ASUPRA UNOR MODELE ŞI ALGORITMI PENTRU PLANIFICAREA ŞI PROGRAMAREA PRODUCŢIEI

dr. Iulian Mircea
mirceaiulian91@yahoo.com
Academia de Studii Economice, ASE Bucureşti

dr. Mihaela Covrig
mihaela_covrig@yahoo.com
Academia de Studii Economice, ASE Bucureşti

dr. Emil Ciobanu
ciobanuemil@ymail.com
Universitatea Europeană „Drăgan" din Lugoj

dr. Vladimir Florian
vladimir@ici.ro
Institutul Naţional de Cercetare-Dezvoltare în Informaticã, ICI Bucureşti

dr. Radu R. Şerban
radu_ser@yahoo.com
Universitatea „Spiru Haret" Bucureşti

Rezumat: În lucrare sunt prezentate câteva modele şi algoritmi pentru planificarea şi programarea producţiei integrate. Planificarea pe termen mediu necesită agregarea, care poate fi realizată în raport cu timpul, resursele şi activităţile productive. Agregarea este în mare măsură o tehnică folosită pentru reducerea complexităţii calculului în problemele de optimizare combinatorială. Noi ilustrăm meritele unei organizaţii virtuale folosind arhitectura unui sistem de fabricaţie distribuit. De asemenea, prezentăm algoritmi de determinare a cuplajelor maximale cu lungimea arcelor minimă în graful ataşat problemei de alocare, pentru găsirea soluţiei problemei Dirichlet şi a problemei de potenţial-tensiune care apare într-un model de ordonare în timp a sarcinilor.

Cuvinte cheie: algoritm, plan, agregare, ordonare, alocare, transfer, graf.

Abstract: In this paper we present some models and algorithms for the integrated production planning and scheduling. Production planning on the medium-term horizon requires aggregation, which can be performed with respect to time, resources, and production activities. Aggregation is a widely used technique for reducing the computational complexity of combinatorial optimization problems. We illustrate the merits of a virtual organization with a distributed manufacturing system architecture. We present algorithms for the determination of maximal couplings with minimal arch length in the graph attached to an allocation problem, and for the determination of the solution of Dirichlet problem and of the potential-voltage problem which appear in a task scheduling model

Key words: algorithm, schedule, order, allocation, transfer, graph.

1. Introducere

Planificarea producţiei (PP) poate fi definită ca un proces de strategii stabilite pentru convertirea materialelor brute în produse finite astfel încât resursele prelucrate să fie utilizate eficient. În acest proces sunt implicaţi factori precum: forţa de muncă, nivelul stocurilor (surplus / lipsă), capacitatea şi rata producţiei (fixe / variabile), cererea previzibilă (deterministă / stochastică), orizontul de planificare (lung / mediu / scurt), planificarea organizaţională (strategică / tactică / operaţională) şi mediu de producţie. Diferitele modele de probleme PP pot fi clasificate în două categorii: modele de planificarea producţiei monolitice (MPP) şi modele de planificare a producţiei ierarhice (HPP). A doua categorie are caracteristici top-down şi deciziile sunt luate secvenţial. Mai întâi sunt luate deciziile agregate şi determinate restricţiile care se impun asupra deciziilor detaliate. Deciziile detaliate furnizează feedback-ul pentru evaluarea calităţii deciziilor agregate. Diferenţa majoră între modelele ierarhice şi monolitice este existenţa nivelurilor de structură în modelele HPP ce reduc varianţa datelor, complexitatea problemei de planificare a producţiei şi o împart într-un număr mai mare sau mai mic de subprobleme independente integrate prin câteva interfeţe. Această abordare este consistentă cu nivelul organizaţional şi deciziile ce conduc la performanţă mai bună a acestui tip de model comparativ cu altele [5]. Activitatea de replanificare joacă un rol important pentru performanţa şi robusteţea sistemelor de fabricaţie distribuită bazate pe modele dinamice. În acest context, este de dorit să se invoce o schemă de replanificare on-line în care modificările de planificare sunt executate concurent cu acţiunile obişnuite de producţie. Astfel, în contrast cu replanificarea off-line, care este făcută în timp ce acţiunile obişnuite de producţie sunt suspendate, replanificarea on-line poate fi asumată ca o acţiune cu prioritate mai mare. Într-un sistem de

fabricaţie centralizat, sarcina replanificării este tradiţional făcută să maximizeze ieşirile. Din contră, obiectivul fundamental al sarcinii de replanificare în producţia multi-agent este să furnizeze criterii flexibile, de exemplu minimizarea perturbaţiei sau echilibrarea forţei de muncă pe resursele agenţilor.

