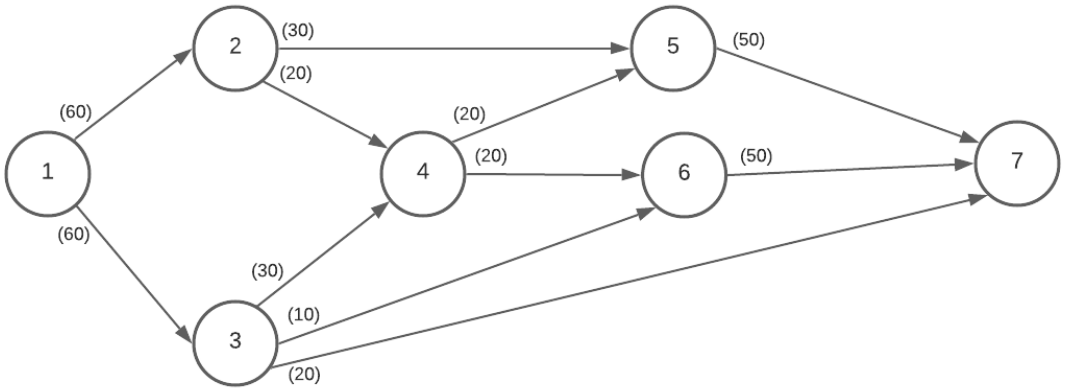
Figura 3.28

minim, fluxul din figura 3.28 este cel maxim.

Aplicând procedeul de marcaj, în figura 3.27, se observă că *t* nu mai poate fi marcat, deci fluxul este maxim.

Flux în rețele

Problema 1: Fluxul tehnologic al unei întreprinderi industriale se poate reprezenta printr-o rețea. Nodurile reprezintă secțiile întreprinderilor iar arcele (i, j) precedențele dintre ele. Numerele înscrise pe arce semnifică producția realizată în secția *i* care se poate prelucra în secția *j* în aceeași unitate de timp. Să se studieze necorelările între capacitățile secțiilor evidențiindu-le pe cele care sunt supradimensionate.



Rezolvare: Observăm că muchiile (1, 2) și (1, 3) au capacitățile egale cu 60, adică în total avem 120 de unități ce pot pleaca din secția 1 către celelalte secții trecând prin 2 și 3. Secția 2 poate procesa în total 50 de unități din care 30 pot pleca spre secția 5 iar 20 spre secția 4. Secția 4 poate procesa doar 40 de unități deși ar putea primi 20 de la secția 2 și 30 de la secția 3. Secția 4 trimite jumătate (20) din produse spre secția 5 iar cealaltă jumătate (20) către secția 6. Secțiile 3 și 5 pot procesa întreaga cantitate primită (60, respectiv 50). Secția 7 ar trebui să proceseze 120 de unități (50 care provin din secția 5, alte 50 din secția 6 și încă 20 din secția 3). Prin optimizări succesive vom încerca să transportăm toate cele 120 de unități de la nodul 1 la nodul 7 al rețelei. Pentru a rezolva această problemă, aplicăm algoritmul Ford- Fulkerson.

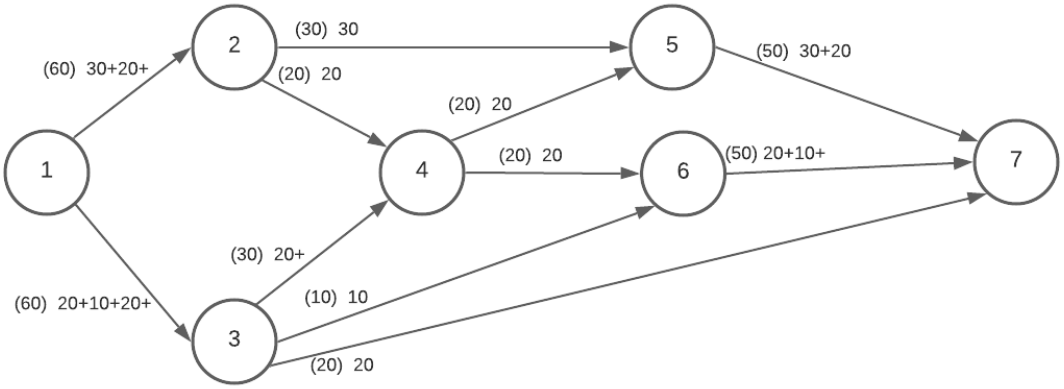
Et. 1. Construirea de drumuri de la nodul 1 la nodul 7.

Observăm următoarele drumuri și identificăm fluxul nenul de valoare cât mai mare ce se propagă în rețea de la secția 1 la secția 7:

Drum		*Flux propagat*	*Rute saturate*
$\mu_1 = (1, 2, 5, 7)$	$\rightarrow$	$\theta_1 = \min(60, 30, 50) = 30$	(2, 5)
$\mu_2 = (1, 2, 4, 5, 7)$	$\rightarrow$	$\theta_2 = \min(30, 20, 20, 20) = 20$	(2, 4), (4, 5), (5, 7)
$\mu_3 = (1, 3, 4, 6, 7)$	$\rightarrow$	$\theta_3 = \min(60, 30, 20, 50) = 20$	(4, 6)
$\mu_4 = (1, 3, 6, 7)$	$\rightarrow$	$\theta_4 = \min(40, 10, 30) = 10$	(3, 6)
$\mu_5 = (1, 3, 7)$	$\rightarrow$	$\theta_5 = \min(30, 20) = 20$	(3, 7)

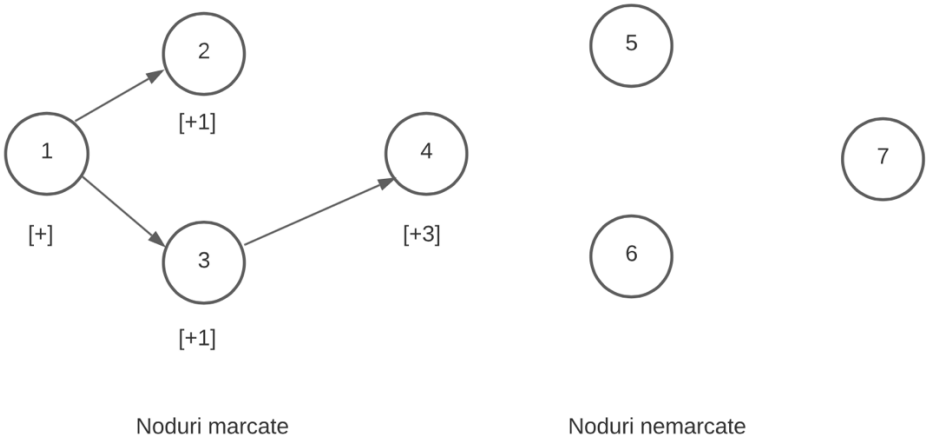
Fluxul complet propagat de-a lungul acestor drumuri are valoarea:

$$v(\varphi) = \sum_{i=1}^{5} \theta_i = 100\ produse$$



Întrucât arcele (1, 2), (1, 3), (3, 4) și (6, 7) sunt nesaturate, capacitățile secțiilor 1, 3 și 6 sunt supradimensionate. Secțiile 2 și 4 sunt subdimensionate (secția 2 ar putea primi 60 unități de flux dar poate procesa doar 50 iar secția 4 ar putea primi 50 unități de flux dar poate procesa doar 40).

Et. 2. Identificarea de lanțuri de augmentare de la nodul 1 la nodul 7 prin aplicarea procedeului de marcare. Observăm că se pot efectua următoarele marcaje:



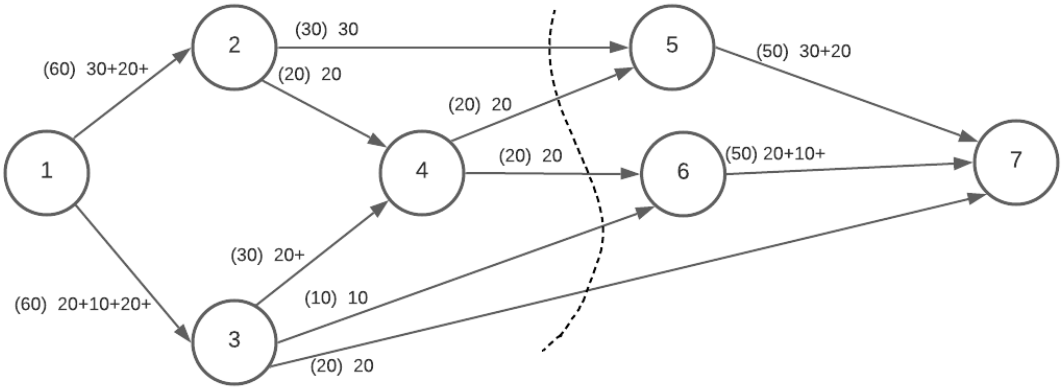
Fluxul curent, a cărui valoare este de 100 unităţi, este fluxul de valoare maximă pentru că nu poate fi marcat nodul final *7.* Valoarea sa fiind mai mică decât 120 înseamnă că secția 7 nu lucrează la întreaga sa capacitate (care ar trebui să fie de 120 de unități). Pentru ca secția 7 să lucreze la capacitatea sa maximă este necesar să mărim capacităţile unor rute de legătură. Aceste rute le identificăm *în tăietura de capacitate minimă*.

Et. 3: Determinarea unei tăieturi de valoare minimă

În etapa finală se pune în evidenţă tăietura de capacitate minimă, separând nodurile marcate de cele nemarcate. Având în vedere că $S^* = \{1, 2, 3, 4\}$ şi $T^* = \{5, 6, 7\}$ tăietura cuprinde arcele:

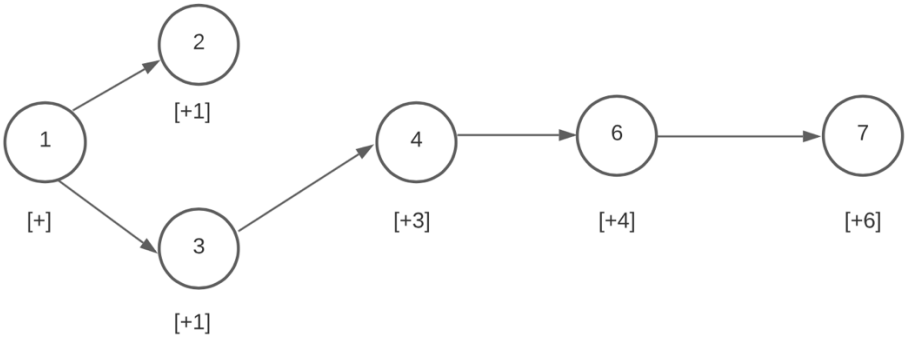
$$(S^*, T^*) = \{(2, 5) , (4, 5) , (4, 6) , (3, 6) , (3, 7)\}$$

Suma capacităţilor celor şase rute din tăietură este: c(S*, T*)= 30+20+20+10+20= 100, ceea ce confirmă faptul că fluxul curent este de valoare maximă.



Orice cantitate suplimentară de flux trebuie să treacă prin una sau mai multe rute din cele cinci ale tăieturii minime. Să presupunem că toate cele 20 unităţi de flux necesare pentru ca secția 7 să lucreze la capacitate maximă sunt repartizate pe arcul (4, 6) a cărei capacitate crește la 40. Prin urmare, capacitatea secției 4 va crește de la 40 la 60 unități de flux (secția 6 va primi 40 de unități de flux de la secția 4 și 10 de la secția 3). Alegerea acestui arc nu este întâmplătoare întrucât observăm că arcul (6,7) nu este saturat, secția 6 având posibilitatea de a procesa suplimentar 20 de unități de flux.

Reluăm etapa a 2-a din algoritmul Ford – Fulkerson încercând să identificăm noi lanțuri de augmentare cu ajutorul procedurii de marcare. Observăm că reușim să marcăm nodul 7 și prin urmare identificăm un lanț de augmentare μ_6 (care este de fapt un drum).



Avem astfel lanţul (drumul) de augmentare:

$$\mu_6 : 1 \to 3 \to 4 \to 6 \to 7$$

care unește secția 1 cu secția 7. Se observă că:

$$B = \{(1, 3),\ (3, 4), (4, 6), (6, 7)\},$$
$$C = \emptyset$$

întrucât toate arcele din lanț sunt marcate cu regula b).

De-a lungul acestui lanţ se propagă un flux de valoare:

$$\theta_6 = min\{c_{13} - \varphi_{13}, c_{34} - \varphi_{34}, c_{46} - \varphi_{46}, c_{67} - \varphi_{67}\}$$

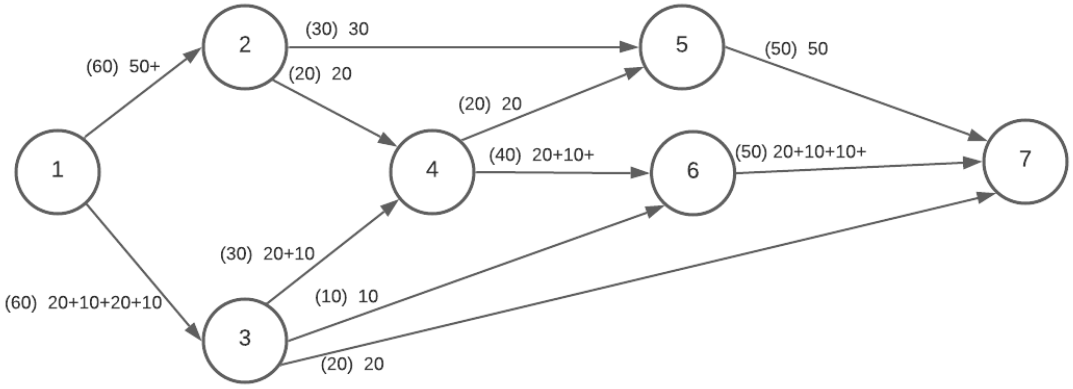
de unde:

$$\theta_6 = \min\{10, 10, 20, 20\} = 10\ produse.$$

Fluxurile pe arcele (1, 3), (3, 4), (4, 6) şi (6, 7) se măresc cu 10 unităţi. Arcele (1, 3) și (3, 4) devin saturate. În acest fel, în rețeaua de transport s-a obținut un nou flux de valoare:

$$v(\varphi) = \sum_{i=1}^{6} \theta_i = 110\ produse$$

Fluxul φ și rețeaua de transport cu valorile actualizate arată astfel:



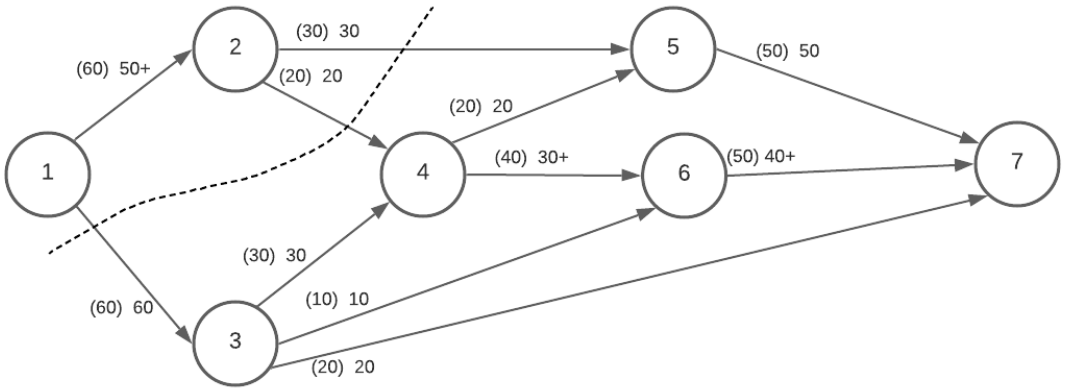
Reluăm etapa a 2-a din algoritmul Ford – Fulkerson încercând să identificăm noi lanțuri de augmentare cu ajutorul procedurii de marcare. Observăm că nu reușim să marcăm nodul 7 și prin urmare nu putem identifica un lanț de augmentare.

În etapa 3 se pune în evidenţă tăietura de capacitate minimă, separând nodurile marcate de cele nemarcate. Având în vedere că S^* = {1, 2} şi T^* = {3, 4, 5, 6, 7} tăietura cuprinde arcele:

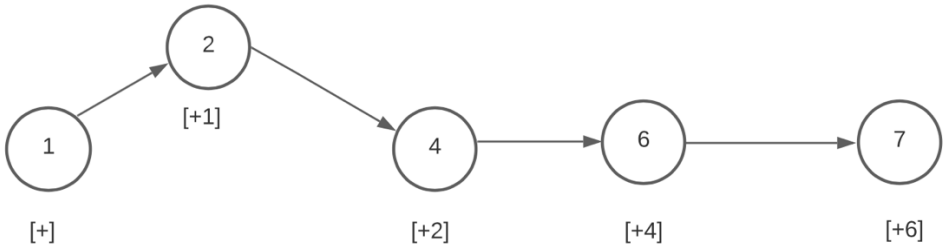
$$(S^*, T^*) = \{(2, 5), (2, 4), (1, 3)\}$$

Suma capacităţilor celor şase rute din tăietură este: c(S*, T*)= 30+20+60= 110 produse, ceea ce confirmă faptul că fluxul curent este de valoare maximă.

Orice cantitate suplimentară de flux trebuie să treacă prin una sau mai multe rute din cele trei ale tăieturii minime. Să presupunem că mărim capacitatea muchiei (2, 4) de la 20 la 30 unități. Prin urmare, capacitatea secției 2 va crește de la 50 la 60 unități de flux (secția 4 va primi 30 de unități de flux de la secția 2 în loc de 20, în plus față de cele 30 primite de la secția 3). Alegem acest arc deoarece secția 4 poate procesa cele 10 unități de flux pe care le-ar primi suplimentar de la secția 2 întrucât arcul (4, 6) nu este saturat.



Reluăm etapa a 2-a din algoritmul Ford – Fulkerson încercând să identificăm noi lanțuri de augmentare cu ajutorul procedurii de marcare. Observăm că reușim să marcăm nodul 7 și prin urmare identificăm un lanț de augmentare μ_7 care este de fapt un drum.



Avem astfel lanţul (drumul) de augmentare:

$$\mu_7 : 1 \to 2 \to 4 \to 6 \to 7$$

care unește secția 1 cu secția 7. Se observă că:

$B = \{(1, 2),\ (2, 4), (4, 6), (6, 7)\},$
$C = \emptyset$

întrucât toate arcele din lanț sunt marcate cu regula b). De-a lungul acestui lanţ se propagă un flux de valoare:

$$\theta_7 = min\{c_{12} - \varphi_{12}, c_{24} - \varphi_{24}, c_{46} - \varphi_{46}, c_{67} - \varphi_{67}\}$$

de unde:

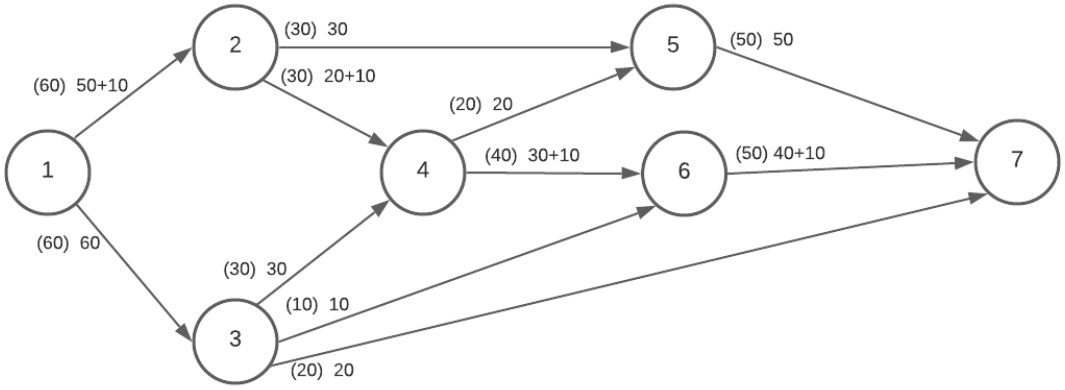
$$\theta_7 = \min\{10, 10, 10, 10\} = 10\ produse.$$

Fluxurile pe arcele (1, 2), (2, 4), (4, 6) şi (6, 7) se măresc cu 10 unităţi și toate devin saturate.

Valoarea noului flux propagat în reţea este:

$$v(\varphi) = \sum_{i=1}^{7} \theta_i = 120\ produse$$

Adăugăm cele 10 unități de flux pe muchiile drumului μ_7. Rețeaua de transport cu valorile actualizate arată astfel:

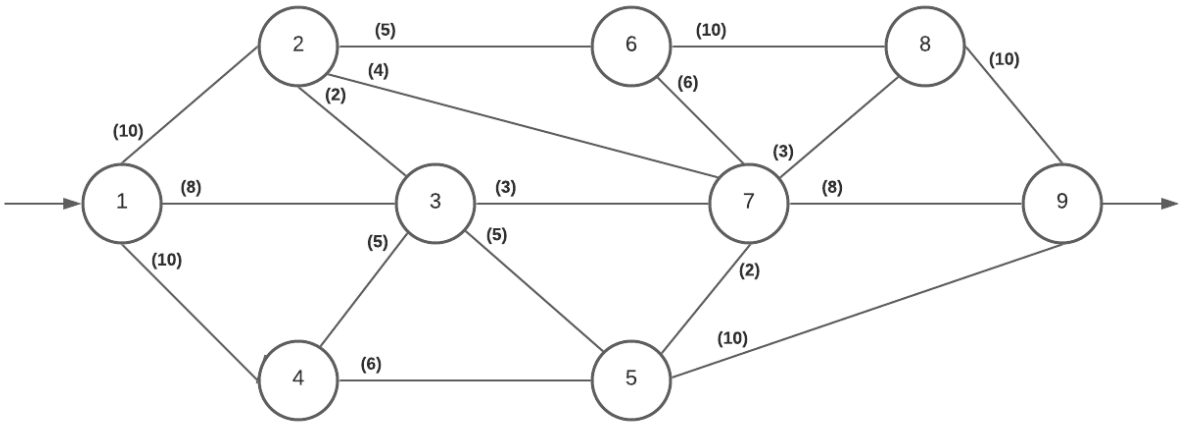


În acest moment toate secțiile din acestă întreprindere lucrează la capacitate maximă iar valoarea fluxului propagat în această rețea de transport este de 120 de produse.

Problema 2: Un proces chimic dispune de o rețea de conducte pentru transferul unui produs lichid dintr-o parte în alta a rețelei. Capacitățile conductelor sunt indicate în paranteze și sunt valabile în ambele sensuri de parcurgere.

Se cere:

i) Care este fluxul maxim propagat în această rețea de la nodul 1 la nodul 9?
ii) Care este fluxul care trece prin conducta dintre nodul 3 și nodul 7? Dar între nodul 7 și 8? Între 6 și 7?
iii) Ce trebuie să facă un decident care dorește să mărească fluxul propagat prin această rețea?



Rezolvare:

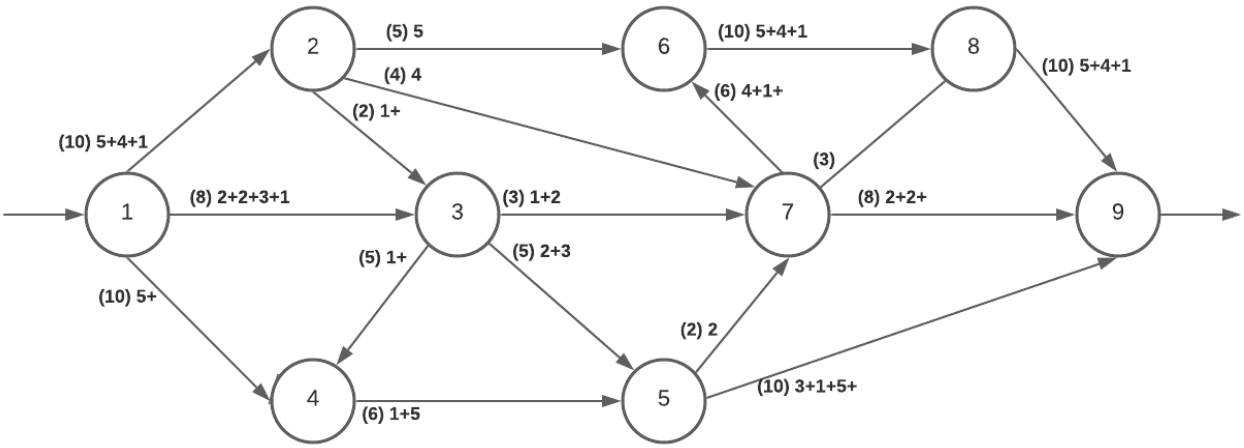
i) Pentru a identifica fluxul maxim propagat în această rețea de la nodul 1 la nodul 9, aplicăm algoritmul Ford- Fulkerson.

Et. 1. Construirea de drumuri de la nodul 1 la nodul 9.

Observăm următoarele drumuri și calculăm fluxul propagat în rețea pe fiecare drum de la nodul 1 la nodul 9 utilizând formula:

$$\theta = \min_{u \epsilon \mu}[c(u) - \varphi(u)]$$

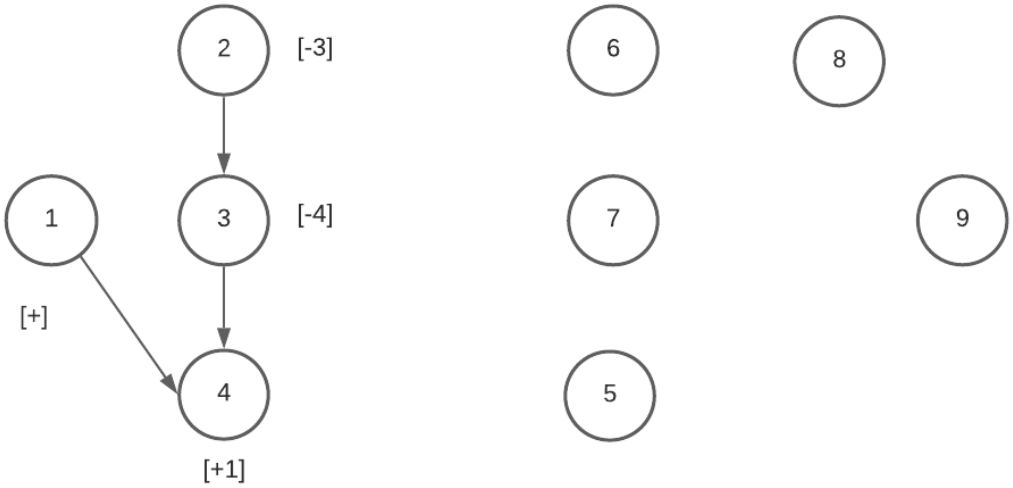
Drum		*Flux propagat*	*Rute saturate*
μ_1 = (1, 2, 6, 8, 9)	→	θ_1 = min (10, 5, 10, 10) = 5	(2, 6)
μ_2 = (1, 2, 7, 6, 8, 9)	→	θ_2 = min (5, 4, 6, 5, 5) = 4	(2, 7)
μ_3 = (1, 2, 3, 7, 6, 8, 9)	→	θ_3 = min (1, 2, 3, 2, 1, 1) = 1	(1, 2), (6, 8), (8, 9)
μ_4 = (1, 3, 7, 9)	→	θ_4 = min (8, 2, 8) = 2	(3, 7)
μ_5 = (1, 3, 5, 7, 9)	→	θ_5 = min (6, 5, 2, 6) = 2	(5, 7)
μ_6 = (1, 3, 5, 9)	→	θ_6 = min (4, 3, 10) = 3	(3, 5)
μ_7 = (1, 3, 4, 5, 9)	→	θ_7 = min (1, 5, 6, 7) = 1	(1, 3)
μ_8 = (1, 4, 5, 9)	→	θ_8 = min (10, 5, 6) = 5	(4, 5)



Fluxul complet propagat de-a lungul acestor drumuri are valoarea:

$$v(\varphi) = \sum_{i=1}^{8} \theta_i = 5 + 4 + 1 + 2 + 2 + 3 + 1 + 5 = 23 \text{ unități de flux}$$

Et. 2. Identificarea lanțurilor de augmentare de la nodul 1 la nodul 9 prin aplicarea procedeului de marcare. Observăm că se pot efectua următoarele marcaje:



Noduri marcate Noduri nemarcate

Deoarece nodul final 9 nu poate fi marcat rezultă că fluxul complet propagat $\nu(\varphi) = 23$ este maxim.

Et. 3: Determinarea unei tăieturi de valoare minimă

În etapa finală se pune în evidenţă tăietura de capacitate minimă, separând nodurile marcate de cele nemarcate. Având în vedere că S^* = {1, 2, 3, 4} şi T^* = {5, 6, 7, 8, 9} tăietura cuprinde arcele:

$$(S^*, T^*) = \{(2, 6) , (2, 7) , (3, 7) , (3, 5) , (4, 5)\}$$

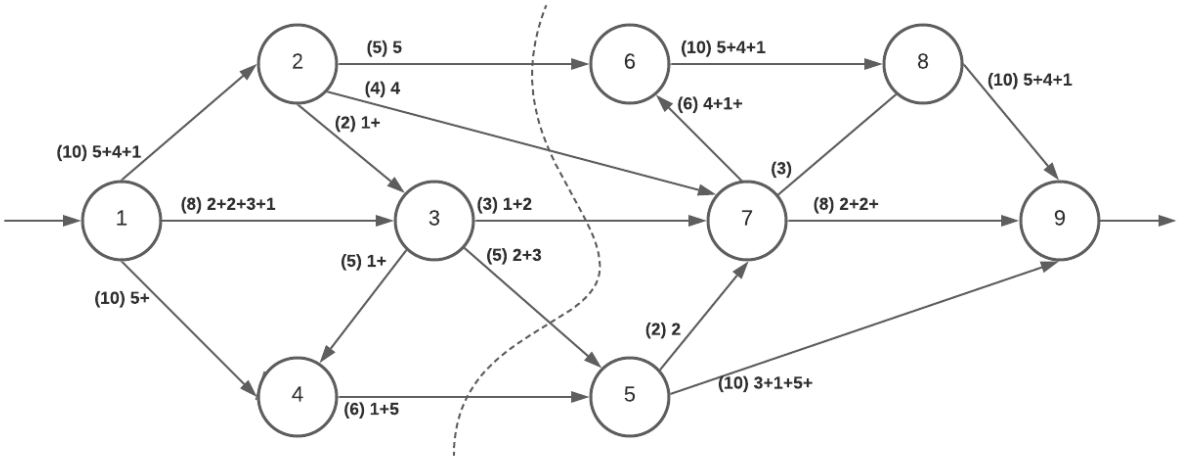
iar capacitatea tăieturii este:

$$c(S^*, T^*) = \sum_{u \in (S^*, T^*)} c(u) = 5 + 4 + 3 + 5 + 6 = 23$$

Conform teoremei Ford- Fulkerson, dacă se verifică relația $\nu(\varphi^*) = c(S^*, T^*) = 23$ atunci fluxul are valoare maximă.

ii) Care este fluxul care trece prin conducta dintre nodul 3 și nodul 7? Dar între nodul 7 și 8? Între 6 și 7?

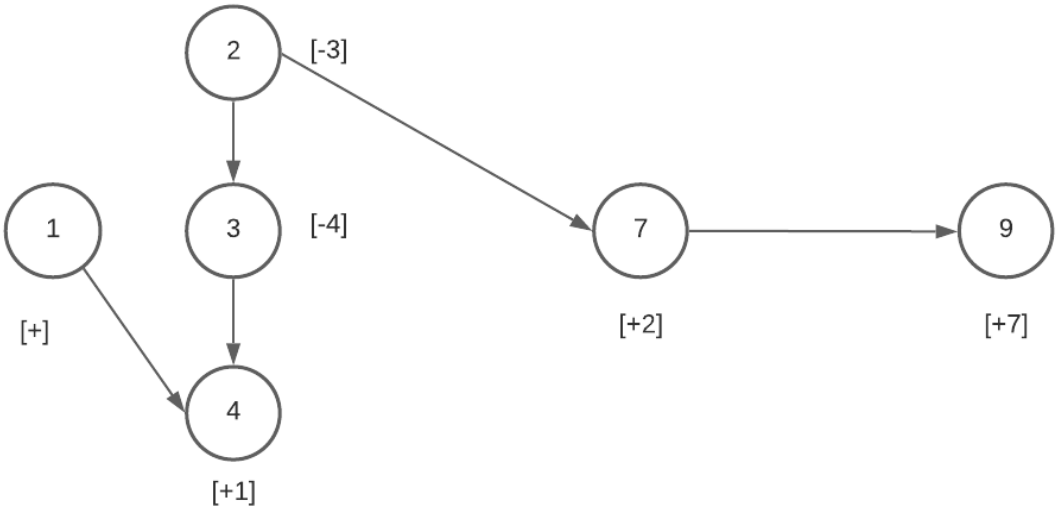
Prin conducta dintre nodul 3 și nodul 7 trec 3 unități de flux, această rută fiind saturată. Prin conducta dintre nodul 7 și nodul 8 nu trece nici o unitate de flux, aceasta fiind goală. Prin conducta ce leagă nodurile 6 și 7 trec 0 unități de flux pe sensul de la 6 la 7 (și 5 unități de flux în sensul de la nodul 7 la nodul 6).



iii) Dacă un decident dorește să mărească fluxul propagat în această rețea de la nodul 1 la nodul 9, atunci va trebui să mărească capacitatea unuia sau mai multor arce de pe tăietură.

Să presupunem că se va înlocui conducta ce conectează nodurile 2 și 7 de capacitate $c_{27} = 4$ cu una de capacitate egală cu 7 ($c'_{27} = 7$).

Reluăm etapa a 2-a din algoritmul Ford – Fulkerson încercând să identificăm noi lanțuri de augmentare cu ajutorul procedurii de marcare. Observăm că reușim să marcăm nodul 7 și prin urmare identificăm un lanț de augmentare μ_7 care este de fapt un drum.



Avem astfel lanţul de augmentare:

$$\lambda_9 : 1 \rightarrow 4 \leftarrow 3 \leftarrow 2 \rightarrow 7 \rightarrow 9$$

care conectează nodul 1 cu nodul 9. Se observă că:

$B = \{(1, 4),\ (2, 7), (7, 9)\}$,
$C = \{(3, 4),\ (2, 3)\}$.

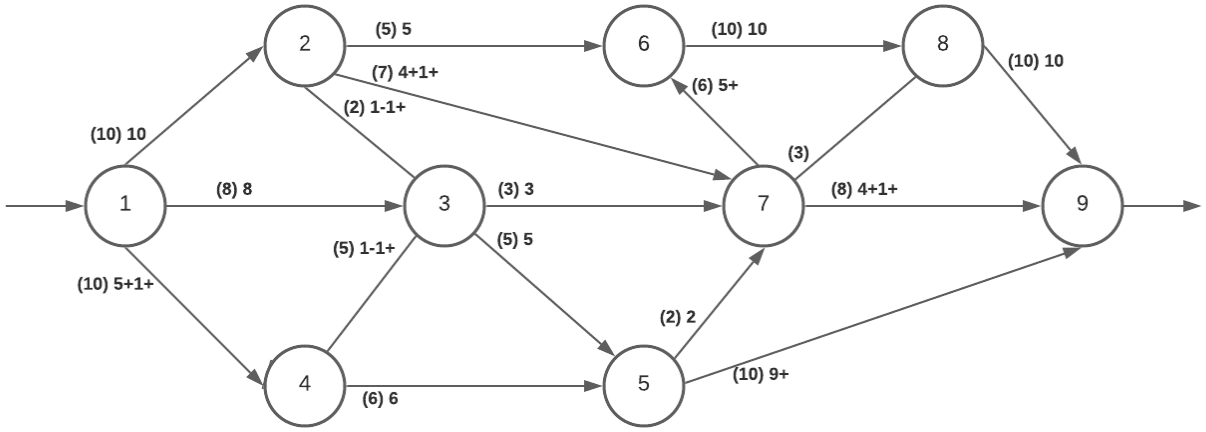
De-a lungul acestui lanţ se propagă un flux de valoare:

$$\theta_9 = min\{\min[c_{14} - \varphi_{14}, c_{27} - \varphi_{27}, c_{79} - \varphi_{79}], \min[\varphi_{34}, \varphi_{23}]\}$$

de unde:

$$\theta_9 = \min\{5, 3, 4, 1, 1\} = 1.$$

Fluxurile pe arcele din *B* ((1, 4), (2, 7) şi (7, 9)) se măresc cu 1 unitate iar cele din *C* ((3, 4) și (2, 3)) se reduc cu 1 unitate de flux. Rețeaua de transport cu valorile actualizate arată astfel:

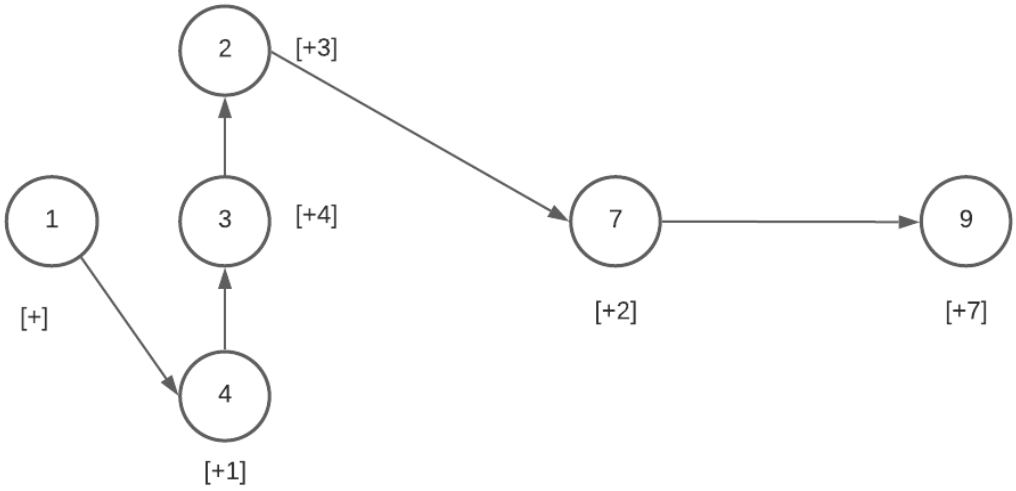


Valoarea noului flux propagat în reţea este:

$$v(\varphi'') = \sum_{i=1}^{9} \theta_i = 23 + 1 = 24$$

Pe graful cu valorile actualizate observăm că, datorită reducerii cu 1 unitate de flux a alocărilor pe muchiile (3, 4) și (2, 3), acestea sunt acum neorientate.

Aplicăm procedura de marcare și identificăm următorul lanț de augmentare (care este de fapt un drum) de la nodul 1 la nodul 9:



$\lambda_{10} : 1 \to 4 \to 3 \to 2 \to 7 \to 9.$

Se observă că:

$B = \{(1, 4), (4, 3), (3, 2), (2, 7), (7, 9)\},$
$C = Ø.$

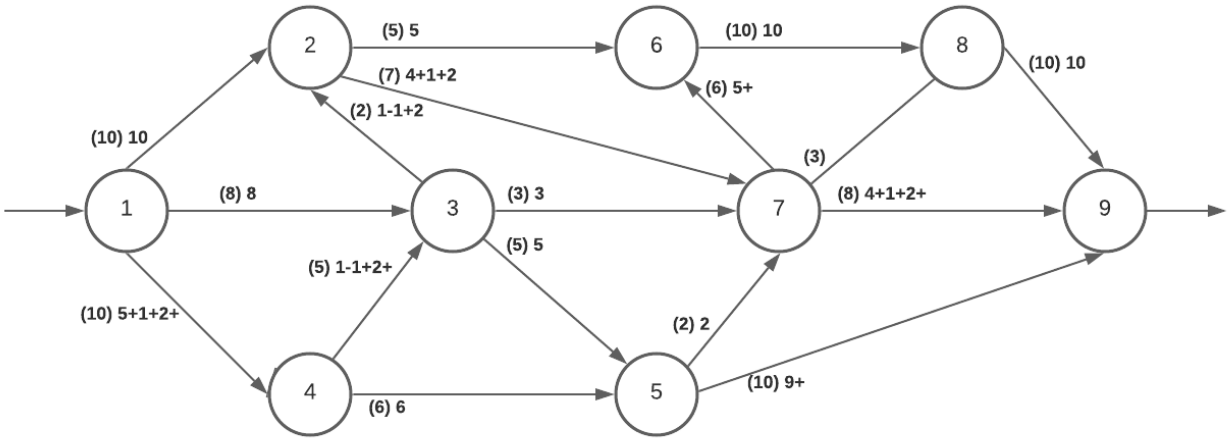
De-a lungul acestui lanţ se propagă un flux de valoare:

$$\theta_{10} = min\{ c_{14} - \varphi_{14}, c_{43} - \varphi_{43}, c_{32} - \varphi_{32}, c_{27} - \varphi_{27}, c_{79} - \varphi_{79}\}$$

de unde:

$$\theta_{10} = \min\{4, 5, 2, 2, 3\} = 2.$$

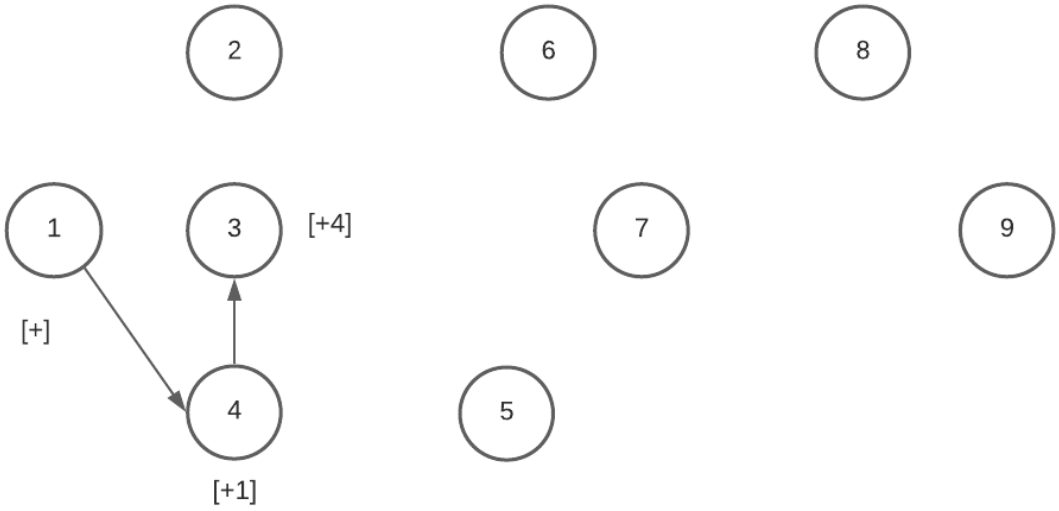
Fluxurile pe arcele din B se măresc cu 2 unități. Rețeaua de transport cu valorile actualizate arată astfel:



Fluxul complet propagat de-a lungul acestor drumuri are valoarea:

$$v(\varphi) = \sum_{i=1}^{10} \theta_i = 23 + 1 + 2 = 26 \text{ unități de flux}$$

Continuăm identificarea lanțurilor de augmentare prin aplicarea procedeului de marcare. Observăm că se pot efectua următoarele marcaje:



Noduri marcate Noduri nemarcate

Deoarece nodul final 9 nu poate fi marcat rezultă că fluxul complet propagat $v(\varphi) = 26$ este maxim.

Et. 3: Determinarea unei tăieturi de valoare minimă

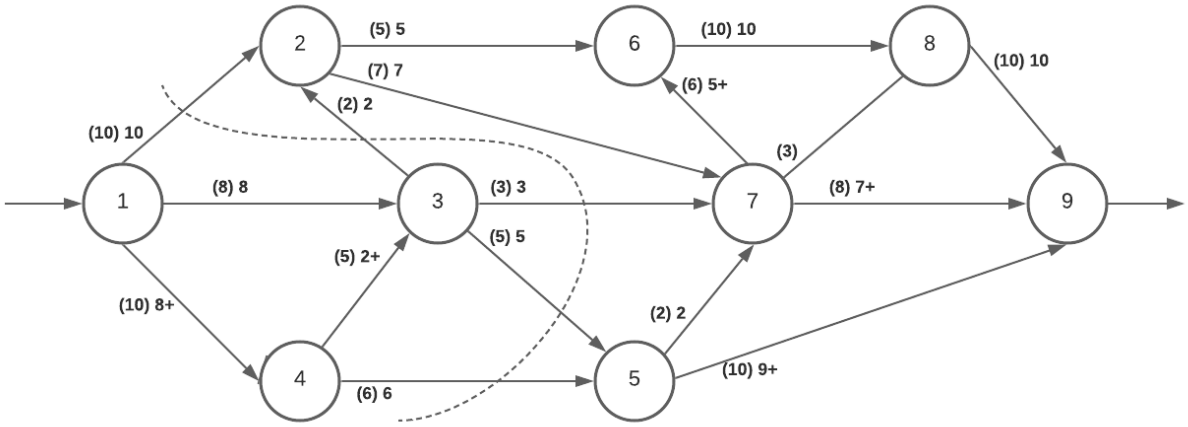
În etapa finală se pune în evidenţă tăietura de capacitate minimă, separând nodurile marcate de cele nemarcate. Având în vedere că S^* = {1, 3, 4} şi T^* = {2, 5, 6, 7, 8, 9} tăietura cuprinde arcele:

$$(S^*, T^*) = \{(1, 2) , (3, 2) , (3, 7) , (3, 5) , (4, 5)\}$$

iar capacitatea tăieturii este:

$$c(S^*, T^*) = \sum_{u \in (S^*, T^*)} c(u) = 10 + 2 + 3 + 5 + 6 = 26$$

Conform teoremei Ford- Fulkerson, dacă se verifică relația $v(\varphi^*) = c(S^*, T^*) = 26$ atunci fluxul are valoare maximă.



Dacă se dorește mărirea fluxului propagat în această rețea, atunci va trebui să mărim capacitatea unuia sau mai multor arce de pe tăietura curentă.

Problema 3: Problema de cerere si oferta

O cantitate de marfă se află disponibilă în centrele A, B, C în cantitățile 30, 40 și 36 u.c. și este solicitată în centrele X, Y, Z în cantitățile 36, 38 și 34. Între surse și destinații există legături directe și transportul se efectuează cu mijloace de transport auto cu capacități limitate, conform matricii urmatoare:

	X	Y	Z
A	10	14	8
B	24	-	18
C	10	20	6

Pe ruta (B, Y) nu se face transport.

Se cere:

i) să se determine un plan optim de transport astfel încât să fie asigurată cererea în centrele X, Y și Z;
ii) sunt suficiente mijloacele de transport auto disponibile pentru a transporta marfa disponibilă în centrele A, B și C către X, Y și Z?

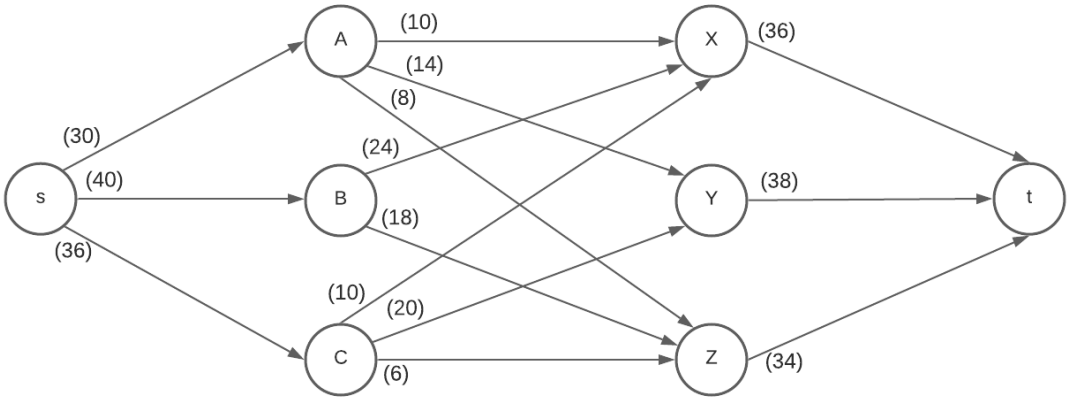
Rezolvare:

Începem prin a trasa rețeaua de transport. Întrucât avem 3 centre de tip sursă (A, B și C), vom adăuga un nod fictiv *s* de la care ducem arce orientate de la *s* către cele 3 surse. Cantitățile disponibile de marfă în cele 3 centre vor reprezenta capacitățile acestor muchii.

În același fel procedăm și cu cele 3 centre de tip destinație (X, Y și Z): vom adăuga un nod fictiv *t* pe care îl conectăm cu cele 3 destinații prin arce orientate către nodul *t* pe care scriem cantitățile solicitate care vor reprezenta capacitățile acestor muchii.

Conexiunile dintre centrele de tip sursă cu cele de tip destinație se vor identifica din tabel iar capacitățile acestor muchii vor fi date de capacitățile mijloacelor auto utilizate pentru transportul acestei mărfi.

Rețeaua de transport va arăta astfel:



Cantitatea totală de marfă disponibilă în A, B și C (adică oferta totală) este de 30+40+36= 106 u.c., iar cantitatea solicitată în X, Y și Z (adică cererea totală) este de 36+38+34=108 u.c. Întrucât Cererea = 108 > 106 = Oferta => nu vom reuși să satisfacem integral cererea tuturor celor 3 centre X, Y și Z.

i) Pentru a determina cantitatea maximă de marfă ce poate fi transportată de la centrele A, B și C către centrele X, Y și Z ținând cont de mijloacele auto disponibile, vom aplica algoritmul Ford- Fulkerson.

Et. 1. Construirea de drumuri de la nodul s la nodul t.

Observăm următoarele drumuri și calculăm fluxul propagat în rețea pe fiecare drum de la nodul s la nodul t utilizând formula:

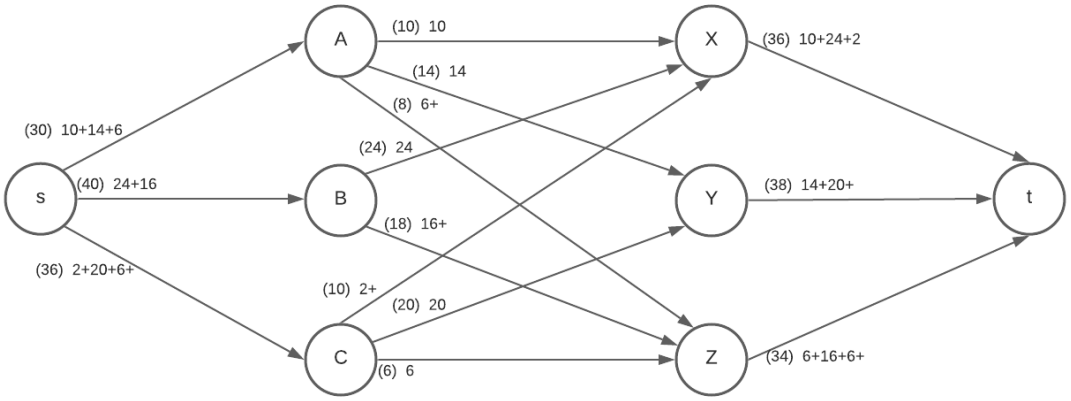
$$\theta = \min_{u \epsilon \mu}[c(u) - \varphi(u)]$$

Drum		*Flux propagat*	*Rute saturate*
μ_1 = (s, A, X, t)	→	θ_1 = min (30, 10, 36) = 10	(A, X)
μ_2 = (s, A, Y, t)	→	θ_2 = min (20, 14, 38) = 14	(A, Y)
μ_3 = (s, A, Z, t)	→	θ_3 = min (6, 8, 34) = 6	(s, A)
μ_4 = (s, B, X, t)	→	θ_4 = min (40, 24, 26) = 24	(B, X)
μ_5 = (s, B, Z, t)	→	θ_5 = min (16, 18, 28) = 16	(s, B)
μ_6 = (s, C, X, t)	→	θ_6 = min (36, 10, 2) = 2	(X, t)
μ_7 = (s, C, Y, t)	→	θ_7 = min (34, 20, 24) = 20	(C, Y)
μ_8 = (s, C, Z, t)	→	θ_8 = min (14, 6, 12) = 6	(C, Z)

Faptul că muchia (s, A) este saturată semnifică faptul că centrul A nu mai are marfă pe stoc (A a trimis 10 u.c. către X, 14 u.c. către Y și 6 u.c. către Z din cantitatea disponibilă de 40 u.c.).

Deoarece muchia (A,Y) este saturată, tirul care transportă marfa de la centrul A la centrul Y este încărcat la capacitatea sa maximă de 14 u.c.

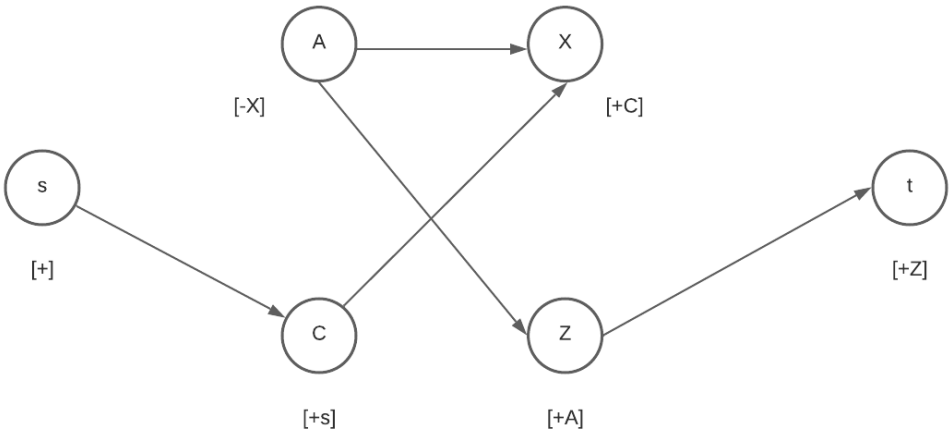
Muchia (X, t) este saturată, ceea ce înseamnă că cererea centrului X de 36 u.c. este integral satisfacută (X primește 10 u.c. de la A, 24 u.c. de la B și 2 u.c. de la C).



Fluxul complet propagat de-a lungul acestor drumuri are valoarea:

$$v(\varphi) = \sum_{i=1}^{8} \theta_i = 10 + 14 + 6 + 24 + 16 + 2 + 20 + 6 = 98\, u.c.$$

Et. 2. Identificarea lanțurilor de augmentare de la nodul s la nodul t prin aplicarea procedeului de marcare. Observăm că se pot efectua următoarele marcaje:



Reușim să marcăm nodul t și prin urmare identificăm un lanț de augmentare λ_9 care conectează nodul s cu nodul t:

$$\lambda_9 : s \rightarrow C \rightarrow X \leftarrow A \rightarrow Z \rightarrow t$$

. Se observă că:

$B = \{(s, C), (C, X), (A, Z), (Z, t)\}$,
$C = \{(A, X)\}$.

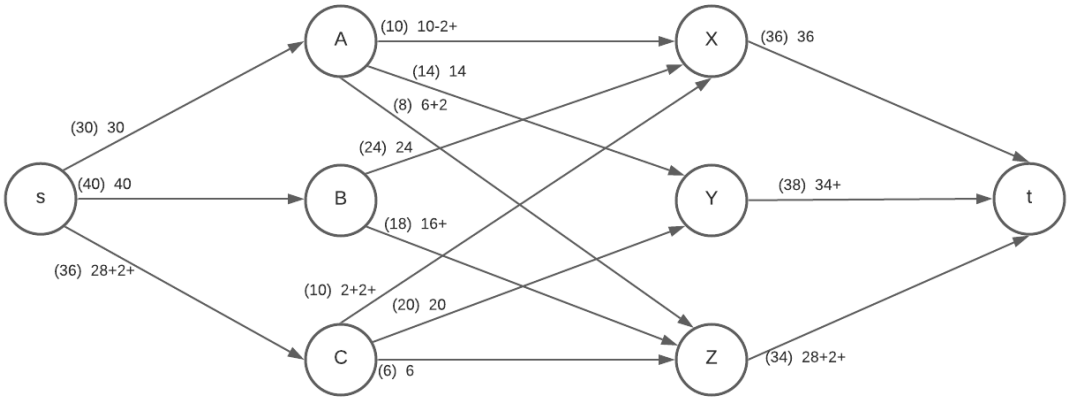
De-a lungul acestui lanţ se propagă un flux de valoare:

$$\theta_9 = min\{[c_{sc} - \varphi_{sc}, c_{cx} - \varphi_{cx}, c_{Az} - \varphi_{Az}, c_{zt} - \varphi_{zt}], [\varphi_{Ax}]\}$$

de unde:

$$\theta_9 = \min\{(36 - 28), (10 - 2), (8 - 6), (34 - 28), 10\} = 2\ u.c.$$

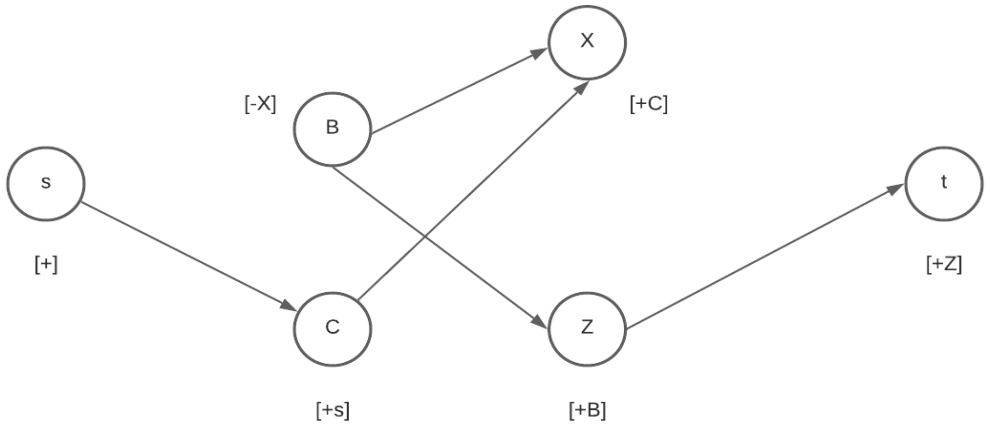
Fluxurile pe arcele din B se măresc cu 2 u.c. iar cele din C se reduc cu 2 u.c. Rețeaua de transport cu valorile actualizate arată astfel:



Valoarea noului flux propagat în reţea este:

$$v(\varphi) = \sum_{i=1}^{9} \theta_i = 98 + 2 = 100\ u.c.$$

Reluăm procedura de marcare pentru a identifica noi lanțuri de augmentare.