Noi presupunem că: i) fiecare task are are acţiunile constituente distribuite pe resurse pentru execuţia concurentă; ii) fiecare agent îndeplineşte funcţia de management, adică acţiunile de planificare şi migrare, pentru una sau mai multe resurse paralele pe care le controlează.

O definiţie a unei organizaţii virtuale (VO) este [10]: o reţea temporară de companii ce conlucrează la exploatarea oportunităţilor care apar şi se dezvoltă cu mare rapiditate. Managementul VO constă în organizarea, alocarea şi coordonarea resurselor şi activităţilor corespunzătoare acestora, precum şi gestionarea relaţiilor interorganizaţionale, cu scopul de a atinge obiectivele propuse în condiţiile respectării restricţiilor impuse. Mai mulţi factori conduc sau determină activităţile de business către utilizarea unei structuri de organizaţie virtuală: viteza afacerilor, costul intrării pe piaţă, ghidarea mai mult după cerinţele clienţilor, nevoia crescândă pentru globalizare. Reţelele ce colaborează devin din ce în ce mai importante în sfera business-ului regional şi global, datorită abilităţii lor de a combina (îmbina) competenţele organizaţionale. Creşterea considerabilă a tranzacţiilor şi a schimburilor între firme impune utilizarea unor modele suport de decizii şi algoritmi eficienţi pentru administrarea cât mai bună a relaţiilor din reţea [11].

Prezentăm câteva noţiuni şi rezultate din teoria grafurilor care sunt utilizate în lucrare.

Fie $G=(X,U)$ un graf finit şi conex. Notăm cu ω_A^+ submulţimea arcelor (x_i, x_j) cu $x_i \in A$ şi $x_j \notin A$, şi cu ω_A^- submulţimea arcelor (x_i, x_j) cu $x_i \notin A$ şi $x_j \in A$, unde $A \subset X$. Funcţia φ definită pe U astfel încât $\varphi(u) > 0 \ \forall u \in U$ şi  este un flux compatibil cu surplusurile $\sigma(g_1, g_2, \ldots, g_n)$ în G. Fie $\gamma=(u_1,u_2,\ldots)$ un ciclu în G cu arcele separate în două mulţimi $A_1(\gamma)$ şi $A_2(\gamma)$ după cum sensul lor coincide sau nu cu sensul de parcurgere a ciclului. Funcţia π definită pe U care îndeplineşte condiţia  este o tensiune (sau diferenţă de potenţial) în G. Considerăm o funcţie r definită pe U, unde valoarea $r_j = r(u_j)$ se numeşte rezistenţa arcului u_j. Pentru fiecare arc u_j avem $r_j \cdot \varphi_j = \pi_j$, unde $\varphi_j = \varphi(u_j)$ şi $\pi_j = \pi(u_j)$. Notând cu S matricea incidenţelor lui G, adică elementele ei sunt

$$s_j^i = \begin{cases} +1, u_j \in \omega_{x_i}^+ \\ -1, u_j \in \omega_{x_i}^- \\ 0, \text{în rest} \end{cases}$$

, avem $S \cdot \varphi = \sigma$. Dacă R este matricea diagonală cu elementele

$$r_j^i = \begin{cases} r_j, i=j \\ 0, i \neq j \end{cases}$$

, atunci $R\varphi - \varphi R = \pi$. Folosind conductanţa (capacitatea) $c_j = \frac{1}{r_j}$ a arcului u_j şi notând C matricea diagonală cu elementele $c_j^i = \begin{cases} c_j, i=j \\ 0, i \neq j \end{cases}$ avem $C \cdot \pi = \varphi$. Fiind dată tensiunea π în G, alegem un vârf oarecare x_i şi îi asociem numărul $p_i = 0$. Trecând la un vârf adiacent x_k îi atribuim numărul $p_k = p_i - \pi(x_i, x_k)$ şi continuăm până epuizăm X (este posibil fiindcă G este conex). Am definit astfel pe X o funcţie p numită potenţialul grafului (poate fi definită şi luând $p_k = p_i + \pi(x_i, x_k)$). Avem $S^T \cdot p = \pi$ (sau varianta $S^T \cdot p = -\pi$).