Reușim să marcăm nodul t și prin urmare identificăm un nou lanț de augmentare λ_{10} care conectează nodul s cu nodul t:

$$\lambda_{10} : s \rightarrow C \rightarrow X \leftarrow B \rightarrow Z \rightarrow t$$

. Se observă că:

$$B = \{(s, C), (C, X), (B, Z), (Z, t)\},$$
$$C = \{(B, X)\}.$$

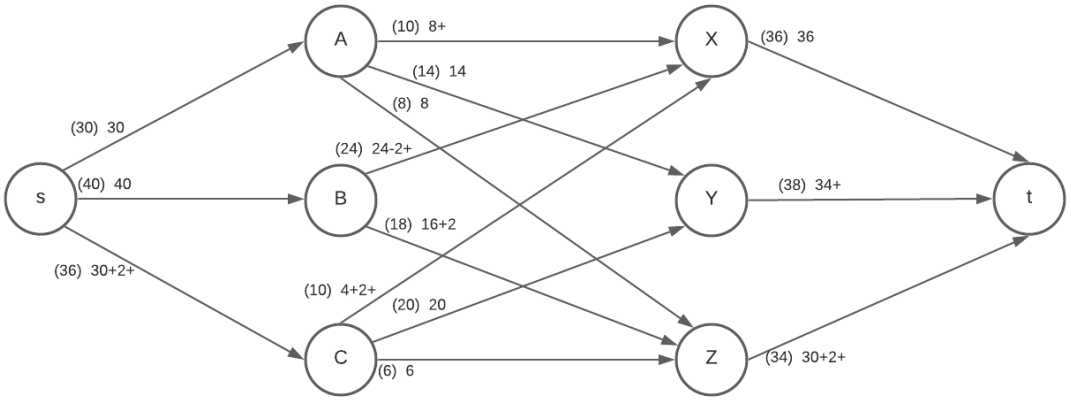
De-a lungul acestui lanţ se propagă un flux de valoare:

$$\theta_{10} = min\{\min [c_{sc} - \varphi_{sc}, c_{cx} - \varphi_{cx}, c_{Bz} - \varphi_{Bz}, c_{zt} - \varphi_{zt}], [\varphi_{Bx}]\}$$

de unde:

$$\theta_{10} = \min\{(36 - 30), (10 - 4), (18 - 16), (34 - 30), 24\} = 2\ u.c.$$

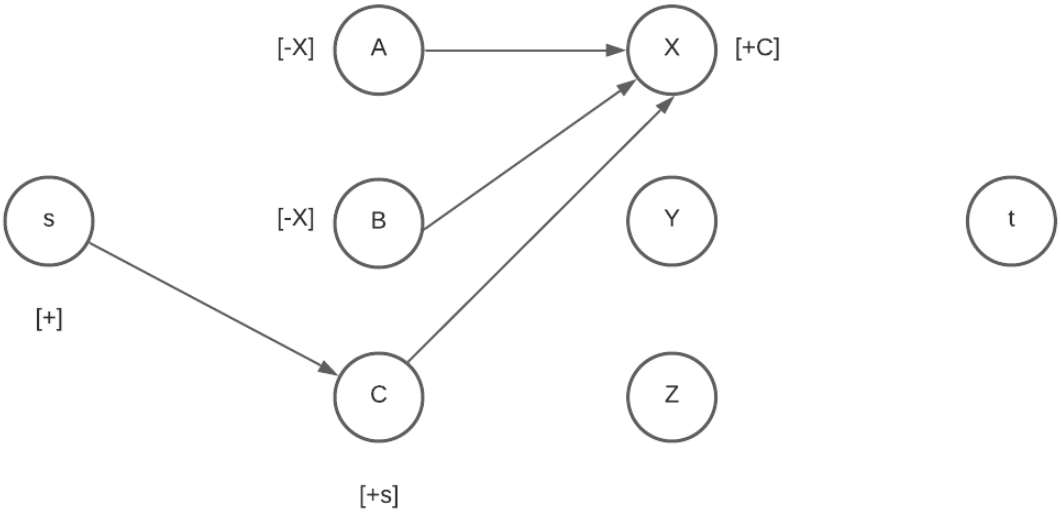
Fluxurile pe arcele din *B* se măresc cu 2 u.c. iar cele din *C* se reduc cu 2 u.c. Rețeaua de transport cu valorile actualizate arată astfel:



Valoarea noului flux propagat în reţea este:

$$v(\varphi) = \sum_{i=1}^{10} \theta_i = 100 + 2 = 102\ u.c.$$

Reluăm procedura de marcare pentru a identifica noi lanțuri de augmentare și constatăm că nodul t nu poate fi marcat:



Prin urmare fluxul propagat este maxim. Dar $\nu(\varphi) = 102 < 108$ ceea ce înseamnă că nu toate cererile sunt acoperite integral (Y primește doar 34 u.c. din cele 38 solicitate iar Z doar 32 u.c. din 34)

Et. 3: Determinarea unei tăieturi de valoare minimă

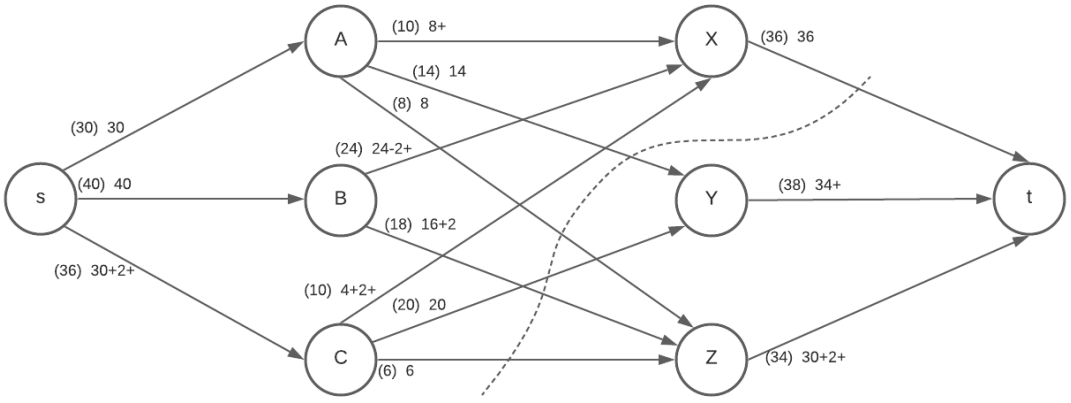
Pentru a identifica tăietura de capacitate minimă, separăm nodurile marcate de cele nemarcate.

Mulțimea nodurilor marcate este S^* = {s, C, X, A, B} şi mulțimea nodurilor nemarcate este T^* = {Y, Z, t}. Tăietura cuprinde arcele:

$$(S^*, T^*) = \{(X, t), (A, Y) , (A, Z), (C, Y) , (B, Z) , (C, Z), \}$$

iar capacitatea tăieturii este:

$$c(S^*, T^*) = \sum_{u \in (S^*, T^*)} c(u) = 36 + 14 + 8 + 20 + 18 + 6 = 102 \text{ u.c.}$$



Conform teoremei Ford- Fulkerson, întrucât se verifică relația $\nu(\varphi^*) = c(S^*, T^*) = 102\ u.c.$, fluxul are valoare maximă.

Un plan optim de transport, ținând cont de limitările curente ale rețelei de transport, presupune următoarele:

- depozitul A trimite 8 u.c. către destinația X, 14 u.c. către Y și 8 către Z; stocul lui A=0;
- depozitul B trimite 22 u.c. către destinația X și 18 către Z; stocul lui B=0;
- depozitul C trimite 6 u.c. către destinația X, 20 u.c. către Y și 6 către Z; stocul lui C=4.
- centrul X va avea cererea de 36 u.c. satisfăcută integral primind 8 u.c de la A, 22 u.c. de la B și 8 u.c. de la C;
- centrul Y va avea cererea de 38 u.c. satisfăcută parțial primind 14 u.c de la A și 20 u.c. de la C; ar mai avea nevoie să primească încă 4 u.c.
- centrul Z va avea cererea de 34 u.c. satisfăcută parțial primind 8 u.c de la A, 18 u.c. de la B și 6 u.c. de la C; ar mai avea nevoie de încă 2 u.c.

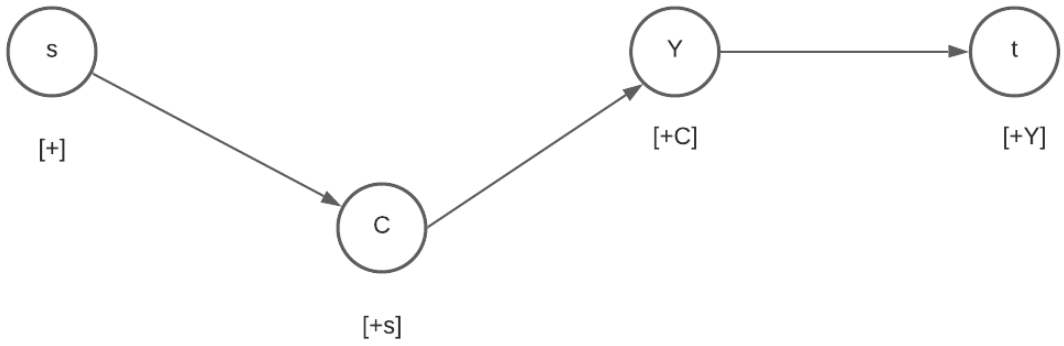
ii) Sunt suficiente mijloacele de transport auto disponibile pentru a transporta marfa disponibilă în centrele A, B și C către X, Y și Z?

Cele 4 u.c. aflate pe stoc în depozitul C ar putea fi trimise către centrul Y și/sau Z dar mijloacele de transport auto care asigură transportul mărfii de la C către Y și Z sunt deja încărcate la capacitate maximă.

Dacă un decident dorește să mărească fluxul propagat în această rețea prin transmiterea celor 4 u.c. de la C către Y și sau Z, atunci va trebui să mărească capacitatea unuia sau mai multor arce de pe tăietură, adică să suplimenteze mijloacele de transport auto de pe muchiile tăieturii astfel încât întreaga cantitate de marfă să poată fi transportată .

Observăm că în C mai avem 4 u.c. pe stoc, iar în Y avem nevoie de 4 u.c. și în Z de 2 u.c. Să presupunem că va alege să trimită toate cele 4 u.c. existente în stoc la centrul C către centrul Y pentru a satisface integral cererea centrului Y. În acest scop, va suplimenta mijloacele de transport auto ce asigură transportul mărfii de la C la Y astfel încât acestea să aibă capacitatea de 24 u.c. $(c'_{CY} = 24)$.

Continuăm identificarea lanțurilor de augmentare prin aplicarea procedeului de marcare. Observăm că se pot efectua următoarele marcaje:



Întrucât reușim să marcăm nodul *t* , am identificat un nou lanț de augmentare (care este de fapt un drum) de la *s* la *t*.

$$\lambda_{11} : s \rightarrow C \rightarrow Y \rightarrow t$$

. Se observă că:

$$B = \{(s, C), (C, Y), (Y, t)\},$$
$$C = \emptyset$$

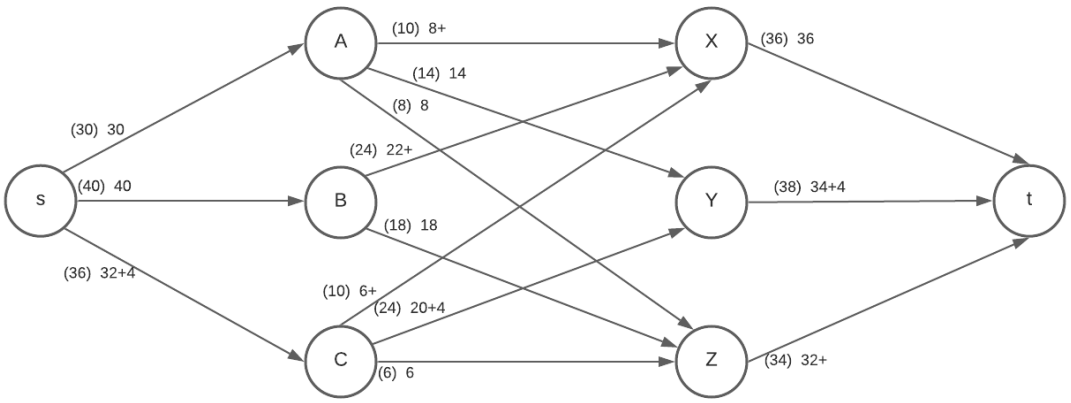
De-a lungul acestui lanţ se propagă un flux de valoare:

$$\theta_{11} = min\{ c_{sc} - \varphi_{sc}, c_{cy} - \varphi_{cy}, c_{yt} - \varphi_{yt}\}$$

de unde:

$$\theta_{11} = \min\{(36 - 32), (24 - 20), (38 - 34)\} = 4\ u.c.$$

Fluxurile pe arcele din *B* se măresc cu 4 u.c. Rețeaua de transport cu valorile actualizate arată astfel:



Valoarea noului flux propagat în reţea este:

$$v(\varphi) = \sum_{i=1}^{11} \theta_i = 102 + 4 = 106\ u.c.$$

În acest moment am epuizat toate stocurile din depozitele A, B și C; cererea centrelor X și Y a fost satisfăcută integral iar Z rămâne cu o cerere nesatisfăcută de 2 u.c.

CAPITOLUL 4
OPTIMIZAREA PROCESELOR ECONOMICE UTILIZÂND PROGRAMAREA LINIARĂ

NOTĂ: Suportul de curs al Capitolelor 4 şi 5 are la bază lucrarea: Nica, V., Ciobanu, Gh., Mustaţă Floare, Mărăcine Virginia, **"*Cercetări operaţionale I - Programare liniară, Probleme de optimizare în reţele de transport şi distribuţie, Teoria jocurilor strategice*",** Editura MATRIX ROM, Bucureşti 1998

4.1. Forma generală a unei probleme de programare liniară

Problemele de maxim şi de minim apar frecvent în cele mai diferite domenii ale matematicilor pure sau aplicate. În domeniul economic, asemenea probleme sunt foarte naturale. Astfel, firmele încearcă să maximizeze profiturile sau să minimizeze costurile. Experţii în planificare macroeconomică se preocupă de maximizarea bunăstării unei comunităţi economico-sociale. Consumatorii doresc să cheltuiască venitul lor într-un mod care să le maximizeze satisfacţia (de natură materială dar şi spirituală etc.)

Programarea liniară se ocupă de o clasă specială de probleme de optimizare care apar deseori în aplicaţiile economice. Aceste probleme constau în maximizarea sau minimizarea unei funcţii *liniare*, numită *funcţie obiectiv*, ale cărei variabile trebuie să satisfacă:

- un sistem de relaţii date sub forma unor ecuaţii şi/sau inecuaţii *liniare nestricte*, denumite generic *restricţii*;
- cerinţa de a lua numai valori numerice *nenegative* (≥ 0).

4.1.1. Exemple

1) *Problema firmei*. Considerăm un *sistem de producţie*, de exemplu o *firmă*, care produce n bunuri $G_1,G_2,...,G_n$ utilizând pentru aceasta m categorii de resurse $R_1,R_2,...,R_m$ (materii prime, forţă de muncă, capacităţi de producţie, combustibili şi energie etc.). Adoptăm ipoteza că tehnologia de transformare a resurselor în bunuri este *liniară* în sensul că:

- Pentru fiecare bun, consumul dintr-o anumită resursă este direct proporţional cu cantitatea produsă.
- Consumurile dintr-o resursă sau alta nu se condiţionează reciproc.

Fie atunci a_{ij} cantitatea din resursa i utilizată pentru producerea unei unităţi din bunul G_j. Fie deasemeni b_i cantitatea disponibilă din resursa R_i şi c_j preţul (sau profitul) unitar al bunului G_j.

- Preţul unui bun nu depinde de cantitatea produsă şi nici de situaţia vânzărilor celorlalte bunuri.

Problema constă în determinarea unui program de fabricaţie care să maximizeze venitul (sau profitul) firmei.

Să notăm cu x_j cantitatea din bunul G_j care urmează a fi produsă. Problema enunţată mai înainte devine:

Să se găsească valorile numerice $x_1,x_2,...,x_n$ care maximizează funcţia ***venit (sau profit) total***:

$$f = c_1x_1 + c_2x_2 + ... + c_nx_n$$

cu satisfacerea restricţiilor:

$$\begin{cases} a_{11}x_1 + a_{12}x_2 + \cdots + a_{1n}x_n \le b_1 \\ a_{21}x_1 + a_{22}x_2 + \cdots + a_{2n}x_n \le b_2 \\ \qquad \cdots\cdots\cdots\cdots\cdots \\ a_{m1}x_1 + a_{m2}x_2 + \cdots + a_{mn}x_n \le b_m \end{cases}$$

şi a condiţiilor de nenegativitate:

$$x_1 \ge 0\,,\ x_2 \ge 0\,,\ \cdots x_n \ge 0$$

Observaţie: Ipotezele de liniaritate făcute nu sunt verificate întotdeauna în practică. Raţiunea lor este dublă:

- conduc la modele matematice în general simple;
- pe baza modelelor liniare se pot formula concluzii calitative şi legităţi economice care îşi menţin valabilitatea - în anumite limite - şi într-un context neliniar.

2) ***Problema dietei*** a devenit o ilustrare clasică a programării liniare, fiind întâlnită în mai toate textele de specialitate. Ea se ocupă cu hrănirea unei colectivităţi, să zicem un grup de militari, în cel mai economic mod cu condiţia satisfacerii anumitor cerinţe de nutriţie. Mai concret, este vorba de a prepara un aliment complex pornind de la *n* sortimente de hrană $F_1,F_2,...,F_n$. Un număr de elemente sau principii nutritive $N_1,N_2,...,N_m$ - proteine, glucide, grăsimi calciu,etc. sunt avute în vedere în sensul că alimentul combinat trebuie să conţină cel puţin $b_1,b_2,...,b_m$ unităţi specifice din fiecare. Să presupunem cunoscute următoarele:

- cantitatea a_{ij} din principiul nutritiv N_i conţinută într-o unitate din tipul de hrană F_j;
- preţul unitar c_j al tipului de hrană F_j.

Notăm cu $x_1,x_2,...,x_n$ cantităţile din felurile de hrană $F_1,F_2,...,F_n$ care trebuie cumpărate în vederea elaborării dietei. Formal, $x_1,x_2,...,x_n$ vor trebui determinate astfel încât:

- ***costul total*** $f = c_1x_1 + c_2x_2 + ... + c_nx_n$ al alimentelor cumpărate să fie minim.
- amestecul să conţină principiile nutritive $N_1,N_2,...,N_m$ în cantităţi cel puţin egale cu $b_1,b_2,...,b_m$, adică:

$$\begin{cases} a_{11}x_1 + a_{12}x_2 + ... + a_{1n}x_n \ge b_1 \\ a_{21}x_1 + a_{22}x_2 + ... + a_{2n}x_n \ge b_2 \\ \qquad \cdots\cdots\cdots\cdots\cdots \\ a_{m1}x_1 + a_{m2}x_2 + ... + a_{mn}x_n \ge b_m \\ x_1 \ge 0, x_2 \ge 0, ..., x_n \ge 0 \end{cases}$$

Din nou au fost tacit utilizate ipotezele de liniaritate întâlnite şi în modelul precedent.

4.1.2 Soluţii admisibile ale unei probleme de programare liniară

Considerăm o problemă de programare liniară (P) cu *m* restricţii egalităţi şi/sau inegalităţi nestricte, *n* variabile şi cu funcţia obiectiv *f*. Un ansamblu de *n* valori numerice care satisfac restricţiile se va numi ***Soluţie*** a programului (P). Dacă în plus sunt verificate şi condiţiile de nenegativitate, ansamblul se numeşte ***Soluţie Admisibilă***. O soluţie admisibilă care maximizează sau minimizează - după caz - funcţia obiectiv se va numi ***Soluţie optimă***. Notând cu $\mathcal{A}$ mulţimea soluţiilor admisibile, problema (P) se scrie:

$$\textit{Să se determine } x^* \in \mathcal{A} \textit{ cu proprietatea: } f(x^*) = \max_{x \in \mathcal{A}} \textit{sau} \min f(x)$$

Este posibil ca (P) să aibă soluţii dar nici una din ele să fie admisibilă: $\mathcal{A} = \emptyset$. Spunem în acest caz că problema (P) este *incompatibilă.* Chiar dacă $\mathcal{A} \neq \emptyset$, este posibil ca funcţia obiectiv să fie nemărginită pe $\mathcal{A}$, adică să existe un şir de soluţii admisibile de-a lungul căruia funcţia obiectiv să tindă spre $+\infty$ sau $-\infty$, după caz. în această situaţie vom spune că (P) are *optim infinit.* Dacă (P) are (cel puţin) o soluţie optimă, zicem că (P) are *optim finit*.

Deoarece eventualele restricţii inegalităţi sunt *nestricte* mulţimea $\mathcal{A}$ este *închisă* (în topologia uzuală a spaţiului R^n), adică *o dată cu un şir convergent de puncte conţine şi limita acestuia*. Această proprietate este esenţială pentru existenţa unei soluţii optime a problemei (P)! Conform unui rezultat clasic al analizei matematice, dacă $\mathcal{A}$ este *mărginită*, atunci f îşi atinge efectiv extremele pe $\mathcal{A}$, şi deci (P) are *optim finit*. În consecinţă, dacă (P) are *optim infinit*, cu siguranţă $\mathcal{A}$ este *nemărginită*. Reciproca nu este în general adevărată: este posibil ca $\mathcal{A}$ să fie nemărginită şi totuşi (P) să aibă optim finit.

4.1.3 Forma canonică a unei probleme de programare liniară

O restricţie a unei probleme (P) de programare liniară se zice *concordantă* dacă este o *inegalitate* de tipul "≤" când funcţia obiectiv se maximizează şi de tipul "≥" când funcţia obiectiv se minimizează. O restricţie *inegalitate* care nu este concordantă se va numi *neconcordantă.* Restricţiile egalităţi nu fac obiectul acestei clasificări.

Spunem că o problemă de programare liniară este în *formă canonică* dacă toate restricţiile ei sunt *inegalităţi concordante*.

În consecinţă, o problemă în formă *canonică de maximizare* arată astfel:

$$\begin{cases} \sum_{j=1}^{n} a_{ij} x_j \le b_i \quad i = 1,\dots,m \\ x_j \ge 0 \quad j = 1,\dots,n \\ (\max) f = \sum_{j=1}^{n} c_j x_j \end{cases} \quad \text{sau matricial} \quad \begin{cases} Ax \le b \\ x \ge 0 \\ (\max) f = cx \end{cases} \quad \text{unde:}$$

$$A = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \cdots & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mn} \end{bmatrix} \quad b = \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_m \end{bmatrix} \quad x = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} \quad c = \begin{bmatrix} c_1 & c_2 & \cdots & c_n \end{bmatrix}$$

O problemă în formă *canonică de minimizare* se va scrie:

$$\begin{cases} \sum_{j=1}^{n} a_{ij} x_j \ge b_i \\ x_j \ge 0 \\ (\min) f = \sum_{j=1}^{n} c_j x_j \end{cases} \quad \Leftrightarrow \quad \begin{cases} Ax \ge b \\ x \ge 0 \\ (\min) f = cx \end{cases}$$

De exemplu, problema firmei (1.1, exemplul 1)) este o formă canonică de maximizare în timp ce problema dietei (1.1, exemplul 2)) este o formă canonică de minimizare.

Orice problemă de programare liniară se poate pune sub o formă canonică de maximizare sau minimizare, fără modificarea mulţimii soluţiilor admisibile, observând că:

- *o egalitate se poate înlocui cu două inegalităţi de sens contrar;*
- *o restricţie neconcordantă devine concordantă prin înmulţire cu -1;*
- *putem schimba sensul optimizării funcţiei obiectiv, graţie formulei generale:*

$$\min_{x\in\mathcal{A}} f(x) = -\max_{x\in\mathcal{A}}\left[-f(x)\right] \qquad (1.3.1)$$

În consecinţă, putem face anumite raţionamente teoretice pe o formă canonică (ca de exemplu în teoria dualităţii liniare), fără ca prin aceasta să restrângem generalitatea.

Exemplul 1.3.1

$$\begin{cases} (\max) f = 2x_1 - 3x_2 + 4x_3 \\ x_1 - 3x_2 + 5x_3 = 3 \\ 3x_1 + x_2 \geq 5 \\ 2x_1 + x_3 \leq 10 \\ x_1 \geq 0, x_2 \geq 0, x_3 \geq 0 \end{cases} \Rightarrow \begin{cases} (\min)(-f) = -2x_1 + 3x_2 - 4x_3 \\ x_1 - 3x_2 + 5x_3 \geq 3 \\ -x_1 + 3x_2 - 5x_3 \geq -3 \\ 3x_1 + x_2 \geq 5 \\ -2x_1 - x_3 \geq -10 \\ x_1 \geq 0, x_2 \geq 0, x_3 \geq 0 \end{cases}$$

Programul (P) *Forma canonică de minimizare a programului (P)*

4.1.4 Forma standard a unei probleme de programare liniară

Spunem că o problemă de programare liniară este în *formă standard* dacă toate restricţiile ei sunt *egalităţi*. Importanţa acestei forme particulare rezultă din faptul că metoda de rezolvare a problemelor de programare liniară care va fi expusă mai departe cere ca problema să fie în această prezentare.

În consecinţă, o problemă (P) care are şi restricţii inegalităţi va fi înlocuită - în vederea rezolvării ei - cu o alta în care toate restricţiile sunt egalităţi. Noua problemă, numită *forma standard a problemei* (P) şi notată (FSP), se construieşte astfel:

- *O restricţie inegalitate din problema originală (P) de tipul "≤" (respectiv de tipul "≥") se transformă în egalitate prin adăugarea (respectiv prin scăderea) unei variabile nenegative din membrul său stâng. Aceste variabile se numesc* ***variabile de abatere*** sau ***de ecart****.*
- *Restricţiile egalităţi nu se modifică.*
- *Noile variabile introduse nu apar în funcţia obiectiv a problemei originale (alternativ, spunem că ele apar cu coeficienţi nuli)*

Exemplul 1.4.1

$$(P)\begin{cases} (\max) f = 7x_1 + 9x_2 + 8x_3 \\ 5x_1 + 2x_2 - x_3 \geq 4 \\ 3x_1 + x_2 + x_3 = 5 \\ x_1 + 2x_2 + 3x_3 \leq 9 \\ x_1 \geq 0, x_2 \geq 0, x_3 \geq 0 \end{cases} \Rightarrow (FSP)\begin{cases} (\max) f = 7x_1 + 9x_2 + 8x_3 \\ 5x_1 + 2x_2 - x_3 - x_4 = 4 \\ 3x_1 + x_2 + x_3 = 5 \\ x_1 + 2x_2 + 3x_3 + x_5 = 9 \\ x_j \geq 0,\ j = 1,\ldots,5 \end{cases}$$

Problema care apare în acest context este aceea de a explica modul în care se obţine soluţia optimă a problemei (P) dacă se cunoaşte soluţia optimă a formei sale standard (FSP). Se poate arăta uşor că între mulţimile de soluţii admisibile $\mathcal{A}_P$, ale problemei (P) şi $\mathcal{A}_{FSP}$, ale problemei (FSP), există

o *corespondenţă bijectivă care conservă soluţiile optime.* Vom arăta cum funcţionează această corespondenţă pe exemplul precedent.

Notând-o cu Φ, aceasta corespondenţă va asocia unei soluţii admisibile $\bar{x} = (\bar{x}_1, \bar{x}_2, \bar{x}_3)$ a problemei (P) vectorul:

$$\Phi(\bar{x}) = (\bar{x}_1, \bar{x}_2, \bar{x}_3, 5\bar{x}_1 + 2\bar{x}_2 - \bar{x}_3 - 4, 9 - \bar{x}_1 - 2\bar{x}_2 - 3\bar{x}_3)$$

care prin construcţie se dovedeşte a fi o soluţie admisibilă a problemei (FSP). Reciproc, unei soluţii admisibile $\tilde{x} = (\tilde{x}_1, \tilde{x}_2, \tilde{x}_3, \tilde{x}_4, \tilde{x}_5)$ a problemei (FSP) corespondenţa inversă Φ^{-1} îi asociază vectorul $(\tilde{x}_1, \tilde{x}_2, \tilde{x}_3)$ care satisface în mod clar restricţiile problemei originale (P). Dacă $\bar{x}$ este soluţia optimă a problemei (P) atunci $\Phi(\bar{x})$ este soluţia optimă a problemei (FSP) şi reciproc, dacă cunoaştem soluţia optimă $\tilde{x}$ a problemei (FSP) , $\Phi^{-1}(\tilde{x})$ reprezintă soluţia optimă a problemei (P).