2. Modelarea agregată în planificarea producţiei

Planificarea producţiei pe termen mediu necesită agregarea. Agregarea poate fi făcută în raport cu timpul, resursele şi activităţile de producţie. Agregarea este în mare măsură o tehnică utilizată pentru reducerea complexităţii computaţionale a problemelor de optimizare combinatoriale. Rezolvarea problemei agregate constă în trei paşi principali. În primul pas, modelul detaliat al problemei este agregat, adică mai multe variabile ale modelului detaliat sunt înlocuite de o variabilă agregată şi mai multe restricţii de o restricţie agregată. Apoi modelul agregat rezultat este rezolvat cu algoritmi adecvaţi. În final, în pasul de dezagregare, rezultatele acceptate pe nivelul agregat sunt proiectate înapoi la nivelul detaliat.

În problema de planificare pe termen mediu, noi considerăm ferestre de timp fixe dar permitem capacităţi flexibile. Criteriile noastre de optimizare sunt minimizarea capacităţii suplimentare necesare şi minimizarea duratei procesului (sau minimizarea producţiei neterminate, engl. „Work In Progress" - *WIP*). În problema de planificare a producţiei detaliată există o mulţime de proiecte *PRO* care să fie executate în orizontul de planificare. Fiecare proiect $P \in PRO$ este caracterizat de cel mai devreme moment de start est_P şi de cel mai târziu moment de terminare lft_P. Proiectul *P* cuprinde o mulţime de cerinţe (sarcini, task-uri) nevidă T_P. Problema de planificare a proiectelor cu resurse restricţionate (RCPSP) este definită de o mulţime de taskuri *T* şi o mulţime de resurse *R*. Fiecare task $t \in T$ are o durată fixată d_t şi necesită o unitate de resurse cumulate reînnoibile $r(t) \in R$ pe toată lungimea perioadei de execuţie a lui. Capacitatea resursei *r* este notată prin $q(r)$ aceasta însemnând că *r* este capabil să proceseze cel mult $q(r)$ taskuri în acelaşi timp. Mai mult, taskurile ce aparţin aceluiaşi proiect pot fi conectate prin restricţii de precedenţă end-to-start. Restricţia de precedenţă ($t_1 \to t_2$) arată că taskul t_1 trebuie terminat înainte de startul taskului t_2, adică $end_{t_1} \le st_{t_2}$.

Soluţia unei RCPSP constă în determinarea timpilor de start admisibili st_t pentru taskuri astfel ca toate restricţiile temporale, de precedenţă şi resurse sunt verificate şi funcţia obiectiv este minimizată. Când planificarea este pe termen mediu considerăm capacităţi flexibile, adică, capacitatea normală $q(r)$ a resursei r poate fi extinsă prin capacităţi suplimentare – cum ar fi prin subcontractare sau overtime –, la un cost proporţional cu cantitatea şi durata de utilizare. Mai întâi se minimizează costul resurselor suplimentare şi WIP în runda doua, cu marginea precedentă asupra utilizării resurselor suplimentare. O ipoteză de bază este că toate materialele neprelucrate ale unui proiect sunt în stoc la startul proiectului, putând astfel utiliza următoarea formulă pentru calculul WIP: $WIP = \sum_{P \in PRO} w_P \cdot (lft_P - st_P)$, unde $st_P = \min_{t \in T_P} st_t$ este timpul de start al proiectului *P* şi w_P este un factor de pondere specific proiectului.

În problemele de planificare pe termen scurt considerăm capacităţile resursei fixe şi neextensibile. Capacităţile $q(r)$ sunt o mulţime de valori determinate dintr-un plan pe termen mediu. Se minimizează durata fabricaţiei, adică timpul maxim de terminare a taskurilor. O extensie a problemei de planificare a proiectelor cu resurse restricţionate (RCPSP) a fost propusă recent de Kis, Markus şi alţii [8], problemă numită RCPSP cu activităţi cu intensitate variabilă (RCPSVP). În extensie se lucrează cu intensitate variabilă, activităţi cu volum fix şi resurse divizibile continuu care se potrivesc cerinţelor planificării producţiei mai bine decât RCPSP, fiind destinată planificării detaliate, la nivel de produs. Un exemplu de problemă RCPSVP este dat de o mulţime finită *PRO* de proiecte, o mulţime *ACT* a activităţilor din proiecte, o mulţime *R* a resurselor reînnoibile continuu divizibile şi un graf aciclic $G = (ACT, E)$ reprezentând restricţiile de precedenţă sfârşit-început dintre activităţi. Fiecare activitate $A \in ACT$ trebuie să fie în întregime procesată între cel mai devreme moment de start