În problemele concrete, variabilele de abatere au interpretări economice precise aşa că în analiza soluţiei optime valorile lor vor fi luate în considerare laolaltă cu valorile variabilelor originale. Astfel, în problema firmei (1.1, exemplul1)) variabilele de abatere x_{n+1}, x_{n+2}, ..., x_{n+m} definite prin:

$$x_{n+i} = b_i - \sum_{j=1}^{n} a_{ij} x_j \quad i = 1, \ldots, m$$

reprezintă ***cantităţi de resurse neconsumate*** şi prin urmare cunoaşterea valorilor lor în soluţia optimă oferă indicaţii utile în analiza modului în care sunt utilizate resursele firmei: materii prime, capacităţi de producţie, forţă de muncă, etc.

În problema dietei (1.1, exemplul 2)) variabilele de abatere:

$$x_{n+i} = \sum_{j=1}^{n} a_{ij} x_j - b_i \quad i = 1, \ldots, m$$

reprezintă *cantităţile de principii nutritive cu care sunt depăşite* nivelele minimale specificate în reţetă.

4.1.5 Rezolvarea grafică a problemelor de programare liniară

Să considerăm problema:

$$\begin{cases} (\max) f = 3x_1 + 4x_2 \\ -3x_1 + 4x_2 \le 12 \\ x_1 + x_2 \le 6 \\ -2x_1 + x_2 \le 2 \\ x_1 \ge 0 \, , \, x_2 \ge 0 \end{cases}$$

Identificăm x_1, x_2 cu abscisa, respectiv ordonata unui punct din planul raportat la un sistem ortogonal de axe. Este cunoscut faptul că mulţimea punctelor din plan ale căror coordonate satisfac prima restricţie coincide cu unul din semiplanele determinate de dreapta d_1 de ecuaţie $-3x_1+4x_2 = 12$. Mai precis, este vorba de semiplanul care conţine originea (0,0), deoarece coordonatele acesteia satisfac evident prima restricţie. În mod analog, următoarele restricţii sunt verificate în semiplanele determinate de dreapta d_2 de ecuaţie $x_1+x_2 = 6$ şi respectiv d_3 de ecuaţie $-2x_1+x_2 = 2$ şi care conţin originea. În fine, condiţia $x_1 \ge 0$ are loc în semiplanul "din dreapta" axei verticale, în timp ce condiţia $x_2 \ge 0$ are loc "deasupra" axei orizontale.

Soluţiile admisibile ale problemei se identifică cu punctele comune celor cinci semiplane. Acestea formează interiorul şi frontiera poligonului OABCD din figura 1.5.1.

Funcţia obiectiv determină – pentru *f* variabil – o mulţime de drepte paralele care intersectează sau nu mulţimea $\mathcal{A}$. Astfel punctele situate pe dreapta $3x_1+4x_2=12$ reprezintă diferite combinaţii ale

mărimilor x_1, x_2 care dau funcţiei obiectiv f aceeaşi valoare 12. Întrucât această dreaptă taie $\mathcal{A}$, rezultă că problema are soluţii admisibile - chiar o infinitate - care oferă funcţiei obiectiv valoarea 12. Dreapta $3x_1+4x_2=24$ nu mai taie $\mathcal{A}$ şi deci nici o soluţie admisibilă a problemei nu este capabilă să asigure funcţiei obiectiv valoarea 24. Conchidem că maximul funcţiei f este "undeva" între 12 şi 24. Se observă uşor că acest maxim se atinge în vârful C al frontierei lui $\mathcal{A}$. Punctul C este intersecţia dreptelor d_1 şi d_2 şi deci coordonatele sale, care reprezintă soluţia optimă a problemei, se determină rezolvând sistemul format din ecuaţiile celor două drepte. Se găseşte în acest fel $x_1^* = \frac{12}{7}$, $x_2^* = \frac{30}{7}$ maximul lui f fiind $22\frac{2}{7}$. Soluţia optimă satisface cu egalitate primele două restricţii şi cu inegalitate strictă pe ceea de a treia.

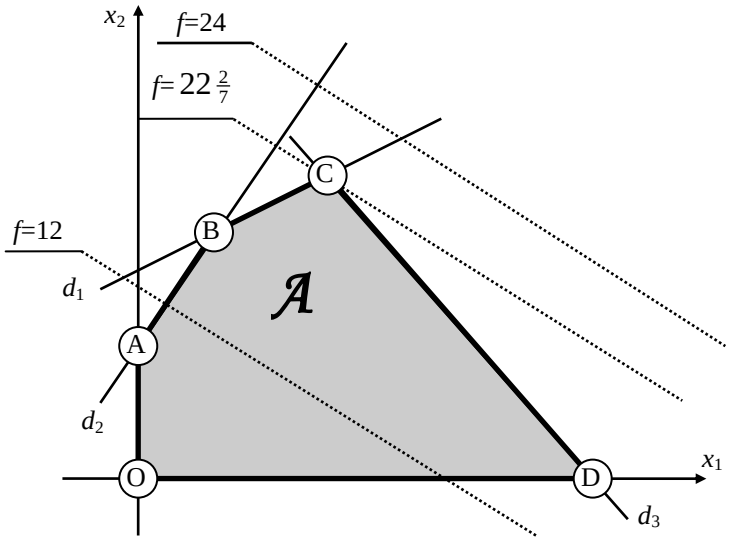


Figura 1.5.1

În mod asemănător se arată că dacă funcţia de maximizat ar fi fost $f = -x_1+x_2$ atunci optimul ar fi fost atins în vârful B de coordonate $x_1=4/5$, $x_2=18/5$.

Examinând acest exemplu putem trage următoarele **concluzii**:

*1. Mulţimea $\mathcal{A}$ este **convexă**, adică o dată cu două puncte conţine şi segmentul care le uneşte. O consecinţă intuitivă a acestei proprietăţi este că soluţia optimă, dacă există, se găseşte "undeva" pe frontiera lui $\mathcal{A}$.*

*2. Frontiera lui $\mathcal{A}$ este un contur poligonal cu un **număr finit de vârfuri** şi o soluţie optimă **se găseşte neapărat într-unul din ele**.*

Aceste concluzii, care se confirmă pe orice altă problemă în două sau trei variabile (mulţimea soluţiilor admisibile putând fi "vizualizată" în planul R^2 sau spaţiul R^3) au constituit sursa întregii teorii a programării liniare.

4.2. Dualitatea în programarea liniară

În principiu, oricărei probleme de programare liniară i se asociază o alta, numită ***duala*** sa şi în esenţă *teoria dualităţii* constă în studiul relaţiilor dintre cele două probleme. Firește, construcţia problemei duale depinde nemijlocit de structura problemei iniţiale denumită şi problema ***primală***. Întotdeauna sensul optimizării în cele două probleme este diferit: dacă în primală funcţia obiectiv se maximizează (minimizează) în dual funcţia obiectiv se minimizează (maximizează). Studiul şi interpretarea economică a problemei duale aduc informaţii suplimentare în analiza proceselor economice şi în fundamentarea deciziilor.

4.2.1 Reguli de construire a problemei duale

Pentru a conferi construcţiei problemei duale maxima generalitate vom slăbi condiţia de nenegativitate impusă tuturor variabilelor, admiţând că unele din ele nu pot lua decât valori nepozitive (≤ 0) în timp ce altele pot lua orice valoare reală.

Cu această observaţie duala unei probleme de programare liniară cu *m* restricţii şi *n* variabile se construieşte după următoarele reguli:

1) Dacă în primală funcţia obiectiv se maximizează (respectiv se minimizează) în problema dual funcţia obiectiv se minimizează (respectiv se maximizează).

2) Restricţiei de rang i , i=1,...,m din primală îi corespunde în dual o variabilă u_i; dacă restricţia primală este o inegalitate concordantă (respectiv neconcordantă, respectiv o egalitate) variabila dual asociată este nenegativă (≥ 0), (respectiv nepozitivă (≤ 0), respectiv fără restricţie de semn).

3) Variabilei x_j , j=1,...,n din problema primală îi corespunde în dual restricţia de rang j. Membrul stâng al acestei restricţii este o combinaţie liniară a variabilelor duale u_i realizată cu coeficienţii variabilei x_j din toate restricţiile primalei (aceştia sunt a_{ij}, i=1,...,m). Termenul său liber este coeficientul c_j al lui x_j din funcţia obiectiv primală. În fine, dacă variabila primală x_j este nenegativă (respectiv nepozitivă, respectiv fără restricţie de semn) restricţia dual asociată va fi o inegalitate concordantă (respectiv neconcordantă, respectiv o egalitate).

4) Coeficienţii funcţiei obiectiv ai problemei duale sunt termenii liberi b_i ai restricţiilor problemei primale.

Exemplul 2.1.1

Problema primală		*Problema dual*
$3x_1 - 2x_2 + x_3 + 4x_4 - x_5 \leq 6$	$\longleftrightarrow$	$u_1 \geq 0$
$2x_1 + 2x_3 + 5x_4 + x_5 = 9$	$\longleftrightarrow$	u_2 f.r.s.
$x_1 + 6x_2 + x_3 + 2x_4 \geq 3$	$\longleftrightarrow$	$u_3 \leq 0$
$x_1 \geq 0$	$\longleftrightarrow$	$3u_1 + 2u_2 + u_3 \geq 8$
$x_2 \geq 0$	$\longleftrightarrow$	$-2u_1 + 6u_3 \geq 3$
x_3 f.r.s.	$\longleftrightarrow$	$u_1 + 2u_2 + u_3 = 1$
x_4 f.r.s.	$\longleftrightarrow$	$4u_1 + 5u_2 + 2u_3 = 5$
$x_5 \leq 0$	$\longleftrightarrow$	$-u_1 + u_2 \leq 7$
$\max f = 8x_1 + 3x_2 + x_3 + 5x_4 + 7x_5$		$\min g(u) = 6u_1 + 9u_2 + 3u_3$

Observaţii: 1) Problema dual are atâtea variabile (respectiv restricţii) câte restricţii (respectiv variabile) are problema primală.

2) Regulile 1) - 4) pun în evidenţă următoarele corespondenţe de termeni prin trecere la dală:

min	↔	max
(restricţie) concordantă	↔	(variabilă) nenegativă
(restricţie) neconcordantă	↔	(variabilă) nepozitivă
(restricţie) egalitate	↔	(variabilă) fără restricţie de semn
termen liber al unei restricţii	↔	coeficient al funcţiei obiectiv

3) Din construcţia de mai sus rezultă următoarea concluzie:

Duala dualei este problema primală.

Spunem că dualitatea *în programarea liniară* are un caracter *involutiv*.

În consecinţă, fiind dată o problemă de programare liniară (P) şi duala sa (Q), vom vorbi despre *cuplul de probleme în dualitate* (P,Q), fără a mai specifica în mod expres care din probleme este primala şi care duala.

4.2.2. Dualele unor forme particulare de probleme de programare liniară

1) *Duala unei forme canonice de maximizare este o formă canonică de minimizare şi reciproc.*

$$\left\{\begin{array}{ll} a_{11}x_1 + a_{12}x_2 + \ldots + a_{1n}x_n \le b_1 & u_1 \ge 0 \\ a_{21}x_1 + a_{22}x_2 + \ldots + a_{2n}x_n \le b_2 & u_2 \ge 0 \\ \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots & \ldots\ldots\ldots \\ a_{m1}x_1 + a_{m2}x_2 + \ldots + a_{mn}x_n \le b_m & u_m \ge 0 \\ x_1 \ge 0 & a_{11}u_1 + a_{21}u_2 + \ldots + a_{m1}u_m \ge c_1 \\ x_2 \ge 0 & a_{12}u_1 + a_{22}u_2 + \ldots + a_{m2}u_m \ge c_2 \\ \vdots & \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \\ x_n \ge 0 & a_{1n}u_1 + a_{2n}u_2 + \ldots + a_{mn}u_m \ge c_n \\ \max f = c_1x_1 + c_2x_2 + \ldots + c_nx_n & \min g = b_1u_1 + b_2u_2 + \ldots + b_mu_m \end{array}\right\}$$

Cu notaţiile matriciale introduse în (1.3) la care se adaugă: $u = [u_1, u_2, \ldots, u_m]$, cuplul format din cele două probleme de mai sus devine:

$$(P)\begin{cases} Ax \le b \\ x \ge 0 \\ \max f(x) = cx \end{cases} \qquad (Q)\begin{cases} uA \ge c \\ u \ge 0 \\ \min g(u) = ub \end{cases}$$

2) Conservarea formei de prezentare se pierde atunci când se consideră duala unei probleme în formă standard.

$$\left\{\begin{array}{lll} \sum_{j=1}^{n} a_{ij}x_j = b_i \quad i = 1,\ldots,m & \leftrightarrow & u_i \ \ f.r.s. \quad i = 1,\ldots,m \\ x_j \ge 0 \quad j = 1,\ldots,n & \leftrightarrow & \sum_{i=1}^{m} a_{ij}u_i \ge c_j \quad j = 1,..,n \\ \max f = \sum_{j=1}^{n} c_j x_j & & \min g = \sum_{i=1}^{m} b_i u_i \end{array}\right\}$$

sau matricial:

$$(P)\begin{cases} Ax = b \\ x \geq 0 \\ \max f(x) = cx \end{cases} \qquad (Q)\begin{cases} uA \geq c \\ u \in R^m\ (f.r.s.) \\ \min g(u) = ub \end{cases}$$

(f.r.s.≡ fără restricţie de semn!)

În mod analog se construieşte duala formei standard în care funcţia obiectiv se minimizează:

$$(P)\begin{cases} Ax = b \\ x \geq 0 \\ \min f(x) = cx \end{cases} \qquad (Q)\begin{cases} uA \leq c \\ u \quad f.r.s. \\ \max g(u) = ub \end{cases}$$

Observaţie: De reţinut este faptul că *dualele a două probleme de programare liniară echivalente sunt ele însele echivalente* adică au aceeaşi mulţime de soluţii admisibile şi aceleaşi soluţii optime.

4.2.3. Teoreme de dualitate

Cu notaţiile matriciale din (2.2) să considerăm cuplul de probleme în dualitate în formă canonică:

$$(P)\begin{cases} (\max) f(x) = cx \\ Ax \leq b \\ x \geq 0 \end{cases} \qquad (Q)\begin{cases} (\min) g(u) = ub \\ uA \geq c \\ u \geq 0 \end{cases}$$

Teorema 2.3.1 *Fie* $\bar{x} = \left[\bar{x}_1, \bar{x}_2, \ldots, \bar{x}_n\right]^T$ *o soluţie admisibilă a problemei* (P) *şi* $\bar{u} = \left[\bar{u}_1, \bar{u}_2, \ldots, \bar{u}_m\right]$ *o soluţie admisibilă a problemei* (Q). *Atunci:*

1) $f(\bar{x}) \leq g(\bar{u})$

2) *Dacă* $f(\bar{x}) = g(\bar{u})$ *atunci* $\bar{x}$ *este o soluţie optimă a problemei* (P) *iar* $\bar{u}$ *este o soluţie optimă a problemei* (Q).

Demonstraţie: 1) Prin ipoteză $A\bar{x} \leq b$, $\bar{x} \geq 0$ şi $\bar{u}A \geq c$, $\bar{u} \geq 0$. Deducem că $\bar{u}A\bar{x} \leq \bar{u}b$ şi $\bar{u}A\bar{x} \geq c\bar{x}$ (nenegativitatea vectorilor $\bar{x}$ şi $\bar{u}$ este esenţială!) de unde:

$$f(\bar{x}) = c\bar{x} \leq \bar{u}A\bar{x} \leq \bar{u}b = g(\bar{u})$$

2) Dacă $f(\bar{x}) = g(\bar{u})$ şi dacă $\bar{x}$ nu ar fi soluţie optimă a problemei (P) ar exista o soluţie admisibilă $\bar{x}'$ mai bună, adică $f(\bar{x}') > f(\bar{x})$. Rezultă inegalitatea $f(\bar{x}') > g(\bar{u})$ contrară celor demonstrate la punctul precedent. ■

Clasificarea cuplurilor de probleme de programare liniară în dualitate este făcută de următoarea teoremă:

Teorema 2.3.2 (**Teorema fundamentală a dualităţii**) *Pentru un cuplu de probleme în dualitate una şi numai una din următoarele situaţii este posibilă:*

1) Ambele probleme au soluţii admisibile; atunci ambele au soluţii optime şi valorile optime ale funcţiilor obiectiv coincid: $f(\bar{x}) = g(\bar{u})$.

2) Numai una din probleme are soluţii admisibile, iar cealaltă nu are; atunci problema compatibilă are optim infinit.

3) Nici una din probleme nu are soluţii admisibile.

Simetria teoremei 2.3.2 nu constituie totuşi un răspuns pentru reciproca teoremei 2.3.1 Specificăm acest lucru în mod expres pentru că în programarea neliniară el nu are loc.

Teorema 2.3.3 *Dacă una din problemele unui cuplu de probleme în dualitate are soluţie optimă atunci şi cealaltă are şi valorile optime ale funcţiilor obiectiv coincid.*

Nu dăm demonstraţia teoremei 2.3.2 deoarece pregătirile necesare depăşesc cadrul impus acestui curs. Vom demonstra însă teorema 2.3.3 după ce vom prezenta metoda generală de rezolvare a problemelor de programare liniară.

Teorema 2.3.4 (Teorema ecarturilor complementare - TEC) *Fie (P,Q) un cuplu de probleme canonice în dualitate:*

$$(P)\begin{cases} Ax \le b \\ x \ge 0 \\ \max\ f(x) = cx \end{cases} \qquad (Q)\begin{cases} uA \ge c \\ u \ge 0 \\ \min\ g(u) = ub \end{cases}$$

Un cuplu de soluţii admisibile $(\bar{x}, \bar{u})$ *este un cuplu de soluţii optime dacă şi numai dacă:*

$$\begin{cases} (\bar{u}A - c)\bar{x} = 0 \\ \bar{u}(b - A\bar{x}) = 0 \end{cases} \qquad (2.3.1)$$

Demonstraţie: Să presupunem relaţiile (2.3.1) verificate de cuplul $(\bar{x}, \bar{u})$:

$\left.\begin{matrix} \bar{u}A\bar{x} - c\bar{x} = 0 \\ \bar{u}b - \bar{u}A\bar{x} = 0 \end{matrix}\right\} \Rightarrow c\bar{x} = \bar{u}b \Rightarrow (\bar{x}, \bar{u})$ este un cuplu de soluţii optime în virtutea teoremei 2.3.1.Reciproc, să presupunem că $(\bar{x}, \bar{u})$ constituie un cuplu de soluţii optime. Atunci $c\bar{x} = \bar{u}b$,în virtutea teoremei fundamentale a dualităţii şi prin urmare:

$$(\bar{u}A - c)\bar{x} + \bar{u}(b - A\bar{x}) = 0$$

Deoarece $\bar{x}, \bar{u}$ sunt soluţii admisibile avem: $(\bar{u}A - c)\bar{x} \ge 0,\ \bar{u}(b - A\bar{x}) \ge 0$ şi deci $(\bar{u}A - c)\bar{x} = 0,\ \bar{u}(b - A\bar{x}) = 0$ relaţii care arată că $\bar{x}, \bar{u}$ verifică (2.3.1).■

În notaţiile secţiunii (2.2) relaţiile matriciale (2.3.1) se pot scrie:

$$\sum_{j=1}^{n}\left(\sum_{i=1}^{m} a_{ij}u_i - c_j\right)x_j = 0$$

$$\sum_{i=1}^{m} u_i\left(b_i - \sum_{j=1}^{n} a_{ij}x_j\right) = 0$$

Aceste relaţii sunt echivalente cu:

$$\left(\sum_{i=1}^{m} a_{ij}u_i - c_j\right)x_j = 0 \qquad j = 1,\ldots,n$$

$$u_i\left(b_i - \sum_{j=1}^{n} a_{ij}x_j\right) = 0 \qquad i = 1,\ldots,m$$

Verificarea acestor egalităţi de către un cuplu de soluţii admisibile $\bar{x} = \left[\bar{x}_1, \bar{x}_2, \ldots, \bar{x}_n\right]^T$ şi $\bar{u} = \left[\bar{u}_1, \bar{u}_2, \ldots, \bar{u}_m\right]$ reprezintă deci *o condiţie necesară şi suficientă de optimalitate*.

Pe lângă formularea precedentă, teorema 2.3.4. mai are următoarea interpretare:

Cuplul $(\bar{x}, \bar{u})$ *de soluţii admisibile este un cuplu de soluţii optime dacă şi numai dacă verifică seturile de implicaţii:*

$$\begin{cases} \bar{x}_j > 0 \quad \Rightarrow \quad \sum_{i=1}^{m} a_{ij}\bar{u}_i = c_j \\ \sum_{j=1}^{n} a_{ij}\bar{u}_i < b_i \quad \Rightarrow \quad \bar{u}_i = 0 \end{cases} \qquad \begin{cases} \bar{u}_i > 0 \quad \Rightarrow \quad \sum_{j=1}^{n} a_{ij}\bar{x}_j = b_i \\ \sum_{i=1}^{m} a_{ij}\bar{u}_i > c_j \quad \Rightarrow \quad \bar{x}_j = 0 \end{cases}$$

Observaţie. Deşi prezentată pe un cuplu de probleme canonice, teorema 2.3.4. este valabilă pentru orice cuplu de probleme în dualitate. Astfel pentru cuplul:

$$\begin{cases} \max f = cx \\ Ax = b \\ x \geq 0 \end{cases} \qquad \begin{cases} \min g = ub \\ uA \geq c \\ u \ f.r.s. \end{cases}$$

ele se reduc la: $(uA - c)x = 0 \quad \Leftrightarrow \quad \left(\sum_{i=1}^{m} a_{ij}u_i - c_j\right)u_j = 0 \quad j = 1,\ldots,n.$

Şi mai concret, considerăm cuplul din:

Exemplul 2.3.1

$$(P)\begin{cases} \min f = 5x_1 - 2x_2 - x_3 \\ 6x_1 + 2x_2 - 3x_3 \leq 6 \\ x_1 + 3x_2 + 2x_3 = 12 \\ 2x_1 - x_2 + 4x_3 \geq 4 \\ x_1, x_2, x_3 \geq 0 \end{cases} \qquad (Q)\begin{cases} \max g = 6u_1 + 12u_2 + 4u_3 \\ 6u_1 + u_2 + 2u_3 \leq 5 \\ 2u_1 + 3u_2 - u_3 \leq -2 \\ -3u_1 + 2u_2 + 4u_3 \leq -1 \\ u_1 \leq 0, u_2 \ f.r.s., u_3 \geq 0 \end{cases}$$

Conform TEC, condiţia necesară şi suficientă pentru ca două *soluţii admisibile* ale celor două probleme:

$$\bar{x} = (\bar{x}_1, \bar{x}_2, \bar{x}_3)^T \text{ şi } \bar{u} = (\bar{u}_1, \bar{u}_2, \bar{u}_3)$$

să fie optime este satisfacerea relaţiilor:

$$(5 - 6u_1 - u_2 - 2u_3)x_1 = 0 \qquad (1)$$
$$(-2 - 2u_1 - 3u_2 + u_3)x_2 = 0 \qquad (2)$$
$$(-1 + 3u_1 - 2u_2 - 4u_3)x_3 = 0 \qquad (3)$$
$$(6 - 6x_1 - 2x_2 + 3x_3)u_1 = 0 \qquad (4)$$
$$(2x_1 - x_2 + 4x_3 - 4)u_3 = 0 \qquad (5)$$

O consecinţă importantă a TEC este faptul că rezolvarea unei probleme de programare liniară este echivalentă cu rezolvarea dualei sale în sensul că dacă se cunoaşte soluţia optimă a uneia din ele putem deduce relativ simplu soluţia optimă a celeilalte. Pentru ilustrare vom determina soluţia optimă a problemei (P) din exemplul precedent ştiind că duala (Q) are soluţia optimă u^* de componente: $u_1^* = 0$, $u_2^* = -\frac{9}{14}$, $u_3^* = \frac{1}{14}$.

Înlocuind u^* în (1) - (5) se constată că relaţiile (2),(3),(4) sunt satisfăcute de orice valori numerice acordate variabilelor x_1, x_2, x_3. În schimb din (1) şi (5) obţinem relaţiile: x_1=0 şi $2x_1 - x_2 + 4x_3 - 4 = 0$ care împreună cu restricţia egalitate din (P) constituie un sistem liniar:

$$\begin{cases} x_1 = 0 \\ 2x_1 - x_2 + 4x_3 = 4 \\ x_1 + 3x_2 + 2x_3 = 12 \end{cases}$$

Rezolvând sistemul, obţinem soluţia optimă a problemei (P):

$$x_1^* = 0 \quad , \quad x_2^* = \frac{20}{7} \quad , \quad x_3^* = \frac{12}{7} \qquad f_{\max} = f(x^*) = -\frac{52}{7} = g(u^*) = g_{\min}$$

Echivalenţa amintită mai înainte este folositoare în rezolvarea efectivă a problemelor de programare liniară ştiut fiind că efortul de calcul este relativ mai mic dacă numărul restricţiilor este

mai mic. În consecinţă, pentru o problemă cu multe restricţii şi un număr restrâns de variabile, va fi mai comod să rezolvăm duala sa.

4.2.4. Interpretarea economică a dualităţii

Reluăm problema firmei din secţiunea 1.1 exemplul 1). Considerăm o întreprindere care produce bunurile $G_1,G_2,...,G_m$ la preţurile $c_1,c_2,...,c_n$ folosind pentru aceasta mai multe resurse $R_1,R_2,...,R_m$ disponibile în cantităţile limitate $b_1,b_2,...,b_m$. Consumurile specifice de resurse pentru unul sau altul din bunurile realizabile de către firmă sunt specificate în matricea:

$$A=\begin{bmatrix} a_{11} & a_{12} & \dots & a_{1n} \\ a_{21} & a_{22} & \dots & a_{2n} \\ \vdots & \vdots & \dots & \vdots \\ a_{m1} & a_{m2} & \dots & a_{mn} \end{bmatrix}$$

Obiectivul firmei este maximizarea veniturilor rezultate din vânzarea bunurilor produse. În ipotezele de liniaritate uzuale, programul liniar pentru determinarea combinaţiei optime de bunuri este:

$$(P)\begin{cases} (\max) f(x) = c_1x_1 + c_2x_2 + \dots + c_nx_n \\ a_{11}x_1 + a_{12}x_2 + \dots + a_{1n}x_n \le b_1 \\ a_{21}x_1 + a_{22}x_2 + \dots + a_{2n}x_n \le b_2 \\ \dots\dots\dots\dots\dots\dots\dots\dots\dots\dots \\ a_{m1}x_1 + a_{m2}x_2 + \dots + a_{mn}x_n \le b_m \\ x_1 \ge 0, x_2 \ge 0, \dots, x_n \ge 0 \end{cases}$$

Să notăm cu $x^* = (x_1^*, x_2^*, \dots, x_n^*)$ combinaţia de bunuri care aduce firmei venitul maxim:

$$V^* = (\max) f = f(x^*) = c_1x_1^* + c_2x_2^* + \dots + c_nx_n^*$$

Dacă admitem că singura posibilitate a firmei de a obţine venitul V^* constă în transformarea resurselor disponibile în bunuri conform programului x^* şi vânzarea acestora este natural să ne întrebăm *care este contribuţia fiecărei resurse în parte la formarea acestui venit.*

La baza discuţiei vom pune aceleaşi ipoteze de liniaritate care ne-au condus la modelul matematic (P) şi anume:

- contribuţia unei resurse la venitul maxim al firmei este - *până la o anumită limită* - direct proporţională cu cantitatea disponibilă;
- nivelul contribuţiei unei resurse nu este condiţionat de celelalte resurse, ci numai de cantitatea în care ea se află disponibilă.

Să notăm atunci cu $u_1^*, u_2^*, \dots, u_m^*$ ***aportul*** unei unităţi din resursa R_1, din resursa R_2, ş.a.m.d. la formarea venitului maxim V^*. În virtutea ipotezelor de liniaritate amintite putem scrie:

$$V^* = b_1u_1^* + b_2u_2^* + \dots + b_mu_m^* \qquad (2.4.1)$$

Producerea unei unităţi din bunul G_j necesită a_{1j} unităţi din resursa R_1, a_{2j} unităţi din resursa R_2, ..., a_{mj} unităţi din resursa R_m.