al său est_A şi cel mai târziu moment de terminare lft_A. Orizontul de timp este împărţit în unităţi discrete de timp, în fiecare unitate de timp τ se execută o parte x_τ^A a activităţii *A*, x_τ^A este intensitatea lui *A* în τ şi $\sum_\tau x_\tau^A = 1$. Mai mult, există o intensitate maximă j_A definită pentru fiecare activitate *A*. Fiecare activitate poate necesita utilizarea simultană a unor resurse, proporţional cu intensitatea ei. De aici, dacă întreaga procesare a activităţii *A* necesită un efort total g_r^A pe resursa *r*, atunci ea ocupă $g_r^A \cdot x_\tau^A$ unităţi ale acestei resurse la momentul τ. Fiecare resursă $r \in R$ are o capacitate normală de q_τ^r unităţi, ce este disponibilă şi liberă la schimbare şi o capacitate adiţional suplimentară de $\hat{q}_\tau^r$ unităţi la un cost de c_τ^r pentru fiecare unitate suplimentară utilizată. Soluţia problemei RCPSVP constă în determinarea unei intensităţi x_τ^A pentru fiecare activitate *A* şi unitate de timp τ astfel încât sunt îndeplinite constrângerile temporale şi de precedenţă, cererea de resursă nu depăşeste resursa disponibilă în orice unitate de timp şi costul total al capacităţii suplimentare utilizate este minim.

Procedura de agregare este bazată pe partiţionarea arborilor de proiect în subarbori conecşi şi unirea taskurilor care aparţin aceluiaşi subarbore într-o singură activitate agregată. Parametrii problemei agregate pot fi calculaţi după cum urmează:

- pentru fiecare restricţie de precedenţă $t_1 \to t_2$ în modelul detaliat, dacă taskurile t_1 şi t_2 sunt înserate în două activităţi diferite A_1 şi A_2, atunci este stabilită între activităţi restricţia de precedenţă $A_1 \to A_2$, altfel restricţia de precedenţă este omisă din modelul agregat. Se observă că graful precedenţelor dintre activităţi va fi tot un arbore.
- dacă notăm durata minimă a activităţii *A* prin $d(A)$ şi cu μ_A un factor de siguranţă al activităţii, atunci intensitatea maximă a acestei activităţi este $j_A = \min\left(1, \frac{\mu_A \Theta}{d(A)}\right)$.
- timpii cei mai devreme de start est_A şi timpii cei mai târzii de terminare lft_A ai activităţilor sunt stabiliţi după timpii proiectelor corespunzătoare. Se presupune că aceştia sunt multiplii întregi de lungimea unităţii de timp agregate Θ. În timpul soluţionării problemei de planificare, rezolvitorul este capabil să deducă ferestre de timp mai mari pentru activităţile care sunt conectate cu altele prin restricţii de precedenţă.
- Modelul agregat utilizează aceeaşi mulţime de resurse ca reprezentarea detaliată. Capacităţile agregate q_τ^r sunt calculate ca o integrală a capacităţilor detaliate peste unitatea de timp agregată τ, redusă printr-un factor de siguranţă al resursei μ_R. Sugerăm aplicarea capacităţilor suplimentare infinite pentru a se evita, de exemplu, problema fără soluţie.
- În sfârşit, cerinţele de resursă pe o activitate sunt sumele cerinţelor de resursă ale taskurilor conţinute, adică $q_r^A = \sum_{t \in A, r(t)=r} q_t$.

Definiţia 1. Modelul agregat al unui proiect este o partiţie a arborelui proiectului în subarbori conecşi astfel ca:

- duratele activităţilor corespunzătoare subarborilor sunt mărginite superior de $\mu_A \cdot \Theta$, care ajută la fezabilitatea abordării agregării;
- înălţimea partiţionării este minimă, pentru ca să asigure optimalitatea agregării potrivit criteriului de *WIP* minim;
- de asemenea, cu condiţiile necesare de mai sus, cardinalitatea partiţiei este minimă, astfel că modelul agregat este păstrat cât mai compact posibil.

Procedura de agregare / dezagregare prezentată este realizabilă în timp. După cele de mai sus despre aproximarea fezabilităţii, putem determina optimalitatea procedurii de agregare / dezagregare potrivit criteriului de *WIP* minim, cu omiterea restricţiilor de resursă. Pentru acest scop notăm planul agregat optimal cu Π^* şi planul detaliat optimal cu Γ^*. Există o dezagregare posibilă care va fi notată cu Γ_{Π^*}.

Teorema 1. Dacă restricţiile de resursă sunt omise, atunci

$$WIP\left(\Gamma_{\Pi^*}\right) \le WIP\left(\Gamma^*\right) + \Theta \cdot \sum_{P \in PRO} w_P .$$

Demonstraţie: Omiţând restricţiile de resursă, ambele planuri Π^* şi Γ^* se compun din activităţi/taskuri deplasate dreapta către timpul cel mai târziu de terminare al proiectului, deoarece aceasta este permisă de restricţiile de precedenţă din proiect. Se observă că Γ^* induce o partiţie a fiecărui arbore de proiect în care o componentă este mulţimea task-urilor proiectului care sunt executate într-o unitate de timp agregată. Toate aceste componente au marginea superioară Θ pentru durata activităţii corespunzătoare. Totuşi, înălţimea partiţiilor induse de Γ^* nu poate fi mai mică decât aceea a partiţiilor de înălţime minimă folosite pentru pregătirea lui Π^*. Aceasta implică faptul că timpul de start actual al proiectelor din Γ^* cade într-un segment al orizontului detaliat ce corespunde timpului de start agregat al proiectului în Π^*. De aici, din Γ_{Π^*}, fiecare proiect startează cel mult Θ mai devreme decât în Γ^*. În consecinţă, avem $WIP\left(\Gamma_{\Pi^*}\right) \le WIP\left(\Gamma^*\right) + \Theta \cdot \sum_{P \in PRO} w_P$.

Fiecare task *i* este realizat asupra unor resurse *k* (posibil mai multe) din care necesită o cantitate dată r_{ik} . Fiecare resursă *k* este disponibilă într-o cantitate dată R_k. Pentru fiecare task *i* este dată durata d_i precum şi o mulţime de restricţii de precedenţă ($i << j$), constrângând ca *i* să fie terminat înainte de startarea lui *j*. Toate datele sunt întregi. Problema este rezolvată când se găseşte o mulţime de date întregi de startare S_i astfel ca:

- pentru toate restricţiile de precedenţă cu $i << j$: $S_i + d_i \le S_j$ (1)

- pentru toate resursele *k* şi toţi timpii *t* : $\sum_{\{i \mid S_i \le t \le S_i + d_i\}} r_{ik} \le R_k$ (2)

Scopul este să se minimizeze timpul cel mai târziu de terminare ($S_i + d_i$) peste toate taskurile *i*. Soluţiile tehnice pentru planificarea şi programarea producţiei au primit atenţia cuvenită în ultimele decade. Planificarea pe termen mediu necesită agregarea. Agregarea poate fi realizată în raport cu timpul, resursele şi activităţile de producţie. Metodologia agregării a fost extensiv studiată în câmpul programării liniare. În sistemele de fabricaţie la comandă, problema programării detaliate a producţiei poate fi prinsă prin modelul clasic al problemei de programare cu restricţii (RCPSP) [1].

Se consideră că obiectele primare x_i, disponibile în cantităţile $a_i, i = \overline{1,m}$ dau prin prelucrare în unităţile z_j, $j = \overline{1,n}$, produsele $z_k, k = \overline{1,q}$ în cantităţile b_k astfel încât să se consume disponibilităţile a_i. I se asociază un graf cu vârfurile $X = \{x_1,\ldots,x_m, y_1,\ldots,y_n, z_1,\ldots,z_q\}$ şi *U* mulţimea arcelor. Vom considera că se utilizează toată gama obiectelor primare şi că se acoperă tot spectrul de produse, adică graful este conex. Vom nota cu c_j^i cantitatea din x_i consumată de y_j pentru a da pe unitatea potenţialului său cantitatea d_k^j din produsul z_k. Fixând *j* avem $\sum_i c_j^i \ge \sum_k d_k^j$. Modelul matematico-economic al repartizării materialelor (obiectelor primare) conduce la următoarea problemă (problema

Dirichlet): pentru $G=(X,U)$ un graf conex şi fără bucle, σ un vector având coordonatele în vârfurile $x_i \in X$ şi r o rezistenţă pe U, se caută un flux φ compatibil cu σ astfel ca $\varphi \cdot r$ să fie o tensiune în G. Dacă $r_j \geq 0, \forall\, j$, atunci soluţia există şi este unică.