Partea din venitul maxim V^* corespunzătoare acestor cantităţi este:

$$a_{1j}u_1^* + a_{2j}u_2^* + \dots + a_{mj}u_m^*$$

Această mărime trebuie să acopere preţul pe care firma îl încasează prin vânzare de vreme ce ea nu are alte resurse de formare a venitului în afara transformării resurselor în bunuri. Se conchide că sistemul $u^* = (u_1^*, u_2^*, \dots, u_m^*)$ trebuie să satisfacă relaţiile:

$$\begin{cases} a_{11}u_1^* + a_{21}u_2^* + \ldots + a_{m1}u_m^* \geq c_1 \\ a_{12}u_1^* + a_{22}u_2^* + \ldots + a_{m2}u_m^* \geq c_2 \\ \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots \\ a_{1n}u_1^* + a_{2n}u_2^* + \ldots + a_{mn}u_m^* \geq c_n \end{cases} \qquad (2.4.2)$$

şi de asemenea condiţiile de nenegativitate:

$$u_1^* \geq 0, u_2^* \geq 0, \ldots, u_m^* \geq 0 \qquad (2.4.3)$$

Sunt suficiente relaţiile (2.4.1) - (2.4.3) pentru determinarea efectivă a valorilor $u_1^*, u_2^*, \ldots, u_m^*$? Teoria dualităţii dă un răspuns afirmativ pe baza următoarelor observaţii:

1) Relaţiile (2.4.2) şi (2.4.3) arată că $u^* = (u_1^*, u_2^*, \ldots, u_m^*)$ constituie o soluţie admisibilă a dualei (Q) asociată problemei (P):

$$(Q)\begin{cases} (\min) g(x) = b_1u_1 + b_2u_2 + \ldots + b_mu_m \\ a_{11}u_1 + a_{21}u_2 + \ldots + a_{m1}u_m \geq c_1 \\ a_{12}u_1 + a_{22}u_2 + \ldots + a_{m2}u_m \geq c_2 \\ \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \\ a_{1n}u_1 + a_{2n}u_n + \ldots + a_{mn}u_m \geq c_n \\ u_1 \geq 0, u_2 \geq 0, \ldots, u_m \geq 0 \end{cases}$$

2) Relaţia (2.4.1) rescrisă în forma $f(x^*) = g(u^*)$, coroborată cu teorema de dualitate 2.3.1, ne arată că u^* este chiar soluţia optimă a problemei duale Q.

Astfel, am găsit un conţinut economic coerent variabilelor duale $u_1, u_2, \ldots, u_m$ din problema duală (Q). Ele reprezintă nişte *valori băneşti asociate la câte o unitate din fiecare resursă disponibilă* a firmei şi exprimă ***aportul unitar*** al fiecărei resurse la formarea ***venitului maxim***. Strict dimensional, variabilele $u_1, u_2, \ldots, u_m$ au semnificaţia unor *preţuri* ataşate resurselor. Totuşi aceste preţuri nu au nimic comun cu valoarea intrinsecă a resurselor ci - aşa cum a rezultat din interpretarea dată - reflectă numai măsura participării lor la formarea venitului maxim. Tocmai pentru a preveni identificarea lor cu preţurile reale ale resurselor, în literatura de specialitate, aceste entităţi au fost denumite ***preţuri umbră*** (*shadow prices*). A rezultat de asemenea şi o modalitate de calcul a acestor mărimi: *se rezolvă problema duală (Q).*

Teoria dualităţii arată că venitul maxim V^* al firmei - privit ca funcţie de cantităţile disponibile de resurse - depinde liniar de aceste disponibile prin intermediul preţurilor duale optime - vezi relaţia (2.4.1)

În consecinţă :

$$\frac{\partial V^*}{\partial b_i} = u_i^* \qquad i = 1, \ldots, m \qquad (2.4.4)$$

Prin urmare preţurile duale optime ne arată cu cât se modifică venitul maxim V al firmei la o variaţie cu o unitate a disponibilului unei resurse.*

Este important de reţinut faptul că aceste concluzii rămân valabile numai în situaţia în care componentele vectorului *b* variază între anumite limite! În cazul general, atunci când se iau în considerare toate valorile posibile ale lui *b*, adică $b \in R_+^m$, funcţia $V^*(b)$ este *subaditivă* şi *pozitiv omogenă*. Aceasta înseamnă:

$$V^*(b + b') \leq V^*(b) + V^*(b') \qquad (\forall)\ b, b' \in R_+^m$$

$$V^*(tb) = tV^*(b) \qquad (\forall)\ t \geq 0, b \in R_+^m$$

Reamintim că în baza TEC soluţiile optime x^*, u^* al problemelor (P) şi (Q) satisfac relaţiile:

$$x_j^* > 0 \Rightarrow a_{1j}u_1^* + a_{2j}u_2^* + \ldots + a_{mj}u_m^* = c_j \qquad (2.4.5)$$

$$a_{i1}x_1^* + a_{i2}x_2^* + \ldots + a_{in}x_n^* < b_i \Rightarrow u_i^* = 0 \qquad (2.4.5')$$

$$u_i^* > 0 \Rightarrow a_{i1}x_1^* + a_{i2}x_2^* + \ldots + a_{in}x_n^* = b_i \qquad (2.4.6)$$

$$a_{1j}u_1^* + a_{2j}u_2^* + \ldots + a_{mj}u_m^* > c_j \Rightarrow x_j^* = 0 \qquad (2.4.6')$$

- (2.4.5) arată că dacă bunul G_j intră în combinaţia optimă atunci preţul său este egal cu partea din venitul maxim al firmei corespunzătoare resurselor încorporate într-o unitate din el.
- (2.4.5') arată că dacă o resursă nu este prevăzută a se consuma în întregime - spunem că este *excedentară* - preţul său dual este 0. Această concluzie reprezintă versiunea liniară a legii cererii şi ofertei - preţul unei mărfi pentru care oferta este mai mare decât cererea trebuie să scadă.
- (2.4.6) arată că o resursă cu preţ dual semnificativ trebuie consumată în întregime; creşterea cu o unitate a disponibilului aducând o creştere a venitului maxim conform (2.4.4).
- (2.4.6') arată că dacă pentru un bun valoarea resurselor încorporate într-o unitate - valoare măsurată în preţurile duale optime - depăşeşte preţul său atunci acesta nu intră în combinaţia optimală. Diferenţa $\sum_{i=1}^{m} a_{ij}u_j^* - c_j$ reprezintă *pierderea potenţială* de venit pe care firma o va înregistra dacă totuşi decide realizarea unei unităţi din bunul *j*.

Condiţiile (2.4.5), (2.4.5'), (2.4.6), (2.4.6') date de TEC se mai numesc şi ***condiţii de echilibru*** de unde şi numele de ***preţuri de echilibru*** dat uneori preţurilor duale optime.

4.3. Structura mulţimii soluţiilor admisibile ale unei probleme de programare liniară

În această secţiune ne vom opri asupra principalelor proprietăţi geometrice pe care le posedă mulţimea soluţiilor unui sistem de ecuaţii şi inecuaţii liniare. Aceste proprietăţi sunt determinante în înţelegerea mecanismului metodei simplex de rezolvare a programelor liniare.

Parcurgerea acestei secţiuni necesită câteva rudimente de calcul matricial şi algebră liniară. Vectorii cu care se va opera vor fi subînţeleşi, după caz, fie linii fie coloane. De regulă, scrierea în text a unui vector se va face în linie ca de exemplu $v = (a_1, a_2, \ldots, a_m)$; dacă este necesar ca v să fie considerat vector coloană se va folosi operatorul de transpunere: $v = (a_1, a_2, \ldots, a_m)^T$.

4.3.1 Câteva elemente de analiză convexă liniară

Fiind date două puncte $x, y \in R^n$ mulţimea:

$$[x,y] = \{z = (1-\alpha)x + \alpha y, 0 \le \alpha \le 1\}$$

se numeşte ***segment*** (închis) cu extremităţile x şi y. Se ştie că în R^2 sau în R^3 acest concept se suprapune peste conceptul geometric uzual. Pentru $\alpha = 0$, respectiv $\alpha = 1$, avem $z \equiv x$, respectiv $z \equiv y$. Punctele $z = (1-\alpha)x + \alpha y$ corespunzătoare valorilor $\alpha \in (0,1)$ se numesc puncte *interioare* ale segmentului $[x,y]$. Pentru $\alpha = \frac{1}{2}$ găsim $z = \frac{1}{2}(x+y) \equiv$ mijlocul segmentului $[x,y]$.

*O mulţime $X \subseteq R^n$ se zice **convexă** dacă o dată cu două puncte conţine şi segmentul care le uneşte.*

Formal :

$$X \text{ convexă} \Leftrightarrow (\forall)x, y \in X, (\forall)\ \alpha \in [0,1], z = (1-\alpha)x + \alpha y \in X$$

Se verifică imediat că intersecţia mai multor mulţimi convexe este o mulţime convexă.

Fie $a = (a_1, a_2, ..., a_n)$ un vector nenul şi b un scalar. Este uşor de văzut că mulţimea:

$$S = \left\{ x = (x_1, x_2, ..., x_n)^T \,\middle|\, ax \le b \Leftrightarrow a_1x_1 + a_2x_2 + ... + a_nx_n \le b \right\}$$

este convexă. Ea se numeşte ***semispaţiu,*** în timp ce mulţimea:

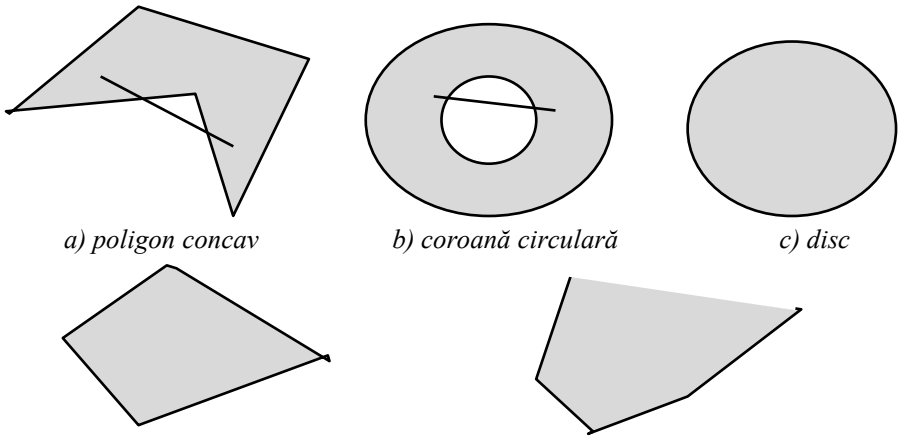
$$H = \left\{ x = (x_1, x_2, ..., x_n)^T \,\middle|\, ax = b \Leftrightarrow a_1x_1 + a_2x_2 + ... + a_nx_n = b \right\}$$

se numeşte ***hiperplan***. Este clar că şi H este o mulţime convexă ca intersecţie a semispaţiului *S* de mai sus, cu semispaţiul:

$$S' = \left\{ x \in R^n \,\middle|\, ax \ge b \Leftrightarrow (-a)x \le -b \Leftrightarrow -a_1x_1 - a_2x_2 - ... - a_nx_n \le -b \right\}$$

O intersecţie finită de semispaţii se numeşte ***mulţime poliedrală***. Evident, o mulţime poliedrală este convexă, reciproca nefiind în general adevărată.

În figura 4.1.1 sunt prezentate câteva mulţimi convexe şi neconvexe în plan. Este clar că mulţimile a) şi b) nu sunt convexe. Discul c) este o mulţime convexă dar nu este poliedrală, fiind în fapt intersecţia infinită a tuturor semispaţiilor care conţin discul şi sunt mărginite de tangentele la circumferinţă. Poligonul convex d) este intersecţia a 4 semispaţii aşa cum se arată în fig.4.1.2



a) poligon concav *b) coroană circulară* *c) disc*

d) poligon convex *e) mulţime poliedrală nemărginită*

Notă: Toate mulţimile specificate sunt presupuse ***închise*** *adică* ***îşi conţin frontierele***.

Figura 3.1.1

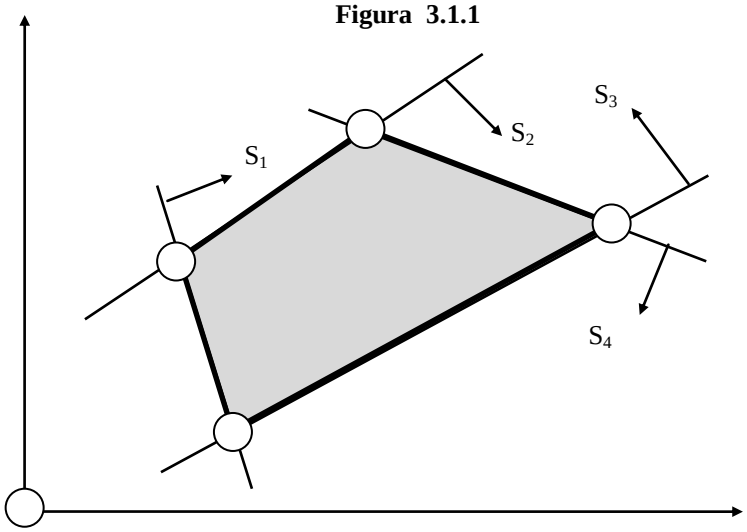


Figura 3.1.2

Din cele de mai sus rezultă că orice *mulţime poliedrală în* R^n *se identifică cu mulţimea soluţiilor unui sistem de ecuaţii şi/sau inecuaţii liniare* în *n* variabile. În particular:

Mulţimea $\mathcal{A}_P$ *a soluţiilor admisibile ale unui program liniar* (P) *este o* ***mulţime convexă****, poliedrală şi închisă. Frontiera sa se compune din toate punctele ale căror coordonate satisfac cu egalitate cel puţin una din restricţii.*

Se numeşte ***vârf*** al unei mulţimi convexe $X \subseteq R^n$ un punct $v \in X$ cu proprietatea că nu există un segment $[x,y] \subset X$ care să conţină pe *v* ca punct interior. În R^2 sau R^3 regăsim conceptul geometric uzual.

O mulţime poliedrală are întotdeauna un număr finit de vârfuri (posibil nici unul); de exemplu, poligonul d) din fig. 4.1.1 are patru vârfuri în timp ce un semispaţiu nu are vârfuri. Discul c) are o infinitate de vârfuri: orice punct de pe circumferinţă are această calitate.

Mulţimile d) şi e) din fig.4.1.1 sunt amândouă poliedrale dar e) este *nemărginită*. Pentru a caracteriza această proprietate avem nevoie de un nou concept, cel de *rază extremă*. Pentru a nu depăşi cadrul orar al cursului vom lăsa la latitudinea cititorului aprofundarea conceptului utilizând ca suport bibliografic www.asecib.ase.ro – Cursuri on-line – 9. Nica V. şi colectiv, ***Cercetări Operaţionale I.*** Vom caracteriza însă în detaliu această situaţie în cadrul metodei Simplex de rezolvare a programelor liniare (vezi secţiunea 4 a Cap. 3).

4.3.2 Teorema centrală a programării liniare

Să considerăm acum un program liniar (P) în care funcţia obiectiv se maximizează şi să ne situăm în cazul în care programul (P) este compatibil adică mulţimea soluţiilor sale admisibile $\mathcal{A}_P$ este nevidă. Am văzut că $\mathcal{A}_P$ este o mulţime poliedrală şi convexă având un număr finit de vârfuri $v^1, v^2, ..., v^p$. Aşa cum va rezulta din secţiunea 4.1, $\mathcal{A}_P$ are cel puţin un vârf, adică $p \geq 1$. Vom enunţa acum *teorema centrală a programării liniare*.

TEOREMA[*)]**.** *Dacă programul (P) are optim finit, atunci o soluţie optimă se găseşte într-unul din vârfurile mulţimii soluţiilor admisibile* $\mathcal{A}_P$ *.*

Importanţa acestei teoreme este covârşitoare: ea reduce problema găsirii unei soluţii optime x^* *din mulţimea, în general infinită,* $\mathcal{A}_P$ *a tuturor soluţiilor admisibile ale programului* (P), *la identificarea acestei soluţii în mulţimea finită a vârfurilor lui* $\mathcal{A}_P$*.*

Recapitulând modul în care diferitele proprietăţi discutate au fost implicate în obţinerea acestui rezultat fundamental să reţinem că:

- *Convexitatea mulţimii soluţiilor admisibile* $\mathcal{A}_P$ *situează soluţiile optime, dacă acestea există, pe frontiera lui* $\mathcal{A}_P$;
- *Deoarece* $\mathcal{A}_P$ *este poliedrală, iar funcţia obiectiv este liniară, cel puţin una din soluţiile optime este un vârf al lui* $\mathcal{A}_P$*.*

Teorema furnizează următorul procedeu "naiv" de rezolvare a unui program liniar (P):

- se "generează" lista (finită) a vârfurilor mulţimii $\mathcal{A}_P$;

[*)] **NOTĂ:** Pentru demonstraţiile Teoremelor şi Lemelor din acest capitol, vezi www.asecib.ase.ro – Cursuri on-line – 9. Nica V. şi colectiv, ***Cercetări Operaţionale I***.

- prin înlocuire succesivă în funcţia obiectiv se reţine vârful care oferă acesteia valoarea maximă (sau minimă, după caz).

Procedeul ridică la rândul său următoarele *probleme principiale*:

1) Cum recunoaştem compatibilitatea programului (P) ?

2) Cum "calculăm" un vârf al mulţimii $\mathcal{A}_P$ sau mai corect spus cum se caracterizează "algebric" un vârf ?

3) Pentru obţinerea soluţiei optime este necesar să generăm toate vârfurile mulţimii $\mathcal{A}_P$? întrebarea este serioasă deoarece şi pentru programe liniare de dimensiuni reduse (adică cu un număr relativ mic de restricţii şi variabile) numărul vârfurilor este foarte mare.

4) Chiar dacă reuşim, prin enumerarea explicită a tuturor vârfurilor, să găsim pe acela care maximizează funcţia obiectiv, aceasta nu înseamnă obligatoriu că am rezolvat programul dat! Este posibil ca programul respectiv să aibă optim infinit! Cum se recunoaşte acest fapt?

Vom răspunde progresiv la toate chestiunile menţionate în secţiunile următoare.

4.3.3 Corespondenţa $\mathcal{A}_P \sim \mathcal{A}_{FSP}$

Considerăm o problemă de programare liniară (P) care conţine cel puţin o restricţie inegalitate şi fie (FSP) forma sa standard (vezi secţiunea 1.4) Pentru simplificarea notaţiilor, vom presupune că (P) este în *formă canonică de maximizare*:

$$(P)\begin{cases} \max f = cx \\ Ax \le b \\ x \ge 0 \end{cases}$$

unde:

$$A=\begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mn} \end{bmatrix} \quad b=\begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_m \end{bmatrix} \quad x=\begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} \qquad (4.3.1)$$

$$c=\begin{bmatrix} c_1 & c_2 & \cdots & c_n \end{bmatrix}$$

ştim din secţiunea 1.3 că orice program liniar poate fi scris în această formă. Forma standard a programului (P) va fi:

$$(FSP)\begin{cases} max\ f = cx \\ Ax + y = b \quad \text{în care: } y=\begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_m \end{bmatrix} \text{ este vectorul variabilelor de abatere.} \\ x \ge 0\ ,\ y \ge 0 \end{cases}$$

Între mulţimile de soluţii admisibile $\mathcal{A}_P \subset R^n$ şi $\mathcal{A}_{FSP} \subset R^{n+m}$ există o *corespondenţă bijectivă* $\Phi(x) = (x, y)$, unde $y = b - Ax$, a cărei inversă este proiecţia $\Phi^{-1}(x,y) = x$. Am remarcat deja în secţiunea 1.4 că prin corespondenţa Φ, soluţiile optime ale celor două probleme se corespund. În fapt, Φ are următoarea proprietate mai generală.

Teorema 4.3.1 *Dacă x este un vârf al mulţimii $\mathcal{A}_P$ atunci $\Phi(x) = (x, y)$ cu $y = b - Ax$ este un vârf al mulţimii $\mathcal{A}_{FSP}$. Reciproc, dacă (x, y) este un vârf al mulţimii $\mathcal{A}_{FSP}$ atunci $\Phi^{-1}(x,y) = x$ este un vârf al mulţimii $\mathcal{A}_P$.*

În baza acestei teoreme precum şi a teoremei centrale a programării liniare (secţiunea 4.2), pentru a rezolva problema (P) este suficient să căutăm *soluţia optimă a formei sale standard* (FSP) printre vârfurile mulţimii $\mathcal{A}_{FSP}$.

Vom vedea în secţiunea următoare cum se caracterizează "algebric" vârfurile mulţimii soluţiilor admisibile ale unui program liniar în formă standard. Tot acolo vom arăta că dacă un program (P) este compatibil atunci $\mathcal{A}_P$ are cel puţin un vârf şi în orice caz un număr finit de asemenea elemente. Pe baza acestor rezultate, vom putea descrie în paragraful 4 o metodă efectivă de rezolvare a unei probleme de programare liniară.

4.3.4. Soluţii de bază ale unui program liniar

Să considerăm acum un program liniar (P) în formă standard:

$$(P)\begin{cases} max\ f = cx \\ Ax = b \\ x \geq 0 \end{cases}$$

în care masivele *A,b,c,x* au semnificaţiile din (4.3.1). Vom pune în evidenţă coloanele matricii A:

$$A = [A^1 , A^2 , \dots , A^n]$$

Definiţie: *Soluţia* $\bar{x} = (\bar{x}_1, \bar{x}_2, \dots, \bar{x}_n)^T$ *a problemei în formă standard (P), nu neapărat admisibilă, se numeşte* ***soluţie de bază*** *dacă mulţimea coloanelor* A^i *corespunzătoare componentelor* $\bar{x}_i \neq 0$ *este* ***liniar independentă****.*

Fie $I(\bar{x})$ mulţimea indicilor $i \in \{1, 2, \dots, m\}$ cu proprietatea că $\bar{x}_i \neq 0$.

Lema 1: *Dacă* $\bar{x}$ *şi* $\bar{x}'$ *sunt soluţii de bază ale programului (P) şi* $I(\bar{x}') \subseteq I(\bar{x})$ *atunci* $\bar{x} = \bar{x}'$ *(şi deci* $I(\bar{x}) = I(\bar{x}')$*).*

Demonstraţie: Este clar că $\bar{x}_i = \bar{x}_i' = 0$ pentru indicii $i \notin I(\bar{x})$. Atunci egalităţile:

$$\sum_{i \in I(\bar{x})} \bar{x}_i A^i = \sum_{i \in I(\bar{x}')} \bar{x}_i' A^i = b$$

implică:

$$\sum_{i \in I(\bar{x})} (\bar{x}_i - \bar{x}_i') A^i = 0$$

de unde rezultă $\bar{x}_i = \bar{x}_i'$ pentru toţi $i \in I(\bar{x})$, deoarece prin ipoteză coloanele A^i, $i \in I(\bar{x})$ sunt liniar independente. ■

Lema 2: *Fie* $\bar{x} = (\bar{x}_1, \bar{x}_2, \dots, \bar{x}_n)^T$ *o soluţie admisibilă a problemei (P) care nu este soluţie de bază. Atunci există un vector* $y \in R^n$ *şi un interval* $[\underline{\lambda}, \bar{\lambda}] \subset R^n \cup \{-\infty, +\infty\}$ *astfel încât:*

1) $Ay = 0$*;*

2) $[\underline{\lambda}, \bar{\lambda}]$ *conţine pe 0 şi nu se reduce la acest punct;* $\underline{\lambda}$ *şi* $\bar{\lambda}$ *nu sunt simultan infinite;*

3) pentru orice $\lambda \in [\underline{\lambda}, \bar{\lambda}]$ *vectorul* $x(\lambda) = \bar{x} + \lambda y$ *este o soluţie admisibilă a problemei (P);*

4) Dacă de exemplu, $\bar{\lambda}$ *este finit şi* $\bar{x}' = x(\bar{\lambda})$ *atunci* $I(\bar{x}') \subseteq I(\bar{x})$ *dar* $|I(\bar{x}')| \leq |I(\bar{x})| - 1$ *adică* $\bar{x}'$ *are mai puţine componente nenule decât* $\bar{x}$.

Demonstraţie: Din ipoteză rezultă că mulţimea coloanelor A^i, $i \in I(\bar{x})$ este liniar independentă.

Există prin urmare scalarii y_i, $i \in I(\bar{x})$ nu toţi nuli astfel încât:

$$\sum_{i \in I(\bar{x})} y_i A^i = 0$$

Punând $y_i = 0$ pentru $i \notin I(\bar{x})$ obţinem un vector $y = (y_1, y_2, \ldots, y_n) \in R^n$ cu proprietatea că:

$$\sum_{i=1}^{n} y_i A^i = 0 \quad \Leftrightarrow \quad Ay = 0$$

Afirmaţia 1) este demonstrată.

Pentru orice $\lambda \in R$ vectorul $x(\lambda) = \bar{x} + \lambda y$ este o soluţie a problemei (P) deoarece $Ax(\lambda) = A\bar{x} + \lambda Ay = b$.

Impunând condiţia de admisibilitate $x(\lambda) \geq 0$ obţinem pentru λ intervalul de valori permise $[\underline{\lambda}, \bar{\lambda}]$ în care:

$$\underline{\lambda} = \begin{cases} \max\limits_{i \in I(\bar{x}), y_i > 0} \left\{ -\bar{x}_i / y_i \right\} \\ -\infty \text{ daca toti } y_i \leq 0 \end{cases} \qquad \bar{\lambda} = \begin{cases} \min\limits_{i \in I(\bar{x}), y_i < 0} \left\{ -\bar{x}_i / y_i \right\} \\ +\infty \text{ daca toti } y_i \geq 0 \end{cases}$$

Avem $\underline{\lambda} < 0 < \bar{\lambda}$ şi deoarece $y \neq 0$, cel puţin una din extremităţile $\underline{\lambda}, \bar{\lambda}$ este finită. Astfel şi afirmaţiile 2) , 3) sunt probate.

Să presupunem, în final că $\bar{\lambda}$ este finit. Atunci va exista un indice $r \in I(\bar{x})$ astfel că: $y_r < 0$ şi $\bar{\lambda} = -\bar{x}_r / y_r$ .Dacă $\bar{x}' = x(\bar{\lambda})$ este clar că $I(\bar{x}') \subseteq I(\bar{x})$ şi cum $\bar{x}'_r = \bar{x}_r + \bar{\lambda} y_r = 0$ iar $\bar{x}_r > 0$ urmează că $|I(\bar{x}')| < |I(\bar{x})|$ şi ultima afirmaţie este dovedită. ■

Teorema 4.4.1 *O soluţie admisibilă $\bar{x} = (\bar{x}_1, \bar{x}_2, \ldots, \bar{x}_n)^T$ a problemei (P) este un vârf al mulţimii $\mathcal{A}_P$ dacă şi numai dacă $\bar{x}$ este o soluţie de bază.*

Demonstraţie: Să presupunem că $\bar{x}$ este vârf dar nu este soluţie de bază. Conform lemei 2 există $y \in R^n$ cu $Ay = 0$ şi intervalul $[\underline{\lambda}, \bar{\lambda}]$ care conţine pe 0 şi nu se reduce la acesta, astfel încât $x(\lambda) = \bar{x} + \lambda y$ să fie o soluţie a programului (P) oricare ar fi $\lambda \in [\underline{\lambda}, \bar{\lambda}]$. Alegem $\varepsilon > 0$ suficient de mic astfel încât $[-\varepsilon, +\varepsilon] \subset [\underline{\lambda}, \bar{\lambda}]$ şi punem: $x^1 = \bar{x} - \varepsilon y$, $x^2 = \bar{x} + \varepsilon y$. Atunci $x^1, x^2 \in \mathcal{A}_P$, $x^1 \neq x^2$ şi $\bar{x} = \frac{1}{2}(x^1 + x^2)$ în contradicţie cu ipoteza că $\bar{x}$ este vârf al mulţimii $\mathcal{A}_P$.

Pentru reciprocă să presupunem că $\bar{x}$ este o soluţie de bază fără a fi vârf. Atunci există $x^1, x^2 \in A_P$, $x^1 \neq x^2$ şi $\alpha \in (0,1)$ astfel încât $\bar{x} = (1-\alpha)x^1 + \alpha x^2$. Pentru $i \notin I(\bar{x})$ avem $(1-\alpha)x_i^1 + \alpha x_i^2 = 0$ şi cum $x_i^1 \geq 0$, $x_i^2 \geq 0$ rezultă $x_i^1 = x_i^2 = 0$. în consecinţă, $I(x^1) \subseteq I(\bar{x})$, $I(x^2) \subseteq I(\bar{x})$ şi în virtutea lemei 1 rezultă $x^1 = x^2 = \bar{x}$, în contradicţie cu ipoteza făcută. ■

Teorema 4.4.2 *Dacă programul în formă standard (P) este compatibil atunci mulţimea soluţiilor sale admisibile $\mathcal{A}_P$ are cel puţin o soluţie de bază, deci un vârf.*

Demonstraţie Fie $\bar{x} = (\bar{x}_1, \bar{x}_2, \ldots, \bar{x}_n)^T$ o soluţie admisibilă a problemei (P). Vom proceda prin inducţie după numărul k al componentelor $\bar{x}_i > 0$ .Dacă $k = 0$ atunci $\bar{x} = (0,0,\ldots,0)$ este o soluţie de bază întrucât o mulţime vidă de vectori este, prin convenţie,liniar independentă. Dacă $k > 0$ există două situaţii de examinat:

1) Coloanele A^i, $i \in I(\bar{x})$ sunt liniar independente. Atunci $\bar{x}$ este o soluţie de bază.