Când G are un prim vârf şi un vârf ultim (intrarea şi ieşirea) avem $S \cdot C \cdot S^T \cdot p = \sigma$. Când nu există aceste vârfuri, ele pot fi introduse fictiv unindu-le de minimele sau maximele grafului prin arce de rezistenţă infinită.

Astfel G este reţea şi funcţia φ defineşte un flux în reţea dacă $0 \leq \varphi(u) \leq c(u), \quad \forall u \in U$ şi $\sum_{u \in \omega_x^-} \varphi(u) - \sum_{u \in \omega_x^+} \varphi(u) = 0, \ \forall x \in X$. În acest caz algoritmul Ford – Fulkerson furnizează fluxul maximal. În problema de programare detaliată a producţiei există o mulţime de proiecte care urmează să fie executate în orizontul de planificare-programare. Fiecare proiect P din mulţime este caracterizat prin cel mai devreme moment de start est_P şi timpul cel mai târziu de terminare lft_P. Proiectul P cuprinde o mulţime de taskuri nevide T_P. Mulţimea totală de taskuri este notată cu T şi fiecare task $t \in T$ are o durată fixată d_t şi necesită o unitate de resursă cumulativă reînnoibilă $r(t) \in R$ pe toată durata execuţiei sale. Capacitatea de resursă r este notată prin $q(r)$. Mai mult, taskurile ce aparţin aceluiaşi proiect pot fi conectate prin restricţii de precedenţă terminat-pornit. Restricţia de precedenţă $(t_1 \to t_2)$ arată că taskul t_1 trebuie terminat înainte de pornirea taskului t_2, adică $end_{t_1} \leq st_{t_2}$.

Reprezentăm problema după cum urmează. Un program Π este definit printr-un cvadruplu $\{X, D, C, O\}$ unde $X = \{x_i\}$ denotă o mulţime finită de variabile, fiecare variabilă x_i poate lua o valoare din domeniul său D_i, C este o mulţime de restricţii definite pe variabile, iar O reprezintă o funcţie obiectiv. Mulţimea de variabile prezente în restricţia N-ară $c(x_{i_1}, ..., x_{i_N}) \in C$, pe scurt c, este notată cu $X_c = \{x_{i_1}, ..., x_{i_N}\}$. Atunci, soluţia programului este o acoperire S a variabilelor, adică $\forall x_i \in X, \exists v_i \in D_i$, astfel ca toate restricţiile să fie satisfăcute $\forall c \in C, (v_{i_1}, ..., v_{i_N}) \in c$. Funcţia obiectiv O ataşează un număr real unei soluţii S. Dacă $O \equiv 0$, adică se caută o soluţie arbitrară, atunci Π este o problemă cu restricţii satisfăcute, altfel discutăm despre o problemă de optimizare cu restricţii, optimizarea însemnând minimizarea lui O. Problema de planificare a proiectelor cu resurse restricţionate, cum a fost descrisă mai înainte, poate fi formulată în acest cadru după cum urmează. Variabilele sunt timpii de start st_t ai task-urilor $t \in T$. Duratele d_t sunt presupuse să fie întregi şi în consecinţă domeniul iniţial al fiecărei variabile timp de start este o mulţime de întregi de la 0 la un număr suficient de mare. Câteva modele de planificare atribuie trei variabile fiecărui task: st_t, d_t şi end_t căutând să rezolve probleme unde duratele nu sunt cunoscute în avans. Aceste domenii sunt mai târziu închise prin restricţii. Funcţiile *Dmin* şi *Dmax* returnează minimul şi respectiv maximul domeniului variabilei date. Astfel, timpul cel mai devreme de start al task-ului t este $est_t = Dmin(st_t)$, timpul cel mai târziu de start al lui t este $lst_t = Dmax(st_t)$, timpul cel mai devreme de terminare al lui t este $eft_t = Dmin(st_t) + d_t$, timpul cel mai târziu de terminare al lui t este $lft_t = Dmax(st_t) + d_t$. Dacă st_t este finit, atunci timpul de sfârşit al lui t va fi notat prin $end_t = st_t + d_t$. Fereastra de timp a lui t este intervalul $[est_t, lft_t]$. Problemele de planificare reale conţin adesea subprobleme corespunzătoare producţiei mai multor produse identice sau membrii ai aceleiaşi familii de produse. Sugerăm o metodă pentru reducerea unei

părţi din deciziile de ordonare între task-urile corespunzătoare proiectelor similare. Aceasta se realizează prin inserţia restricţiilor de precedenţă în reprezentarea problemei, fiind astfel posibilă o reducere semnificativă a spaţiului de căutare. Vom introduce câteva noţiuni utilizate în soluţionarea problemelor de planificare.