2) Coloanele A^i, $i \in I(\bar{x})$ sunt liniar dependente. Conform lemei 2 va exista o soluţie admisibilă $\bar{x}'$ cu $I(\bar{x}') \subset I(\bar{x})$ dar cu mai puţine componente nenule decât $\bar{x}$. Repetând

raţionamentul, este clar că într-un număr finit de paşi se ajunge la situaţia 1) adică la o soluţie de bază. ■

Consecinţă: *Mulţimea $\mathcal{A}_P$ a soluţiilor admisibile ale unui program liniar compatibil are cel puţin un vârf.*

Demonstraţie: Să presupunem că (P) nu este în formă standard, altminteri avem teorema 4.4.2. Fie (FSP) forma standard a programului (P). Deoarece avem corespondenţa bijectivă Φ: $\mathcal{A}_P \cong \mathcal{A}_{FSP}$ deducem că $\mathcal{A}_{FSP} \neq \emptyset$ pentru că prin ipoteză $\mathcal{A}_P \neq \emptyset$. Prin teorema 4.4.2 mulţimea $\mathcal{A}_{FSP}$ are cel puţin un vârf; acesta, prin proiecţia Φ^{-1} va fi un vârf al mulţimii $\mathcal{A}_P$, graţie teoremei 4.3.1. ■

Având în vedere caracterizarea algebrică a vârfurilor dată în teorema 4.4.1, teorema centrală a programării liniare poate fi formulată şi în următorii termeni.

Teorema 4.4.3 *Dacă un program liniar în formă standard are optim finit, cel puţin una din soluţiile sale optime este o soluţie de bază.*

Mai rămâne de arătat că mulţimea soluţiilor admisibile ale unui program liniar are un număr finit de vârfuri. În virtutea teoremelor 4.3.1 şi 4.4.1, aceasta revine la a arăta că un program liniar (P) în formă standard are un număr finit de soluţii admisibile de bază. Faptul rezultă nemijlocit din aceea că *numărul sistemelor liniar independente ce pot fi extrase dintr-o mulţime finită de vectori este finit*. Vom preciza acest lucru sub forma unei teoreme în secţiunea următoare, în care vom introduce un concept uşor diferit de cel de soluţie de bază, acela de *soluţie asociată unei baze* a programului (P). Noul concept are avantajul de a fi mai uşor de manipulat în practică.

4.3.4. Baze ale unui program liniar în formă standard. Soluţia asociată unei baze

În notaţiile secţiunii precedente facem ipoteza:

$$\text{rang}A = m < n \qquad (4.5.1)$$

Ipoteza ne asigură că ecuaţiile ce compun sistemul liniar $Ax = b$ al restricţiilor sunt independente şi că acest sistem are o infinitate de soluţii. Să notăm că ipoteza nu implică în mod necesar şi existenţa soluţiilor admisibile (adică cu toate componentele nenegative) pentru sistemul considerat. După cum vom vedea în secţiunea 4.6 ipoteza făcută nu este deloc restrictivă.

*Deoarece rang$A$ = m, în matricea A va exista cel puţin un grup de **m** coloane liniar independente, constituind deci o bază a spaţiului R^m.* Un asemenea grup se va numi *bază a programului (P)* şi numărul acestora este finit, nedepăşind $C_n^m = \dfrac{n!}{m!(n-m)!}$.

Fie ***B*** o bază a programului (P), ***I*** mulţimea indicilor coloanelor din B şi ***J*** mulţimea indicilor coloanelor din A care nu sunt în B. Renumerotând convenabil variabilele programului vom scrie matricea A în forma:

$$A = [B\ ,\ S] \quad \text{cu} \quad B = [A^i]_{i \in I} \ , \quad S = [A^j]_{j \in J}$$

Partiţionăm corespunzător vectorul (coloană) x al variabilelor:

$$x = \begin{bmatrix} x^B \\ x^S \end{bmatrix} \qquad \text{cu} \quad x^B = \left[x_i\right]_{i \in I} \ , \quad x^S = \left[x_j\right]_{j \in J}$$

ca şi vectorul (linie) al coeficienţilor funcţiei obiectiv:

$$c = [c^B, c^S] \qquad \text{cu } c^B = [c_i]_{i \in I} \ , \qquad c^S = [c_j]_{j \in J}$$

în raport cu baza B aleasă, variabilele x_i , $i \in I$ se vor numi *variabile bazice*, iar toate celelalte *nebazice* sau *secundare*.

Scriem sistemul $Ax = b$ în forma:

$$[B,S]\begin{bmatrix} x^B \\ x^S \end{bmatrix} = b \quad \Leftrightarrow \quad Bx^B + Sx^S = b$$

Deoarece B este o bază a spaţiului R^m, B (ca matrice!) este nesingulară deci inversabilă. Înmulţind la stânga cu B^{-1} obţinem:

$$x^B + \bar{S}x^S = \bar{b} \quad \text{cu} \quad \bar{S} = B^{-1}S = \left[\bar{a}_{ij}\right]_{i\in I, j\in J}, \bar{b} = B^{-1}b = \left[\bar{b}_i\right]_{i\in I} \qquad (4.5.2)$$

Va fi util să punem în evidenţă coloanele matricii $\bar{S}$:

$$\bar{S} = B^{-1}\left[A^j\right]_{j\in J} = \left[B^{-1}A^j\right]_{j\in J} = \left[\bar{A}^j\right]_{j\in J}$$

cu

$$\bar{A}^j = B^{-1}A^j = \left[\bar{a}_{ij}\right]_{i\in I} \qquad (4.5.3)$$

Utilizăm (4.5.2) pentru a elimina din expresia funcţiei obiectiv variabilele bazice:

$$f = \left[c^B, c^S\right]\begin{bmatrix} x^B \\ x^S \end{bmatrix} = c^B x^B + c^S x^S = c^B(\bar{b} - \bar{S}x^S) + c^S x^S =$$
$$= c^B\bar{b} - (c^B\bar{S} - c^S)x^S = \bar{f} - \bar{c}^S x^S \qquad (4.5.4)$$

unde:

$$\bar{f} = c^B\bar{b}\,(= c^B B^{-1}b) = \sum_{i\in I} c_i\bar{b}_i \qquad (4.5.5)$$

şi:

$$\bar{c}^S = c^B\bar{S} - c^S\,(= c^B B^{-1}S - c^S) = \left[\bar{c}_j\right]_{j\in J}$$

în care:

$$\bar{c}_j = c^B\bar{A}^j - c_j\,(= c^B B^{-1}A^j - c_j) = \sum_{i\in I} c_i\bar{a}_{ij} - c_j \qquad (4.5.6)$$

Astfel, în raport cu baza B, programul (P) poate fi adus la următoarea formă echivalentă, conform (4.5.2) şi (4.5.4):

$$(P_B)\begin{cases} \max f = \bar{f} - \bar{c}^S x^S \\ x^B + \bar{S}x^S = \bar{b} \\ x^B \geq 0, x^S \geq 0 \end{cases} \quad \Leftrightarrow \quad (P_B)\begin{cases} \max f = \bar{f} - \sum_{j\in J}\bar{c}_j x_j \\ x_i + \sum_{j\in J}\bar{a}_{ij}x_j = \bar{b}_i \, i \in I \\ x_i \geq 0, i \in I; x_j \geq 0, j \in J \end{cases} \qquad (4.5.7)$$

(P_B) se numeşte ***forma explicită a programului (P) în raport cu baza B***. Comparând (P_B) cu (P) constatăm că *efectul înmulţirii cu B^{-1} a sistemului $Ax = b$ este explicitarea variabilelor bazice x_i , $i \in I$ în funcţie de cele nebazice x_j , $j \in J$.*

Luând în (P_B):

$$x^S = 0 \qquad \Leftrightarrow \quad x_j = 0 \,,\, j \in J \qquad (4.5.8)$$

obţinem:

$$x^B = \bar{b} \quad \Leftrightarrow \quad x_i = \bar{b}_i \quad i \in I \qquad (4.5.9)$$

Vectorul:

$$\bar{x} = \begin{bmatrix} \bar{b} \\ 0 \end{bmatrix} \begin{matrix} \leftarrow x^B \\ \leftarrow x^S \end{matrix} \qquad (4.5.10)$$

este evident o soluţie a programului (P), numită ***soluţia asociată bazei B***. Vom spune că B este o *bază admisibilă* dacă soluţia asociată (4.5.10) este admisibilă, ceea ce revine la a spune că $\bar{b}_i \geq 0\,, i \in I$.

Prin construcţie, *soluţia asociată unei baze a programului (P) este o soluţie de bază* în sensul definiţiei din secţiunea precedentă. Reciproca nu este în general adevărată. Astfel, dacă $\bar{x} = (\bar{x}_1, \bar{x}_2, \ldots, \bar{x}_n)^T$ este o soluţie de bază a programului (P), numărul componentelor nenule este ≤ *m*. Dacă acest număr este exact *m* atunci $\bar{x}$ este soluţia asociată bazei formate din coloanele matricii A corespunzătoare celor *m* componente nenule. Spunem în acest caz că $\bar{x}$ este o ***soluţie de bază nedegenerată*.** Dacă numărul componentelor nenule este < *m*, coloanele corespunzătoare acestor componente pot fi completate până la o bază în mai multe moduri, astfel că $\bar{x}$ se asociază la mai multe baze! Dacă se întâmplă aşa spunem că $\bar{x}$ este o ***soluţie degenerată***.

Deoarece (P) are un număr finit de baze, el va avea şi un număr finit de soluţii asociate acestor baze şi de aici un număr finit de soluţii de bază.

În baza relaţiilor (4.5.8), (4.5.9) constanta $\bar{f}$ din (4.5.5) reprezintă *valoarea funcţiei obiectiv* a programului (P) în soluţia (4.5.10) asociată bazei B. Componentele $\bar{c}_j\,, j \in J$ (definite în (4.5.6)) ale vectorului $\bar{c}^S$ din (4.5.4) vor juca un rol esenţial în caracterizarea optimalităţii unei soluţii admisibile de bază. Ele se numesc ***costuri reduse*** (notate Δ_j în alte materiale de specialitate) şi sunt întotdeauna *asociate variabilelor nebazice*. Coeficienţii numerici ai formei explicite (4.5.7) se trec într-un tabel **(T_B)** al cărui format este indicat în tabelul 4.5.1. Tabelul (T_B) se numeşte ***tabelul simplex asociat bazei* B**. Coloana "B" conţine vectorii bazei **B**, coloana "CB" conţine **C**oeficienţii din funcţia obiectiv ai variabilelor **B**azice iar în coloana "VVB" apar **V**alorile **V**ariabilelor **B**azice. În ultima linie a tabelului, numită şi *linia test*, apar valoarea $\bar{f}$ a funcţiei obiectiv în soluţia asociată bazei B precum şi costurile reduse $\bar{c}_j$, $j \in J$ corespunzătoare coloanelor nebazice. Formula (4.5.5) arată că $\bar{f}$ este produsul scalar al coloanelor "CB" şi "VVB" în timp ce relaţia (4.5.6) arată că $\bar{c}_j$ se obţine din produsul scalar al coloanelor "CB" şi "A^j" scăzând costul c_j scris deasupra coloanei "A^j".

Tabelul 4.5.1

				c_i		c_r		c_j		c_k	
CB	B	VVB	…	A^i	…	A^r	…	A^j	…	A^k	…
⋮	⋮	⋮		⋮		⋮		⋮		⋮	
c_i	A^i	$\bar{b}_i$	…	1	…	0	…	$\bar{a}_{ij}$	…	$\bar{a}_{ik}$	…
⋮	⋮	⋮		⋮		⋮		⋮		⋮	
c_r	A^r	$\bar{b}_r$	…	0	…	1	…	$\bar{a}_{rj}$	…	$\boxed{\bar{a}_{rk}}$	…
⋮	⋮	⋮		⋮		⋮		⋮		⋮	
	f	$\bar{f}$	…	*	…	*	…	$\bar{c}_j$	…	$\bar{c}_k$	…

(T_B)

Extindem definiţia costului redus şi la coloanele "A^i" , $i \in I$: $\bar{c}_i = c_i - c_i = 0$. Aceste costuri reduse au fost notate în linia test cu asteriscuri (*) spre a le deosebi de eventualele costuri reduse nule $\bar{c}_j$, $j \in J$ care, după cum vom vedea în secţiunea 4.4.2, indică - "la optim" - prezenţa mai multor soluţii optime de bază.

4.3. Metoda simplex de rezolvare a programelor liniare de optimizare va fi prezentată în cadrul cursului de Cercetări Operaționale din anul 3

Exemplul 4.2.1 Considerăm programul:

$$(P)\begin{cases} \max f = 2x_1 + x_2 \\ x_1 - x_2 \le 4\ ;\ 3x_1 - x_2 \le 18\ ;\ -x_1 + 2x_2 \le 6 \\ x_1 \ge 0\ ,\ x_2 \ge 0 \end{cases}$$

a cărui mulţime de soluţii admisibile $\mathcal{A}_P$ este vizualizată în figura 4.2.1.

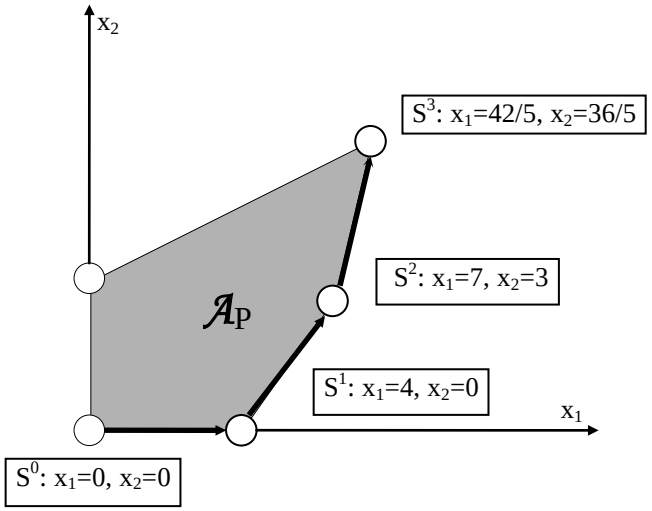


Figura 4.2.1

Aducem (P) la forma standard adăugând variabilele de abatere x_3, x_4, x_5:

$$(FSP)\begin{cases} \max f = 2x_1 + x_2 \\ x_1 - x_2 + x_3 = 4 \\ 3x_1 - x_2 + x_4 = 18 \\ -x_1 + 2x_2 + x_5 = 6 \\ x_j \ge 0\ ,\ j = 1,\ldots,5 \end{cases}$$

Se observă că matricea A a coeficienţilor formei standard (FSP) conţine baza unitară E = [A^3,A^4,A^5] (deci FSP este chiar forma bună a lui P pentru aplicarea algoritmului Simplex Primal) a cărei soluţie asociată:

$$x_1 = 0\ ,\ x_2 = 0\ ,\ x_3 = 4\ ,\ x_4 = 18\ ,\ x_5 = 6$$

este admisibilă.

4.4.3. Determinarea unei soluţii admisibile de start (Metoda bazei artificiale)

După cum s-a specificat, aplicarea algoritmului simplex necesită cunoaşterea unei baze admisibile de start precum şi a formei explicite asociate acesteia, celelalte forme explicite deducându-se una din alta prin pivotare gaussiană. Chiar şi pentru probleme de dimensiuni mici, găsirea unei

asemenea baze de start prin simpla inspectare a coloanelor matricii A, se dovedeşte a fi o treabă complicată, ne mai vorbind de faptul că este posibil ca problema să nu aibă soluţii admisibile. În plus, să nu uităm că teoria metodei simplex s-a bazat esenţial pe ipoteza că restricţiile problemei sunt liniar independente, lucru iarăşi greu de verificat în practică. Pentru obţinerea formei explicite iniţiale avem nevoie de câteva pregătiri.

Vom spune că programul în formă standard (P) este în ***formă bună*** dacă matricea A conţine o ***submatrice unitate*** de ordinul *m* (= numărul restricţiilor) iar *termenii liberi* sunt *nenegativi*. Dacă este aşa, (P) satisface condiţia (4.5.1) iar soluţia asociată bazei unitare este admisibilă şi poate fi considerată ca soluţie de start pentru aplicarea algoritmului simplex. Dacă (P) nu este în formă bună, el se poate aduce la această formă, notată (FBP), în felul următor:

- *în caz că unele restricţii ale programului iniţial au termeni liberi negativi înmulţim aceste restricţii cu -1; în acest fel toţi termenii liberi ai restricţiilor problemei de rezolvat vor fi* ≥ 0.
- *Se aduce problema la forma standard adăugând variabile de abatere în restricţiile inegalităţi.*
- *Dacă matricea programului rezultat nu conţine toate coloanele matricii unitate de ordinul m, în anumite restricţii se vor adăuga noi variabile* ***nenegative*** *pentru crearea coloanelor lipsă; aceste noi variabile se numesc* ***variabile artificiale*** *şi, spre deosebire de cele de abatere, apar şi în funcţia obiectiv cu un coeficient comun, foarte mare în valoare absolută. Coeficientul va fi* ***negativ*** *dacă funcţia obiectiv se* ***maximizează*** *şi* ***pozitiv în caz contrar***.

Se observă imediat că *dacă programul iniţial (P) este compatibil*, soluţiile sale admisibile se identifică cu *acele soluţii ale formei bune în care* ***variabilele artificiale au valoarea zero***. Prin faptul că variabilele artificiale sunt însoţite în expresia funcţiei obiectiv de nişte "penalizări" foarte mari, metoda simplex este "instruită" să caute tocmai asemenea soluţii! Şi astfel, rezolvarea formei bune ne conduce la unul din următoarele cazuri:

1. Forma bună are optim infinit. Atunci şi programul iniţial are ***optim infinit.***

2. Forma bună are optim finit dar în soluţia optimă cel puţin o variabilă artificială are valoare nenulă. Atunci programul original este ***incompatibil.***

3. Forma bună are optim finit şi în soluţia optimă toate variabilele artificiale au valoarea zero. Ignorând valorile acestor variabile se obţine soluţia optimă a programului iniţial.

Exemplul 4.3.1 Considerăm următorul program împreună cu forma sa standard:

$$(P)\begin{cases} (max)\, f = 2x_1 + 3x_2 \\ 2x_1 + x_2 \geq 40 \\ x_1 + 3x_2 \geq 30 \\ x_1 + x_2 \leq 30 \\ x_1 \geq 0\,,\ x_2 \geq 0 \end{cases} \Rightarrow (FSP)\begin{cases} (max)\, f = 2x_1 + 3x_2 \\ 2x_1 + x_2 - x_3 = 40 \\ x_1 + 3x_2 - x_4 = 30 \\ x_1 + x_2 + x_5 = 30 \\ x_j \geq 0\,,\ j = 1,\ldots,5 \end{cases}$$

Se constată că (FSP) nu este în forma bună neconţinând decât un singur vector al matricii unitare de ordinul 3. Vom creea o asemenea matrice introducând în primele două restricţii variabilele artificiale x_6 şi x_7. Obţinem programul:

$$(FBP)\begin{cases} (max)\, f = 2x_1 + 3x_2 - Mx_6 - Mx_7 \\ 2x_1 + x_2 - x_3 + x_6 = 40 \\ x_1 + 3x_2 - x_4 + x_7 = 30 \\ x_1 + x_2 + x_5 = 30 \\ x_j \geq 0\,,\ j = 1,\ldots,7 \end{cases}, \quad M >> 0$$

Putem aplica algoritmul simplex programului (FBP) plecând de la baza unitară $E = [A^6, A^7, A^5]$ şi de la soluţia asociată acesteia:

$$x_1 = x_2 = x_3 = x_4 = 0 , \quad x_5 = 30 , x_6 = 40 , x_7 = 30 .$$

Punctele din R^2 corespunzătoare celor patru soluţii generate de algoritm sunt $S^0 = (0,0)$, $S^1 = (0,10)$, $S^2 = (18,4)$, $S^3 = (10,20)$. Urmărind fig. 4.3.1 se constată că punctele S^0 şi S^1, ce corespund unor soluţii ale programului (FBP) în care cel puţin o variabilă artificială are valoare nenulă, sunt în afara mulţimii $\mathcal{A}_P$ în timp ce S^2 şi S^3, corespunzătoare unor soluţii în care $x_6 = x_7 = 0$, sunt vârfuri ale acestei mulţimi.

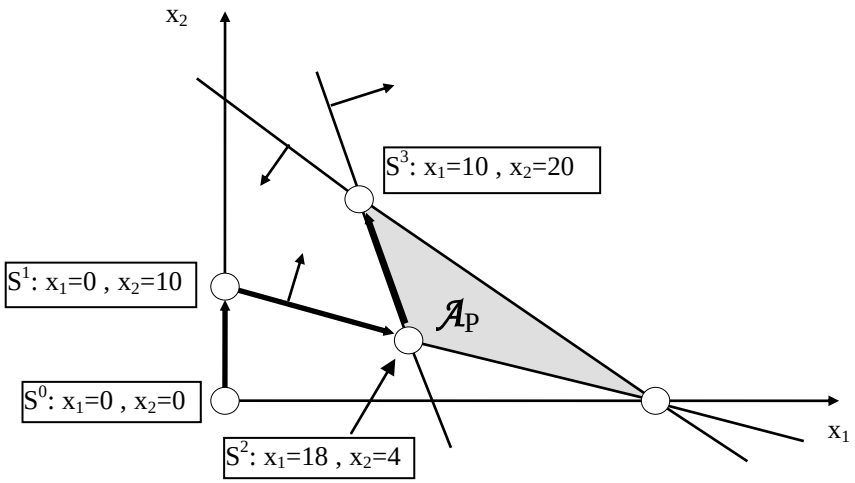


Figura 4.3.1

CAPITOLUL 5
PROBLEME DE OPTIMIZARE IN RETELE DE TRANSPORT SI DISTRIBUTIE

IMPORTANT: Suportul de curs al acestui capitol are are la bază lucrarea: Nica Vasile, Ciobanu Gh., Mustaţă Floare, Mărăcine Virginia, **"*Cercetări operaţionale I - Programare liniară, Probleme de optimizare în reţele de transport şi distribuţie, Teoria jocurilor strategice*"** Editura MATRIX ROM, Bucureşti 1998.

5.1. Modelarea problemelor de transport şi distribuţie

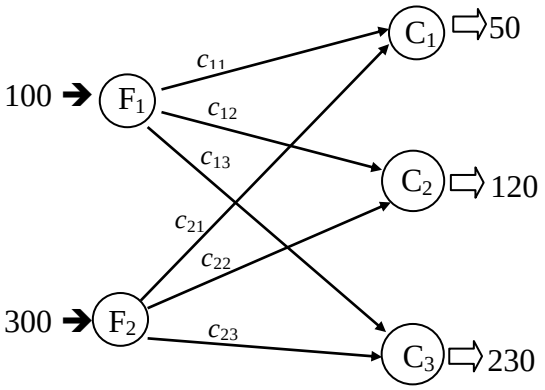
Într-o mare varietate de contexte se pune problema *deplasării* unei cantităţi Q ce poate fi materie, energie, informaţie, etc. din unele locuri numite *surse* în alte locuri numite *destinaţii*, această deplasare realizându-se pe anumite *rute de legătură*. Unităţile indivizibile ale cantităţii Q care se deplasează de-alungul rutelor se vor numi *unităţi de flux.*

5.1.1. O clasificare a problemelor de transport şi distribuţie

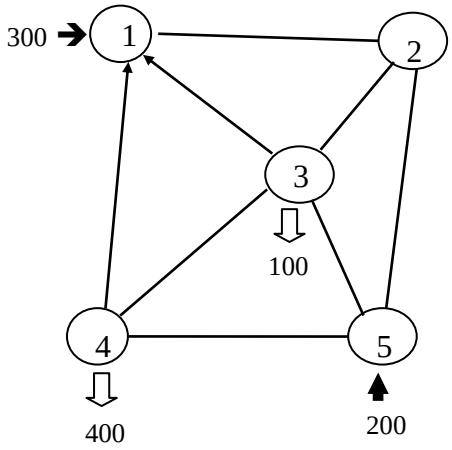
Pentru Cercetarea Operaţională, problema enunţată va prezenta interes numai dacă respectă următoarele ipoteze:

a) cel puţin o sursă poate aproviziona mai multe destinaţii şi cel puţin o destinaţie poate primi unităţi de flux de la mai multe surse.

Rutele de legătură pot avea şi alte puncte comune în afara surselor şi destinaţiilor, numite *puncte intermediare* sau *de tranzit.* Nu sunt excluse legăturile directe între surse sau între destinaţii. În principiu, orice rută poate fi parcursă în ambele sensuri, dar pot exista şi rute cu sens unic.



a)



b)

Figura 5.1.1

Ansamblul surselor, destinaţiilor, al punctelor intermediare şi al rutelor de legătură se va numi ***reţea de transport****;* el se identifică cu un *graf neorientat sau parţial orientat* ca în figura 5.1.1.

b) Unele rute de legătură pot avea limitări superioare şi / sau inferioare pentru volumul unităţilor de flux ce se deplasează într-un sens sau altul. Aceste limitări poartă numele de ***capacităţi*** (inferioare, respectiv superioare). În continuare, vom avea în vedere numai cazul în care toate capacităţile inferioare sunt egale cu *zero*, capacităţile superioare fiind exprimate prin numere pozitive.

*c) Există un cost al deplasării unei unităţi de flux de la un punct al reţelei la altul,*cost care poate fi exprimat în *bani, timp* sau *distanţă.* Sunt situaţii în care acest cost poate semnifica *profitul* obţinut de pe urma deplasării. Pe aceeaşi rută, atât costurile cât şi capacităţile pot diferi în funcţie de sensul de parcurgere.

Ipoteza a) va fi întotdeauna presupusă în timp ce ipotezele b) şi c) pot fiinţa separat sau simultan.

1) În prezenţa ipotezei c) şi absenţa condiţiei b) se pune problema deplasării cantităţii de flux Q de la surse la destinaţii *la un cost total minim.* Dacă sursele sunt în legătură directă cu destinaţiile obţinem ***problema clasică de transport***, care va face obiectul secţiunilor imediat următoare. Cazul general, în care exisă şi puncte intermediare, este cunoscut sub numele de *problema transferului* şi el nu face obiectul cursului de faţă. În cazul particular al unei singure surse *s*, al unei singure destinaţii *t* şi a unei singure unităţi de flux se obţine ***problema drumului de cost minim de la s la t***.

2) În prezenţa ipotezei b) şi absenţa ipotezei c) se pune problema dacă reţeaua, ale cărei rute sunt *capacitate, este capabilă să permită acoperirea integrală a cererilor în punctele de destinaţie.* Pentru aceasta, se va rezolva problema determinării volumului *maxim* Q* de unităţi de flux ce pot fi deplasate de la surse la destinaţii. Dacă Q* < Q vor exista destinaţii a căror cerere este acoperită doar în parte şi atunci se ridică problema *măririi capacităţii de transfer a reţelei.* Am descris succint ***problema fluxului maxim***.

3) În prezenţa simultană a ipotezelor b) şi c) se pune *problema satisfacerii cererilor în punctele de destinaţie la un cost de transport minim.* Ca şi în cazul precedent vom avea în vedere o problemă modificată: vom determina mai întâi cantitatea maximă de flux ce poate fi deplasată de la surse la destinaţii şi apoi modul de organizare al deplasării astfel încât costul operaţiei să fie minim. Aceasta este ***problema fluxului (maxim) de cost minim***.

În secţiunile următoare *ne vom ocupa numai de problema clasică de transport*; ea va fi privită ca o problemă de programare liniară cu o structură specială şi rezolvată prin metodele programării liniare. Celelalte probleme identificate vor fi tratate în cuprinsul Capitolului 1 al cursului, Teoria Grafurilor, rspectiv în cadrul cursului de Cercetări Operaţionale din anul 3.

5.1.2. Problema clasică de transport. Problema de transport echilibrată (PTE)

Un produs omogen (de exemplu, bere) se află disponibil în localităţile F_1, F_2, ..., F_m în cantităţile a_1, a_2, ..., a_m şi este cerut pentru consum în centrele C_1, C_2, ..., C_n în cantităţile b_1, b_2, ..., b_n. Se presupune cunoscut costul c_{ij} al transportului unei unităţi de produs de la F_i la C_j. *Se pune problema satisfacerii cererii în punctele de consum la un cost total de transport minim*.

Centrele furnizoare, centrele consumatoare, legăturile directe între ele şi costurile unitare de transport sunt vizualizate de obicei printr-un *graf orientat* (ca în figura 5.1.1 *a*).

Evident, o condiţie necesară şi suficientă pentru existenţa unei soluţii a problemei formulate este ca totalul cantităţilor disponibile să acopere totalul cererilor:

$$\sum_{i=1}^{m} a_i \geq \sum_{j=1}^{n} b_j \qquad (5.2.1)$$

În continuare, condiţia (5.2.1) va fi presupusă îndeplinită. Vom presupune de asemenea că :

$$a_i > 0,\ i = 1,...,m \ \text{şi}\ b_j > 0,\ j = 1,...,n.$$

Dacă notăm cu x_{ij} cantitatea livrată de furnizorul F_i consumatorului C_j, modelul matematic al problemei (clasice) de transport este:

(PT) *Să se determine* $(x_{ij}^*)\ i = 1,\ldots,m\ ,\ j = 1,\ldots,n$ *care satisfac restricţiile:*

$$\sum_{j=1}^{n} x_{ij} \leq a_i \quad i = 1,\ldots,m \qquad (5.2.2)$$

$$\sum_{i=1}^{m} x_{ij} \geq b_j \quad j = 1,\ldots,n \qquad (5.2.3)$$

condiţiile de nenegativitate:

$$x_{ij} \geq 0 \quad i = 1,...,m; \quad j = 1,...,n$$

şi care minimizează fincţia obiectiv:

$$f = \sum_{i=1}^{m} \sum_{j=1}^{n} c_{ij} x_{ij} \qquad (5.2.4)$$

Inegalităţile (5.2.2) exprimă cerinţa ca totalul livrărilor fiecărui furnizor să se încadreze în disponibil; inegalităţile (5.2.3) arată că cererea fiecărui consumator trebuie să fie acoperită prin totalul cantităţilor primite; în fine, (5.2.4) este expresia costului total al transportului.

Vom spune că problema de transport (PT) este *echilibrată* dacă:

$$\sum_{i=1}^{m} a_i = \sum_{j=1}^{n} b_j \qquad (5.2.5)$$

Se observă imediat că (5.2.5) atrage după sine satisfacerea cu egalitate a restricţiilor (5.2.2) şi (5.2.5). Prin urmare, modelul matematic al unei probleme de transport echilibrate este:

$$(\text{PTE})\begin{cases} \sum_{j=1}^{n} x_{ij} = a_i \; i = 1,...,m & (5.2.6) \\ \sum_{i=1}^{m} x_{ij} = b_j \; j = 1,...,n & (5.2.7) \\ x_{ij} \geq 0 \\ (\min) f = \sum_{i=1}^{m}\sum_{j=1}^{n} c_{ij} x_{ij} \end{cases}$$

Remarcăm faptul că (PTE) este o problemă de programare liniară în formă standard, cu $m + n$ restricţii şi $m \cdot n$ variabile.

Se arată uşor că matricea A a coeficienţilor sistemului de restricţii din (PTE), care apare în tabelul 1.1.1, are rangul $m + n - 1$. Aceasta înseamnă că în sistemul (5.2.6) - (5.2.7) putem elimina o ecuaţie fără ca mulţimea soluţiilor admisibile să se modifice. În consecinţă:

	x_{11}	$...x_{1j}...$	x_{1n}	.....	x_{i1}	$...x_{ij}...$	x_{in}	.....	x_{m1}	$...x_{mj}...$	x_{mn}
1	1	....1...	1		0	....0...	0		0	.....0...	0
⋮											
i	0	....0...	0		1	....1...	1		0	.....0...	0
⋮											
m	0	....0...	0		0	....0...	0		1	.....1...	1
1	1	....0...	0		1	....0...	0		1	.....0...	0
⋮											
j	0	....1...	0		0	....1...	0		0	.....1...	0
⋮											
n	0	....0...	1		0	....0...	1		0	.....0...	1

Tabelul 1.1.1

Orice soluţie de bază a problemei (PTE) are cel mult m + n - 1 componente nenule.

O soluţie *de bază* a problemei (PTE) se va numi ***nedegenerată*** dacă are exact $m + n - 1$ componente *nenule*; altminteri, ea se va zice ***degenerată.***

În continuare, vom presupune că orice soluţie a problemei (PTE) este nedegenerată. Cazul în care (PTE) are şi soluţii degenerate va fi analizat în secţiunea 5.2.5.

Observaţie: Orice soluţie $\bar{x} = (\bar{x}_{ij})$ a PTE va fi înscrisă într-un *tabel* cu *m rânduri*, corepunzătoare furnizorilor şi *n coloane* corespunzătoare consumatorilor. Tabelul va avea *m·n celule* sau *rute*; celula din rîndul *i* şi coloana *j*, notată (F_i ,C_j) sau simplu (*i, j*), va conţine componenta $\bar{x}_{ij}$ a soluţiei $\bar{x}$.

Exemplul 1.2.1 Să considerăm problema de transport echilibrată definită de următoarele date:

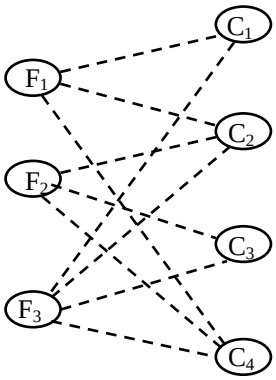
Tabelul 1.2.1

	C_1	C_2	C_3	C_4	Disponibil
F_1	3	3	1	4	15
F_2	3	4	3	6	17
F_3	4	3	6	2	18
Necesar	10	12	9	19	50

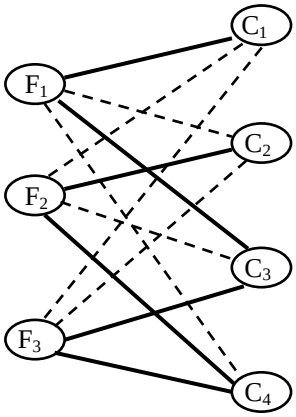
Modelul matematic:

$$\begin{cases} x_{11}+x_{12}+x_{13}+x_{14}=15 \\ x_{21}+x_{22}+x_{23}+x_{24}=17 \\ x_{31}+x_{32}+x_{33}+x_{34}=18 \\ x_{11}+x_{21}+x_{31}=10 \\ x_{12}+x_{22}+x_{32}=12 \\ x_{13}+x_{23}+x_{33}=9 \\ x_{14}+x_{24}+x_{34}=19 \\ x_{ij} \geq 0 \\ (\min) f = 3x_{11}+3x_{12}+x_{13}+4x_{14}+3x_{21}+4x_{22}+ \\ +3x_{23}+6x_{24}+4x_{31}+3x_{32}+6x_{33}+2x_{34} \end{cases}$$

Toate legăturile de la fiecare furnizor *Fi* la fiecare consumator *Cj* pot fi vizualizate cu ajutorul desenului alăturat numit, aşa cum ştim, GRAF. Aşa cum se observă, furnizorii şi consumatorii sunt reprezentaţi cu ajutorul unor "noduri" unite prin "muchii" de legătură.



Obţinerea unei soluţii nedegenerate a PTE nu presupune utilizarea tuturor muchiilor de legătură dintre furnizorii *Fi* şi consumatorii *Cj*. De exemplu, în graful anterior, selectăm în scopul transportului bunurilor între *Fi* şi *Cj* muchiile desenate cu linie continuă şi obţinem în acest fel desenul de mai jos:



Dacă anulăm în sistemul restricţiilor toate variabilele x_{ij} cu proprietatea că muchiile corespunzătoare $\{F_i, C_j\}$ nu vor fi utilizate (desenate cu linie punctată), rezultă sistemul şi soluţia:

$$\begin{cases} x_{11}+x_{13}=15 \\ x_{22}+x_{24}=17 \\ x_{33}+x_{34}=18 \\ x_{11}=10 \\ x_{22}=12 \\ x_{13}+x_{33}=9 \\ x_{24}+x_{34}=19 \end{cases} \Rightarrow \begin{cases} x_{11}=10 \\ x_{13}=5 \\ x_{22}=12 \\ x_{24}=5 \\ x_{33}=4 \\ x_{34}=14 \end{cases}$$

	C_1	C_2	C_3	C_4
F_1	10		5	
F_2		12		5
F_3			4	14

5.2. Adaptarea metodei simplex la rezolvarea PTE

Fiind o problemă de programare liniară, PTE se poate rezolva cu ajutorul metodei simplex. Totuşi, algoritmul simplex va avea în acest caz o descriere specifică datorită unei proprietăţi importante pe care o are matricea A a coeficienţilor PTE. Într-adevăr, se poate demonstra că *orice determinant extras din A are valoarea -1 , 0 sau 1*. În consecinţă, dacă disponibilele a_1, a_2,..., a_m şi cererile b_1, b_2,..., b_n sunt exprimate prin numere *întregi*, orice soluţie de bază va avea componentele întregi şi astfel PTE *va avea cel puţin o soluţie optimă cu componente întregi.*

5.2.1. Determinarea unei soluţii admisibile de bază iniţiale

Să considerăm următorul procedeu general de construire a unei soluţii admisibile a PTE. Componentele ei vor fi determinate progresiv şi înscrise într-un tabel aşa cum s-a menţionat în observaţia din secţiunea 5.1.2.

Iniţializare: Toate cele $m \cdot n$ rute ale tabelului sunt considerate *neblocate*.

Etapa k, $k \geq 1$.

- *Se alege* o rută (F_{i_k}, C_{j_k}) dintre cele *neblocate*.
- Se pune $x_{i_k j_k} = \min(a_{i_k}, b_{j_k})$ şi se *blochează* ruta *aleasă*. Vom spune că pe ruta (în celula) (F_{i_k}, C_{j_k}) s-a făcut *alocarea* $x_{i_k j_k}$.
- Se actualizează:

$$a_{i_k} \leftarrow a_{i_k} - x_{i_k j_k}$$
$$b_{j_k} \leftarrow b_{j_k} - x_{i_k j_k}$$

- Dacă $a_{i_k} = 0$ se pune $x_{i_k j} = 0$ pe toate rutele (F_{i_k}, C_j), $j \neq j_k$ încă neblocate, după care acestea se declară *blocate.*

Dacă $b_{j_k} = 0$ se pune $x_{ij_k} = 0$ pe toate rutele (F_i, C_{j_k}), $i \neq i_k$ încă neblocate, după care acestea se declară *blocate*.

- Dacă toate cele $m \cdot n$ rute au fost blocate STOP. Altminteri, se actualizează $k \leftarrow k + 1$ şi se reiau operaţiile de mai sus.

Se constată fără dificultate că ansamblul de valori numerice $x = (x_{ij})$ rezultate în urma aplicării algoritmului constituie o soluţie admisibilă a PTE; *se poate arăta că x este o soluţie de bază.*

În ipoteza că valorile x_{ij} au fost trecute *progresiv* în tabelul menţionat la început, este uşor de văzut că *dacă la fiecare etapă se blochează fie rutele aparţinând unui rând fie cele aparţinând unei coloane, atunci soluţia construită are exact* ***m + n - 1 componente nenule****, altfel spus este* ***nedegenerată****.* Dacă din contră, după efectuarea unei alocări, se blochează *simultan* atât rândul cât şi coloana rutei în care s-a făcut alocarea, soluţia rezultată va fi *degenerată*! Vom discuta în secţiunea 5.2.5 ce trebuie făcut în acest caz.

Toate metodele de generare a unei soluţii iniţiale pentru PTE au la bază procedura de mai sus. Ele *se deosebesc prin modul de alegere* a rutelor (F_{i_1}, C_{j_1}), (F_{i_2}, C_{j_2}), ...Astfel:

1) În metoda **colţului de nord - vest (N - V)** , prima rută aleasă este (F_1, C_1). Celelalte se aleg astfel încât $1 = i_1 \leq i_2 \leq i_3 \leq \ldots$ şi $1 = j_1 \leq j_2 \leq j_3 \leq \ldots$

2) În metoda **costului (unitar de transport) minim** prima rută (F_{i_1}, C_{j_1}) aleasă corespunde *celui mai mic cost unitar* de transport. La etapa k, ruta (F_{i_k}, C_{j_k}) corespunde *celui mai mic cost* unitar de transport de pe rutele încă *neblocate* la această etapă.

3) Metoda **diferenţelor maxime (Vogel)** este o metodă mai elaborată. Presupunem costurile unitare de transport înscrise într-un tabel cu *m* rânduri şi *n* coloane.

- Pe fiecare rând şi pe fiecare coloană a acestui tabel se calculează *diferenţa* dintre *cel mai mic* cost de transport şi cel *imediat* superior; dacă costul minim *nu* este unic, diferenţa se va lua egală cu *zero*.
- Se identifică rândul sau coloana cu *cea mai mare* diferenţă şi aici, *în ruta de cost minim*, se execută prima *alocare* din algoritmul precedent.
- Se refac diferenţele pe rândurile şi coloanele *neblocate* folosindu-se numai *costuri "neblocate"*, după care se reia procedura de alocare.

Exemplul 2.1.1 Vom genera o soluţie de bază iniţială pentru problema de transport echilibrată din exemplul 1.2.1, folosind pe rând cele trei metode sus amintite.

1) Prin metoda colţului N - V se obţine soluţia:

10 (1)	5 (2)		
	7 (3)	9 (4)	1 (5)
			18 (6)

Tabelul 2.1.1

Atenţie: numerele înscrise în paranteze indică *ordinea* alocărilor!

Costul asociat acestei soluţii: $f = 10 \times 3 + 5 \times 3 + 7 \times 4 + 9 \times 3 + 1 \times 6 + 18 \times 2 = 142$ u.m.

2) Prin metoda costului minim rezultă soluţia:

	6 (4)	9 (1)	
10 (3)	6 (5)		1 (6)
			18 (2)

Tabelul 2.1.2

Costul asociat: $f = 123$ u.m.

3) Prin metoda diferenţelor maxime se obţine soluţia:

	5 (6)	9 (1)	1 (3)
10 (4)	7 (5)		
			18 (2)

Tabelul 2.1.3

Costul asociat: $f = 122$ u.m.

(Cititorul este îndemnat să refacă " în dinamică" construcţia soluţiilor de mai sus!)

Observaţie: Metoda colţului N - V, deşi mai simplă, produce în general soluţii cu cost mai ridicat deoarece nu ţine seama în nici un fel de costurile unitare de transport. Celelalte metode, acordând prioritate rutelor "mai ieftine", dau soluţii mai apropiate de soluţia optimă.

Experimentele numerice au arătat că metoda diferenţelor maxime produce de foarte multe ori chiar soluţia optimă sau în orice caz o soluţie foarte apropiată de aceasta, aşa încât mulţi utilizatori preferă s-o adopte ca soluţie suboptimală. În aplicaţiile numerice vom lucra numai cu Metoda costului minim pe tabel, respectiv cu Metoda Vogel.

5.2.2. Testarea optimalităţii unei soluţii admisibile de bază a PTE

Fie $\bar{x} = (\bar{x}_{ij})$ o soluţie admisibilă de bază a PTE, presupusă nedegenerată. Obiectivul este de a găsi o condiţie pentru recunoaşterea optimalităţii sale. Să considerăm duala problemei de transport echilibrate:

(Q) *Să se determine* $u^* = (u_1^*, u_2^*, \ldots, u_m^*)$ *şi* $v^* = (v_1^*, v_2^*, \ldots, v_n^*)$
care maximizează funcţia:

$$g = \sum_{i=1}^{m} a_i u_i + \sum_{j=1}^{n} b_j v_j$$

cu restricţiile:

$$u_i + v_j \le c_{ij} \quad i = 1, \ldots, m;\ j = 1, \ldots, n \qquad (5.2.1)$$

Teorema ecarturilor complementare (cap. 4, teorema 2.3.4) arată că soluţia $\bar{x}$ este optimă dacă şi numai dacă există $(\bar{u}, \bar{v}) = (\bar{u}_1, \ldots, \bar{u}_m, \bar{v}_1, \ldots, \bar{v}_n)$ care satisfac restricţiile (5.2.1) ale dualei (Q), astfel încât $(\bar{x}, \bar{u}, \bar{v})$ să verifice relaţiile:

$$x_{ij}(u_i + v_j - c_{ij}) = 0 \quad i = 1, \ldots, m;\ j = 1, \ldots, n \qquad (5.2.2)$$

Să notăm cu *I* mulţimea rutelor (F_i,C_j) (sau mai simplu (*i*, *j*)) cu proprietatea că $\bar{x}_{ij} \neq 0$. Deoarece soluţia $\bar{x}$ a fost presupusă nedegenerată, mulţimea *I* are *m* + *n* - *1* elemente. Din (5.2.2) rezultă că, pentru a fi optimă, soluţia duală ($\bar{u}$,$\bar{v}$) de mai sus trebuie să verifice relaţiile:

$$u_i + v_j = c_{ij} \quad (\forall)\ (i, j) \in I \qquad (5.2.3)$$

Să remarcăm că (5.2.3) este un *sistem liniar* cu *m* + *n* - 1 ecuaţii şi *m* + *n* variabile. Deoarece I *se identifică cu un arbore maximal* în graful G asociat PTE (a se vedea Capitolul 1!), sistemul (5.2.3) este întotdeauna *compatibil nedeterminat* soluţiile sale fiind de forma:

$$u_i = u_i^0 + k \quad i = 1, \ldots, m \quad ; \quad v_j = v_j^0 - k \quad j = 1, \ldots, n \qquad (5.2.4)$$

unde *k* este un parametru iar (u^0, v^0) = $(u_1^0, \ldots, u_m^0, v_1^0, \ldots, v_n^0)$ este o soluţie particulară a sistemului (5.2.3). Rezultă că valorile expresiilor:

$$\Delta_{ij} = u_i + v_j - c_{ij} \qquad (5.2.5)$$

nu depind de soluţia ($\bar{u}$,$\bar{v}$) a sistemului (5.2.3), deoarece:

$$\Delta_{ij} = (u_i^0 + k) + (v_j^0 - k) - c_{ij} = u_i^0 + v_j^0 - c_{ij}$$

Din consideraţiile de mai sus deducem următorul criteriu de recunoaştere a optimalităţii soluţiei $\bar{x}$:

Determinăm o soluţie particulară ($\bar{u}$,$\bar{v}$) a sistemului (5.2.3) şi calculăm $\Delta_{ij} = \bar{u}_i + \bar{v}_j - c_{ij}$ *pentru toate cuplurile (i, j)*$\notin I$ (pentru (*i, j*)$\in I$ este evident că $\Delta_{ij} = 0$).

Dacă toţi Δ_{ij} ***calculaţi sunt*** ≤ 0***, atunci*** $(\bar{u}, \bar{v})$ ***este o soluţie a dualei (Q) care împreună cu*** $\bar{x}$ ***satisface condiţiile de ecart complementar* (5.2.2) *şi, în consecinţă,*** $\bar{x}$ ***este o soluţie optimă a PTE.***

Exemplul 2.2.1 Vom testa optimalitatea soluţiei determinate în exemplul 5.1.1 prin metoda diferenţelor maxime.

Sistemul $u_i + v_j = c_{ij}$ $(i,j) \in$ I

	v_1	v_2	v_3	v_4
u_1		5	9	1
u_2	10	7		
u_3				18

$\Rightarrow$

	$u_1 + v_2 = 3$	$u_1 + v_3 = 1$	$u_1 + v_4 = 4$
$u_2 + v_1 = 3$	$u_2 + v_2 = 4$		
			$u_3 + v_4 = 2$

Tabelele 2.2.1 - 2.2.2

Determinăm o soluţie particulară a acestui sistem luând $u_1 = 0$

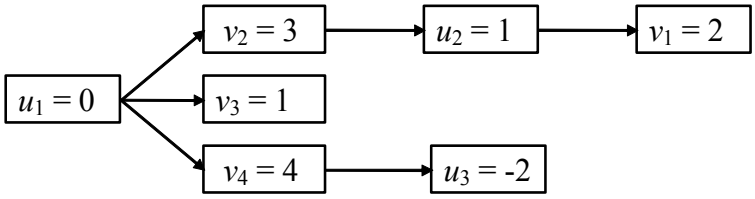


Figura 5.2.1

Este interesant de urmărit rezolvarea sistemului pe graful (arborele) asociat soluţiei $\bar{x}$:

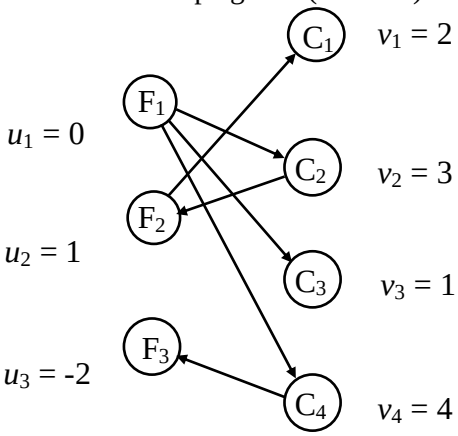


Figura 5.2.2

Calculăm mărimile $\Delta_{ij} = u_i + v_j - c_{ij}$ numai pentru rutele "*neocupate*" adică pentru rutele $(i,j) \notin I$ (Pentru cele "*ocupate*", adică pentru rutele $(i,j) \in I$, ştim că $\Delta_{ij} = 0$; în tabelul 2.2.3 aceste zerouri au fost înlocuite cu asteriscuri pentru a le deosebi de eventualele mărimi Δ_{ij} nule, asociate unor rute neocupate. *Prezenţa unor asemenea mărimi în cazul în care soluţia curentă este optimă arată că aceasta nu este unică!*

	$v_1 = 2$	$v_2 = 3$	$v_3 = 1$	$v_4 = 4$
$u_1 = 0$	-1	*	*	*
$u_2 = 1$	*	*	-1	-1
$u_3 = -2$	-4	-2	-7	*

Tabelul 2.2.3

Constatăm că soluţia testată verifică criteriul de optimalitate.

5.2.3 Îmbunătăţirea unei soluţii de bază

Să presupunem că soluţia $\bar{x}$ considerată în secţiunea precedentă nu verifică testul de optimalitate; aceasta înseamnă că:

$$\textit{există o rută } (i_0, j_0) = (F_{i_0}, C_{j_0}) \notin I \textit{ cu proprietatea } \Delta_{i_0 j_0} > 0.$$

Vom construi o soluţie admisibilă de bază $\bar{x}'$ mai bună decît $\bar{x}$ în sensul că $\bar{x}'$ implică un cost total de transport mai mic.

Adăugăm muchia $\{F_{i_0}, C_{j_0}\}$ la graful - arbore H corespunzător soluţiei $\bar{x}$. Conform unei proprietăţi a arborilor se va forma un unic *ciclu*. Deoarece arborele H împreună cu muchia adăugată fac parte din graful asociat PTE care este *bipartit*, ciclul format are un număr *par* de muchii. Să parcurgem muchiile ciclului într-unul din cele două sensuri posibile, plecând de exemplu din nodul F_{i_0}:

$$F_{i_0} \to C_{j_0} \to F_{i_1} \to C_{j_1} \to \ldots \to F_{i_p} \to C_{j_p} \to F_{i_0} \qquad (5.3.1)$$

Fie θ > 0. Construim o soluţie variabilă $\tilde{x} = (\tilde{x}_{ij})$ a PTE punând:

$$\tilde{x}_{i_0j_0} = \theta, \tilde{x}_{i_1j_0} = \bar{x}_{i_1j_0} - \theta, \tilde{x}_{i_1j_1} = \bar{x}_{i_1j_1} + \theta, \ldots, \tilde{x}_{i_pj_p} = \bar{x}_{i_pj_p} + \theta, \tilde{x}_{i_0j_p} = \bar{x}_{i_0j_p} - \theta$$
$$\tilde{x}_{ij} = \bar{x}_{ij} \text{ in rest} \qquad (5.3.2)$$

Deoarece soluţia $\bar{x}$ este presupusă nedegenerată, pentru θ suficient de mic, (5.3.2) este o soluţie admisibilă a PTE. Costul asociat soluţiei $\tilde{x}$ va diferi de costul asociat soluţiei $\bar{x}$ prin valoarea:

$$\Delta f = f(\tilde{x}) - f(\bar{x}) = \theta(c_{i_0j_0} - c_{i_1j_0} + c_{i_1j_1} - c_{i_2j_1} + \ldots + c_{i_pj_p} - c_{i_0j_p}) =$$
$$\theta[c_{i_0j_0} - (\bar{u}_{i_1} + \bar{v}_{j_0}) + (\bar{u}_{i_1} + \bar{v}_{j_1}) - (\bar{u}_{i_2} + \bar{v}_{j_1}) + \ldots + (\bar{u}_{i_p} + \bar{v}_{j_p}) - (\bar{u}_{i_0} + \bar{v}_{j_p})] =$$
$$\theta(c_{i_0j_0} - \bar{u}_{i_0} - \bar{v}_{j_0}) = -\theta \cdot \Delta_{i_0j_0} < 0 \quad \text{de unde:}$$

$$f(\tilde{x}) = f(\bar{x}) - \theta\Delta_{i_0j_0} \qquad (5.3.3)$$

Relaţia (5.3.3) arată că $\tilde{x}$ implică un cost total de transport mai mic decât soluţia curentă $\bar{x}$, diferenţa fiind cu atât mai mare cu cât θ sau $\Delta_{i_0j_0}$ este mai mare.

Pentru a menţine admisibilitatea soluţiei (2.3.2) este necesar ca:

$$\bar{x}_{i_1j_0} - \theta \geq 0,\ \bar{x}_{i_2j_1} - \theta \geq 0, \ldots,\ \bar{x}_{i_0j_p} - \theta \geq 0$$

de unde rezultă că θ nu poate depăşi valoarea:

$$\theta_0 = \min\left\{\bar{x}_{i_1j_0}, \bar{x}_{i_2j_1}, \ldots, \bar{x}_{i_0j_p}\right\} = \bar{x}_{i_sj_{s-1}} \qquad (5.3.4)$$

Luăm în (5.3.2) θ = θ_0 şi notăm cu $\bar{x}'$ soluţia $\tilde{x}$ corespunzătoare. Ea va avea cel mult *m* + *n* - 1 componente nenule; într-adevăr, în noua soluţie $\bar{x}'_{i_0j_0} = \theta_0 > 0$ iar în vechea soluţie $\bar{x}_{i_0j_0} = 0$. Pe de altă parte, componenta $\bar{x}'_{i_sj_{s-1}}$ corespunzătoare minimului din (5.3.4) este acum nulă în timp ce în vechea soluţie $\bar{x}$ era pozitivă. Noua soluţie $\bar{x}'$ este şi o soluţie de bază, deoarece ea corespunde arborelui H' obţinut din $H \cup \left\{F_{i_0}, C_{j_0}\right\}$ îndepărtând muchia $\left\{F_{i_s}, C_{j_{s-1}}\right\}$.

Exemplul 2.3.1 Considerăm problema de transport echilibrată din exemplul 1.2.1 şi soluţia $\bar{x}$ generată prin metoda colţului N-V în exemplul 2.1.1. Aplicăm acestei soluţii testul de optimalitate din secţiunea precedentă.

10	5		
	7	9	`1
			18

Sistemul $u_i + v_j = c_{ij}$ $(i,j) \in I$

$u_1 + v_1 = 3$	$u_1 + v_2 = 3$		
	$u_2 + v_2 = 4$	$u_2 + v_3 = 3$	$u_2 + v_4 = 6$
			$u_3 + v_4 = 2$

Tabelele 2.3.1 -2.3.2

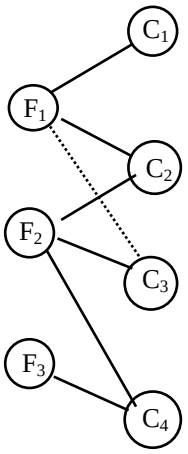
Determinăm o soluţie particulară a sistemului luând de exemplu $u_2 = 0$, după care calculăm mărimile $\Delta_{ij} = u_i + v_j - c_{ij}$ pentru rutele "neocupate":

	$v_1 = 4$	$v_2 = 4$	$v_3 = 3$	$v_4 = 6$
$u_1 = -1$	*	*	1	1
$u_2 = 0$	1	*	*	*
$u_3 = -4$	-4	-3	-7	*

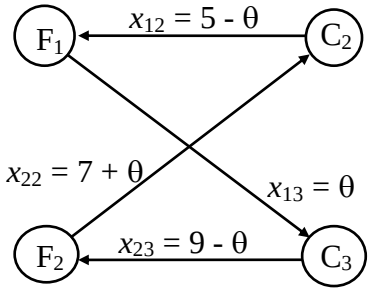
Tabelul 2.3.3

Rutele "ocupate" au fost marcate cu asteriscuri!

Deoarece există şi mărimi Δ_{ij} pozitive, soluţia testată nu este optimă. Considerăm ruta neocupată (F_1,C_3) în care $\Delta_{13} = 1 > 0$. Adăugăm la arborele H al soluţiei $\bar{x}$ muchia {F_1,C_3}:



Rezultă ciclul:



Pe muchiile ciclului au fost puse în evidenţă componentele din soluţia variabilă $\tilde{x}$ care depind de parametrul $\theta > 0$.

Acest ciclu se poate pune în evidenţă şi în tabelul 2.3.1 al soluţiei $\bar{x}$ printr-un ***contur poligonal*** care începe din celula (1,3) şi "coteşte" în unghi drept prin celulele ocupate (2,3) , (2,2) , (1,2) - vezi tabelul 2.3.4.

10	5 - θ (-)	(+) θ	
	7 + θ (+)	9 - θ (-)	1
			18

Tabelul 2.3.4

Conturul va avea un număr *par* de "*colţuri"* deoarece acestea corespund muchiilor ciclului.

În tabelul 2.3.4 apare în fapt soluţia variabilă $\tilde{x}$ definită în (2.3.2). Costul asociat al transportului, calculat cu relaţia (2.3.3) are valoarea:

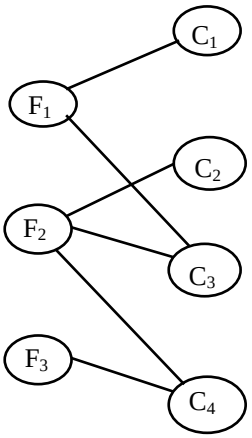
$$f(\tilde{x}) = f(\bar{x}) - \theta\Delta_{13} = 143 - 1\cdot\theta$$

Pentru determinarea comodă a lui θ_0 din (2.3.4) putem proceda astfel: *marcăm* succesiv colţurile conturului poligonal cu „+" şi „–" începând cu + în celula (1,3). Atunci θ_0 este exact *minimul* componentelor soluţiei $\bar{x}$ care sunt situate în celulele marcate cu „–": $\theta_0 = \bar{x}_{12} = 5$. Noua soluţie $\bar{x}'$ apare în Tabelul 2.3.5:

10		5	
	12	4	1
			18

Tabelul 2.3.5

şi corespunde arborelui H dedus din $H \cup \{F_1, C_3\}$ eliminând muchia $\{F_1, C_2\}$:



Invităm cititorul să repete calculele făcute în acest exemplu plecând de la soluţia din tabelul 2.3.5.

5.2.4. Algoritm de rezolvare a PTE. Convergenţă

Ca şi până acum ne vom referi la problema de transport echilibrată (PTE) al cărei model matematic a fost prezentat în secţiunea 5.1.2 şi vom presupune în continuare că *toate* soluţiile sale de bază sunt

nedegenerate. Dezvoltările teoretice din secţiunile precedente ca şi exemplele ilustrative conduc la următorul algoritm de rezolvare a PTE.

Iniţializare. Se determină printr-o metodă oarecare (vezi secţiunea 5.2.1) o soluţie admisibilă de bază de start $\bar{x} = (\bar{x}_{ij})$.

Conţinutul unei iteraţii. 1) Se asociază furnizorilor variabilele u_1, u_2,..., u_m şi consumatorilor variabilele v_1, v_2,..., v_n. Asociem fiecărei rute *ocupate* (i,j) (aceasta însemnând $\bar{x}_{ij} > 0$) o ecuaţie de forma $u_i + v_j = c_{ij}$. Se determină o soluţie particulară $(\bar{u}, \bar{v})$ a sistemului format. Pentru aceasta se acordă o valoare particulară (întotdeauna ***zero***) uneia dintre variabile (de regulă, *celei care apare de cele mai multe ori*); valorile celorlalte variabile se determină apoi *în mod unic*.

2) (*Test de optimalitate*) Se calculează mărimile $\Delta_{ij} = \bar{u}_i + \bar{v}_j - c_{ij}$ pentru toate rutele *neocupate* (adică acolo unde $\bar{x}_{ij} = 0$). Dacă toţi $\Delta_{ij} \le 0$ soluţia curentă $\bar{x}$ este *optimă*. În caz contrar:

3) Se identifică ruta (i_0,j_0) cu *cel mai mare* $\Delta_{i_0 j_0}$ *pozitiv*. În tabelul soluţiei $\bar{x}$ se identifică *unicul contur poligonal* care începe şi sfârşeşte în celula (i_0, j_0) şi coteşte în unghi drept numai prin celule ocupate (acest contur corespunde *ciclului* format în arborele H asociat soluţiei $\bar{x}$, după adăugarea muchiei $\{F_{i_0}, C_{j_0}\}$!). Se marchează alternativ cu ”+” şi ”–” colţurile ciclului, începând cu + din celula (i_0,j_0). Se calculează θ_0 ca fiind *minimul* componentelor $\bar{x}_{ij}$ aflate în celulele marcate cu ”–”.

4) (*Construcţia unei noi soluţii*) Se adună θ_0 la valorile $\bar{x}_{ij}$ aflate în celulele marcate cu ”+” şi se scade acelaşi θ_0 din valorile $\bar{x}_{ij}$ înscrise în celulele marcate cu ”–”. Valorile $\bar{x}_{ij}$ aflate în celulele nemarcate cu ”+” sau ”–” nu se modifică. Costul asociat soluţiei $\bar{x}'$ rezultate are valoarea:

$$f(\bar{x}') = f(\bar{x}) - \theta_0 \cdot \Delta_{i_0 j_0} \qquad (5.4.1)$$

Se revine la pasul 1) în cadrul unei noi iteraţii.

În ceea ce priveşte convergenţa algoritmului, *dacă toate soluţiile de bază ale PTE sunt nedegenerate, algoritmul descris se termină într-un număr finit de iteraţii cu determinarea unei soluţii optime.*

Într-adevăr, formula (5.4.1) arată că la fiecare iteraţie valoarea funcţiei obiectiv descreşte semnificativ. Cum numărul soluţiilor de bază admisibile este finit, algoritmul se opreşte obligatoriu într-un număr finit de paşi, ultima soluţie testată fiind optimă.

5.2.5. Degenerare

Algoritmul de rezolvare a PTE şi convergenţa acestuia au fost prezentate în ipoteza că toate soluţiile admisibile de bază ale problemei sunt **nedegenerate (***soluţia are exact m + n - 1 componente nenule***)**. Şansa ca o problemă de transport să aibe soluţii degenerate este însă foarte mare şi în plus nu avem nici un criteriu pe baza căruia să recunoaştem în prealabil existenţa acestor soluţii. Este important să observăm că:

D*acă în rezolvarea unei PTE am pornit cu o soluţie nedegenerată şi apoi toate soluţiile construite au fost de asemenea nedegenerate, procesul iterativ este necesarmente finit.*

În virtutea acestei observaţii, va fi important să ştim cum procedăm dacă soluţia de start este degenerată sau dacă degenerarea apare pe parcursul aplicării algoritmului. În principiu, evitarea degenerării se face prin “uşoara” *perturbare* a unora din datele problemei de aşa manieră încât noua problemă să aibe numai soluţii de bază nedegenerate! Soluţiile celor două probleme vor diferi “cu puţin” unele de altele astfel că, după rezolvarea problemei perturbate prin “revenire” la problema iniţială se obţine soluţia optimă a acesteia din urmă. Am considerat că este mai simplu şi mai bine să explicăm *tehnica de perturbare* pe câteva exemple particulare; în orice altă situaţie similară se va proceda absolut analog.

Exemplul 2.5.1 Considerăm problema de transport echilibrată definită de datele din tabelul 2.5.1:

	C_1	C_2	C_3	C_4	Disponibil
F_1	4	2	5	4	100
F_2	6	7	3	8	100
F_3	3	5	4	5	100
Necesar	110	90	50	50	300

Tabelul 2.5.1

În tabelul 2.5.2 este dată soluţia generată prin metoda diferenţelor maxime:

10 (4)	90 (1)		
		50 (2)	50 (5)
100 (3)			

Tabelul 2.5.2

Soluţia este *degenerată* deoarece are $5 < 6 = 3 + 4 - 1$ componente nenule. Această situaţie se datorează faptului că la alocarea a 4-a disponibilul curent al furnizorului F_1 a fost egal cu necesarul curent al consumatorului C_1 (=10) şi ca urmare, după efectuarea alocării, atât rândul 1 cât şi coloana 1 au fost blocate! Pentru a obţine o soluţie nedegenerată *perturbăm puţin* datele problemei originale în următorul mod. Fie $\varepsilon > 0$ un număr foarte mic.

- Mărim cu ε necesarul consumatorului C_1 acesta devenind b_1=110+ ε.
- Pentru reechilibrarea problemei mărim cu acelaşi ε disponibilul unui furnizor *activ* (adică cu disponibilul curent nenul) *altul* decît F_1; de exemplu modificăm disponibilul lui F_2: $a_2 = 100 + \varepsilon$.

Reluăm alocarea a 4-a:

$$x_{11} = \min(10 + \varepsilon, 10) = 10$$

Actualizăm:

$$b_1 \leftarrow \varepsilon \qquad a_1 \leftarrow 0$$

Continuând aplicarea metodei diferenţelor maxime, rezultă în final soluţia nedegenerată:

10 (4)	90 (1)		
ε (5)		50 (2)	50 (6)
100 (3)			

Tabelul 2.5.3

dar, pentru problema *perturbată*!!

Aplicăm acestei soluţii algoritmul de rezolvare a PTE:

	$v_1 = 0$	$v_2 = -2$	$v_3 = -3$	$v_4 = 2$
$u_1 = 4$	*	*	-4	2
$u_2 = 6$	*	-3	*	*
$u_3 = 3$	*	-4	-4	0

⇒

(−) 10	90		(+)
(+) ε		50	(−) 50
100			

Tabelele 2.5.4 - 2.5.5

"Redistribuim" în colţurile ciclului valoarea:

$\theta_0 = \min\{x_{11} = 10, x_{24} = 50\} = 10 \Rightarrow$

Obţinem soluţia:

	90		10
10 + ε		50	40
100			

Tabelul 2.5.6

Pentru ε = 0 se găseşte o soluţie nedegenerată a problemei originale căreia îi aplicăm, în continuare algoritmul:

	v_1=6	v_2=6	v_3=3	v_4=8
u_1 = -4		90		10
u_2=0	10		50	40
u_3=-3	100			

$\Rightarrow$

-2	*	-6	*
*	-1	*	*
*	-2	-4	0

Tabelele 2.5.7 - 2.5.8

Noua soluţie, notată $\bar{x}$, este optimă. Deoarece $\Delta_{34} = 0$, problema mai are o soluţie optimă de bază $\bar{x}'$ care se obţine folosind conturul poligonal asociat în tabelul 2.5.7 celulei (3,4):

	90		10
50		50	
60			40

Tabelul 2.5.9

În acord cu teoria generală a programării liniare, problema dată va avea o infinitate de soluţii optime de forma:

$x = \alpha \bar{x} + \beta \bar{x}'$
unde $\alpha + \beta = 1$
$\alpha \geq 0$, $\beta \geq 0$

	90		10
$10\alpha + 50\beta$		50	40α
$100\alpha + 60\beta$			40β

Tabelul 2.5.10

Exempul 2.5.2 Să rezolvăm acum problema:

	C_1	C_2	C_3	C_4	Disponibil
F_1	4	2	5	4	110
F_2	6	7	3	8	80
F_3	3	5	4	5	90
Necesar	120	90	50	20	280

Tabelul 2.5.11

pornind de la soluţia de bază nedegenerată din tabelul 2.5.12, determinată prin metoda diferenţelor maxime.

20 (4) ⊖	90 (1)		⊕
10 (5) ⊕		50 (2)	20 (6) ⊖
90 (3)			

Tabelul 2.5.12

$\Rightarrow$

	90		20
30		50	
90			

Tabelul 2.5.13

Propunem cititorului să verifice că această soluţie nu este optimă şi că $\Delta_{14} = 2 > 0$. În tabelul 2.5.12 este indicat şi conturul poligonal asociat rutei (1,4). Marcând succesiv colţurile conturului cu „+” şi „– ” se constată că minimul θ_0 al valorilor numerice din celulele marcate cu „– ” nu este unic: $\theta_0 = x_{11} = x_{24} = 20$.

Aceasta face ca noua soluţie, indicată în tabelul 2.5.13, să fie *degenerată*. Pentru a evita degenerarea, modificăm puţin valoarea uneia din variabilele x_{11} sau x_{24}; luăm de exemplu: $x_{11} = 20 + \varepsilon$, ceea ce înseamnă să considerăm $a_1 = 110 + \varepsilon$ şi $b_1 = 120 + \varepsilon$. De această dată minimul $\theta_0 = x_{24} = 20$ este unic, astfel că după “redistribuirea” sa în colţurile conturului poligonal indicat se obţine soluţia nedegenerată din tabelul 2.5.14:

ε	90		20
30		50	
90			

Tabelul 2.5.14

$\Rightarrow$

	$v_1 = 0$	$v_2 = -2$	$v_3 = -3$	$v_4 = 0$
$u_1 = 4$	*	*	-4	*
$u_2 = 6$	*	-3	*	-2
$u_3 = 3$	*	-4	-4	-2

Tabelul 2.5.15

dar, a problemei modificate! Din tabelul alăturat 2.5.15 rezultă că soluţia construită este optimă. Luând $\varepsilon = 0$ obţinem soluţia optimă a problemei iniţiale care este deja afişată în tabelul 2.5.13.

5.3. Variante ale problemei de transport

În dezvoltările teoretice din secţiunile precedente condiţia de echilibru (5.2.5) a fost esenţială. În foarte multe contexte practice însă, această condiţie nu este îndeplinită De asemenea, este posibil ca unele ipoteze sau constante ale problemei de transport să se modifice de la o perioadă la alta antrenând schimbări de amploare mai mică sau mai mare în soluţia optimă. În fine, nu puţine sunt situaţiile concrete ce nu implică "transporturi" în sensul strict al cuvîntului, dar care pot fi modelate ca probleme de transport.

5.3.1 Probleme de transport neechilibrate

a) În cazul în care în problema generală de transport (secţiunea 5.1.2) totalul cantităţilor disponibile la furnizori întrece totalul cererilor consumatorilor:

$$\sum_{i=1}^{m} a_i > \sum_{j=i}^{n} b_j$$

ne putem reduce la o problemă de transport echilibrată introducând un ***consumator fictiv*** C_{n+1} a cărui "cerere" să fie egală cu *excesul de disponibil*:

$$b_{n+1} = \sum_{i=1}^{m} a_i - \sum_{j=1}^{n} b_j$$

Costurile unitare de transport de la furnizorii reali către C_{n+1} se iau egale cu *zero*. După rezolvarea problemei echilibrate, *cantităţile "livrate" consumatorului fictiv se vor interpreta drept cantităţi rămase în stocurile furnizorilor.*

b) Dacă totalul cantităţilor disponibile este mai mic decât cererea totală:

$$\sum_{i=1}^{m} a_i < \sum_{j=i}^{n} b_j$$

problema de transport, aşa cum a fost ea definită în secţiunea 1.2, este ***incompatibilă*** (vezi inegalitatea (5.2.1)). *Putem încerca o rezolvare parţială a cererilor*, introducând un ***furnizor fictiv*** F_{m+1} al cărui "disponibil" să fie egal cu *cererea neacoperită*:

$$a_{m+1} = \sum_{j=1}^{n} b_j - \sum_{i=1}^{m} a_i$$

Din nou, costurile unitare de transport pe rutele ce leagă acest "furnizor" de consumatorii reali se iau egale cu *zero*. Obţinem o problemă de transport echilibrată, în a cărei soluţie optimă, *cantităţile "livrate" de furnizorul fictiv se vor interpreta drept cereri neacoperite.*

Mult mai aproape de realitate ni se pare următoarea abordare. Să presupunem că furnizorii F_1, F_2,..., F_m sunt bazine carbonifere iar consumatorii C_1, C_2,..., C_n sunt termocentrale. Să admitem că într-o perioadă normală de lucru (să zicem o lună) cantitatea de cărbune Q necesară termocentralelor, reprezentată prin suma $b_1+b_2+...+b_n$ a cererilor este egală cu cantitatea totală de cărbune posibil de livrat de către centrele miniere, cantitate reprezentată prin suma $a_1+a_2+...+a_m$. Cunoscând costurile unitare de transport ale cărbunelui pe calea ferată sau cu alte mijloace (naval, auto) se poate determina un program de satisfacere a necesarului de cărbune al termocentralelor care să implice un cost total minim. Să presupunem că în luna următoare sunt anunţate o serie de acţiuni greviste la unele centre miniere. Este posibil ca nu toate sindicatele miniere din acelaşi bazin carbonifer să adere la grevă ceea ce face ca producţia de cărbune să scadă într-o măsură mai mică sau mai mare. Fie $a_1', a_2', \ldots, a_m'$ producţiile lunare în condiţii de criză şi Q'< Q suma acestora.

Într-o asemenea situaţie critică este mai logic ca fiecare termocentrală să primească o parte proporţională cu cererea sa în condiţii normale de aprovizionare, adică:

$$\frac{b_1'}{b_1} = \frac{b_2'}{b_2} = \ldots = \frac{b_n'}{b_n},$$

$b_1', b_2', \ldots, b_n'$ fiind cantităţile ce urmează a fi primite în situaţia de criză. Noile cantităţi se pot deduce uşor, observând că fiecare raport $\frac{b_j'}{b_j}$ este egal cu:

$$\frac{b_1' + b_2' + \cdots + b_n'}{b_1 + b_2 + \cdots + b_n} = \frac{a_1' + a_2' + \cdots a_m'}{a_1 + a_2 + \cdots + a_m} = \frac{Q'}{Q}$$

de unde: $b_j' = \frac{Q'}{Q} b_j \quad j = 1, \ldots, n$.

O dată stabilite cantităţile b_i' avem o problemă de transport echilibrată pe care o rezolvăm cu algoritmul descris.

Exemplul 3.1.1 Patru termocentrale C_1, C_2, C_3, C_4 se aprovizionează cu cărbune de la trei mine F_1, F_2, F_3. Necesarul lunar al termocentralelor, producţiile lunare ale minelor şi costurile transportului unei unităţi fizice de cărbune (1000 t.) pe diferitele rute sunt date în tabelul 3.1.1

	C_1	C_2	C_3	C_4	Disponibil
F_1	3	2	1	5	120
F_2	4	3	7	2	180
F_3	3	3	5	6	200
Necesar	100	110	140	150	500

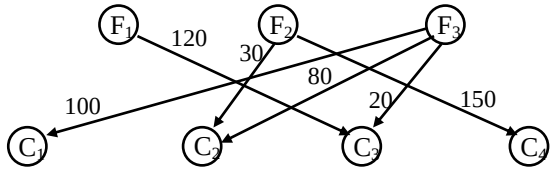
Tabelul 3.1.1

Cu metoda diferenţelor maxime, se obţine direct programul optim de aprovizionare din tabelul 3.1.2. Rutele utilizate în acest program sunt evidenţiate în figura alăturată.

		120	
	30		150
100	80	20	

Tabelul 3.1.2

$\Rightarrow$



Costul asigurării transporturilor din program se ridică la 1150 u.m.

Pentru luna următoare unele sindicate miniere preconizează o serie de acţiuni greviste. Ca urmare a acestora se estimează că producţia totală de cărbune va scade cu 30% fiind repartizată astfel: 100 mii t. la mina F_1 şi numai 120, respectiv 130 mii t. la minele F_2 şi F_3 deci un total de 350 mii t. faţă de o cerere de 500 mii t.

Problema repartizării producţiei diminuate se poate pune în două moduri:

- *urmărind în exclusivitate criteriul minimizării cheltuielilor de transport.* "Reechilibrăm" problema prin introducerea unei "mine" fictive F_4 a cărei producţie lunară să fie egală cu cantitatea cu care s-a diminuat producţia curentă a minelor reale, adică 150 mii t. Neexistând transporturi efective între F_4 şi C_1, C_2, C_3, C_4 costurile unitare de transport pe rutele corespunzătoare vor fi luate, firesc, egale cu zero. Vezi tabelul 3.1.3.

	C_1	C_2	C_3	C_4	Disponibil
F_1	3	2	1	5	100
F_2	4	3	7	2	120
F_3	3	3	5	6	130
F_4	0	0	0	0	150
Necesar	100	110	140	150	500

Tabelul 3.1.3

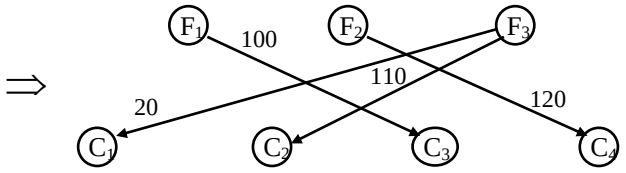
Rezultă două soluţii optime indicate în tabelele 3.1.4 şi 3.1.5. Costul de transport aferent este de 730 u.m.

În prima variantă numai cererea termocentralei C_2 este integral acoperită, C_1 primind numai 20%, C_3 numai 71% iar C_4 numai 80% din necesarul curent. în a doua variantă C_1 primeşte cantitatea normală, C_2

numai 27% iar C_3 şi C_4 procentele anterioare. După cum se vede reducerea cu 30% a producţiei normale este repartizată foarte diferit pe consumatori.

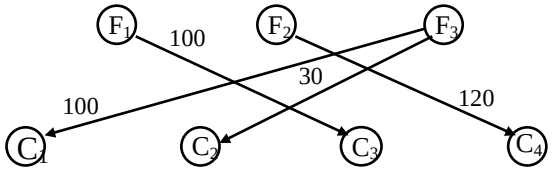
	C_1	C_2	C_3	C_4
F_1			100	
F_2				120
F_3	20	110		
F_4	80		40	30

Tabelul 3.1.4



	C_1	C_2	C_3	C_4
F_1			100	
F_2				120
F_3	100	30		
F_4		80	40	30

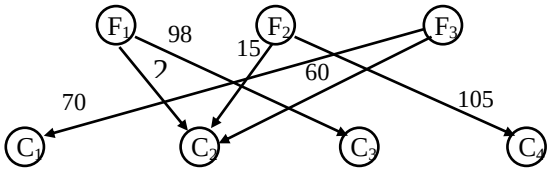
Tabelul 3.1.5



- *repartizând producţia diminuată proporţional cu cererile normale.*

Producţia diminuată reprezintă 70% din cea normală, astfel că termocentralele C_1, C_2, C_3, C_4 ar urma să primească 100·0,7=70 mii t, 110·0,7=77 mii t, 140·0,7=98 mii t, respectiv 150·0,7=105 mii t.
Rezolvând problema echilibrată rezultată obţinem soluţia:

	2	98	
	15		105
70	60		



Exemplul 3.1.2 Datorită dezvoltării şi extinderii capacităţilor de producţie, conducerea firmei X a decis să facă noi angajări în fiecare din cele cinci fabrici ale sale, conform datelor din următorul tabel:

Fabrica	I	II	III	IV	V	Total
Nr. de noi angajaţi	45	74	50	82	63	314

Noul personal este recrutat din 3 oraşe mari aflate în zonă, prin intermediul unor agenţii specializate. Contactând aceste agenţii, firma a găsit convenabile următoarele oferte:

Agenţia din oraşul	A	B	C	Total
Număr de oferte convenabile ptr. firmă	120	100	154	374

Fabricile sunt situate într-o zonă rurală aşa că, în discuţiile cu sindicatele interesate, firma a convenit să suporte cheltuielile zilnice de întoarcere de la locul de muncă la oraş, la toţi angajaţii noi, cheltuieli evaluate la 12 u.m. pe persoană × km. Distanţele în km dintre fabrici şi oraşe sunt indicate în următorul tabel:

	I	II	III	IV	V
A	6	2	2	6	3
B	14	9	4	5	3
C	10	4	11	3	4

Pentru început conducerea firmei este interesată în a cunoaşte câte persoane ar putea fi angajate astfel încât cheltuielile totale de transport să fie cât mai mici cu putinţă.

Întrucât disponibilul de personal este mai mare decât cererea, vom introduce o "fabrică fictivă" VI a cărei cerere să fie de 374-314 = 60 noi angajaţi. Obţinem o problemă echilibrată de transport cu datele din tabelul 3.1.6 al cărei obiectiv este *minimizarea numărului total de persoane× km.*

Deoarece pe rutele care leagă oraşele A, B, C de "fabrica" VI nu vor avea loc transporturi de personal, costurile unitare au fost luate egale cu zero.

	I	II	III	IV	V	VI	Disponibil
A	6	2	2	6	3	0	120
B	14	9	4	5	3	0	100
C	10	4	11	3	4	0	154
Cerere	45	74	50	82	63	60	374

Tabelul 3.1.6

Aplicând algoritmul de rezolvare descris în secţiunea 2.4 se obţine următorul program posibil de angajări (vezi tabelul 3.1.7).Toţi cei 120 de candidaţi din oraşul A vor fi angajaţi: 45 la fabrica I, 62 la fabrica II şi restul la fabrica III. la fel, candidaţii din B vor fi angajaţi în totalitate: 37 la fabrica III şi 63 la fabrica V. Din C vor fi acceptate numai 94 de oferte din cele 154 disponibile adică 61%. Rezultă un total (minim) de 1051 oameni × km transportaţi pentru care firma trebuie să plătească zilnic 12612 u.m.

	I	II	III	IV	V	VI
A	45	62	13			
B			37		63	
C		12		82		60
	45	74	50	82	63	

Tabelul 3.1.7

Conducerea firmei este de părere că adoptarea acestui program ar creea o imagine nefavorabilă firmei pe piaţa forţei de muncă prin "discriminarea" potenţialilor lucrători din C faţă de cei din A sau B şi decide să examineze şi alte variante. Astfel, pentru a nu apare ca "incorectă" faţă de candidaţii potenţiali dintr-un oraş sau altul, s-a decis ca surplusul de 60 de oferte ce nu vor putea fi acceptate să fie repartizat în mod egal între cele trei oraşe, adică 20 de fiecare. Firma doreşte să ştie care va fi efectul acestei hotărâri asupra cheltuielilor cu transportul noilor angajaţi.

Reluăm problema fixând numărul de oferte acceptabile la 120 - 20 = 100 pentru oraşul A, 100 - 20 = 80 pentru B şi 154 - 20 = 134 pentru C (total 314). Rezultă soluţia din tabelul 3.1.8.

	I	II	III	IV	V
A	45	22	33		
B			17		63
C		52		82	

Tabelul 3.1.8

Conform acesteia, numărul total de persoane×km transportaţi va creşte la 1087, implicând cheltuieli zilnice în valoare de 13044 u.m., cu 3,43 % mai mari decât în varianta studiată anterior. Noul program satisface oferta de forţă de muncă în proporţie de 83,3 % pentru A, 80 % pentru B şi 87 % pentru C.

Plecând de la ultima soluţie, conducerea firmei doreşte să cunoască ce implicaţii ar putea avea asupra cheltuielilor de transport satisfacerea ofertelor *în aceeaşi proporţie.*

Notând cu a_1, a_2, a_3 volumul ofertelor acceptabile din A, B, repectiv C este necesar ca:

$$\frac{a_1}{120} = \frac{a_2}{100} = \frac{a_3}{154} = \frac{a_1 + a_2 + a_3}{120 + 100 + 154} = \frac{314}{374}$$

din care rezultă: $a_1 = 101$, $a_2 = 84$, $a_3 = 129$. Cu noile date se obţine programul:

	I	II	III	IV	V
A	45	27	29		
B			21		63
C		47		82	

Tabelul 3.1.9

Soluţia găsită implică 1089 oameni × km de transportat zilnic la un cost de 13068 u.m., cu 3,6 % mai mare decât în prima variantă.

Fireşte, în adoptarea deciziei asupra variantei finale a programului de noi angajări, conducerea firmei poate să ţină seama şi de alte cerinţe care nu au fost avute în vedere în studiul întreprins. În consecinţă, soluţiile sintetizate în tabelele 3.1.7 , 3.1.8 şi 3.1.9 trebuie considerate ca simple "scenarii" menite să ajute factorii decizionali în luarea unei hotărâri cât mai bune!

5.3.2. Blocarea unor rute

Până în prezent am admis că orice rută dintre un furnizor şi un consumator poate fi utilizată la un cost de transport mai mic sau mai mare. Sunt cazuri în care, din diferite motive una sau mai multe rute nu pot fi utilizate. "Blocarea" acestor rute se va face prin introducerea unor costuri unitare de transport foarte mari. Concret, dacă ruta (F_i,C_j) nu mai poate fi folosită vom lua c_{ij} = M unde M este o constantă pozitivă foarte mare.

Exemplul 3.2.1 În exemplul 3.1.1 am determinat programul lunar normal de aprovizionare cu cărbune al celor patru termocentrale în ipoteza că toate rutele erau disponibile (tabelul 3.1.2). Evident, acest program nu va suferi nici o modificare în cazul în care se blochează o rută ce nu era prevăzută a fi utilizată. Să presupunem că în luna următoare ruta (F_3,C_3) se va închide temporar din cauza unor lucrări de modernizare. În acest fel, mina F_3 nu mai poate aproviziona direct termocentrala C_3. Pentru a determina schimbările din programul actual cauzate de această întrerupere reevaluăm mărimile Δ_{ij} luând de astă dată în calcul c_{33} = M » 0.

	v_1= 3	v_2=3	v_3=M	v_4=2
u_1=1-M			120	
u_2 = 0		30		150
u_3 = 0	100	80	20	

$\Rightarrow$

1-M	2-M	*	-2-M
-1	*	M-7	*
*	*	*	-4

Tabelele 3.2.1 - 3.2.2

Deoarece Δ_{23} = M - 7 > 0, soluţia curentă nu mai este optimă; ea se îmbunătăţeşte folosind conturul poligonal indicat.

Noul program de transport, pus în evidenţă în tabelul 3.2.3 nu mai utilizează ruta blocată (F_3,C_3) şi ca urmare costul său creşte, ajungând la 1190 u.m.

		120	
	10	20	150
100	100		

Tabelul 3.2.3