Definiţia 2. Două mulţimi de taskuri P şi Q sunt izomorfe (notăm $P \equiv Q$) dacă şi numai dacă există o bijecţie între ele $\beta : P \leftrightarrow Q$ astfel ca pentru fiecare pereche de taskuri $p \in P$ şi $q \in Q$,

$\beta(p,q) \Rightarrow d_p = d_q \wedge r(p) = r(q)$ şi

$\beta(p_1,q_1) \wedge \beta(p_2,q_2) \Rightarrow (p_1 \rightarrow p_2) \Leftrightarrow (q_1 \rightarrow q_2)$.

Definiţia 3. O mulţime de taskuri $P \subseteq T$ este închisă dacă şi numai dacă ea este o componentă conexă maximală în graful de precedenţe al lui *T*.

Definiţia 4. Spunem că mulţimile închise de task-uri P şi Q sunt o pereche progresivă dacă şi numai dacă există $P^* \subseteq P$ şi $Q^* \subseteq Q$ astfel încât $P^* \equiv Q^*$ şi nu există precedenţe care vin în P^* şi care pleacă din Q^*. Această relaţie o notăm $P \overrightarrow{\rightarrow} Q$.

Mulţimile închise de cerinţe (taskuri) într-o problemă de planificare sunt definite fără ambiguităţi. La fel relaţiile perechii progresive sunt determinate neambiguu mai puţin pentru două mulţimi închise de taskuri P şi Q izomorfe, $P \equiv Q$, în care direcţia relaţiei dintre ele va fi aleasă arbitrar.

Definiţia 5. O soluţie a unei probleme de planificare este numită progresivă dacă şi numai dacă pentru fiecare pereche progresivă $P \overrightarrow{\rightarrow} Q$, execuţia lui P precede Q, în sensul formal că pentru fiecare pereche de taskuri $p \in P^*$ şi $q \in Q^*$ astfel ca $\beta(p,q)$ atunci $p --\succ q$ (restricţia de precedenţă start-to-start, adică $st_p \le st_q$). Vom referi acest tip de restricţii de precedenţă start-to-start ca restricţii progresive.

Notăm că dacă resursa $r(p)$ este unară, atunci $(p --\succ q) \Leftrightarrow (p \rightarrow q)$.

Teorema 2. Dacă problema de planificare are soluţie, atunci ea are o soluţie progresivă.

Demonstraţie: Se arată cu un algoritm care pleacă de la o soluţie arbitrară a problemei şi prin perechi de task-uri schimbate iterativ generează o soluţie progresivă. În fiecare pas al algoritmului se selectează o pereche progresivă $P \overrightarrow{\rightarrow} Q$ astfel ca nişte restricţii progresive dintre P şi Q sunt încălcate în actualul program (plan) *S*. Atunci algoritmul determină un program modificat *S'* prin schimbarea tuturor perechilor de taskuri din P şi Q care încalcă restricţiile progresive după cum urmează: $\forall\ p \in P, q \in Q : \beta(p,q) \wedge st_p^S > st_q^S \Rightarrow st_p^{S'} = st_q^S$, şi $st_q^{S'} = st_p^S$. Pentru toate celelalte taskuri $t \in T$, $st_t^{S'} = st_t^S$. S' este fezabil [4], mai mult el este atins într-un număr finit de paşi, deoarece algoritmul realizează o sortare peste mulţimile închise de taskuri în acord cu ordinea parţială definită de relaţiile perechilor progresive. De aici se arată că inserţia restricţiilor progresive potrivit definiţiei 5 prezervă consistenţa problemei de planificare-programare.

Următorul algoritm euristic [8] este utilizat pentru construirea soluţiilor parţial completabile ale problemelor de planificare. Pentru fiecare resursă $r \in R$ definim

$M_{r,\tau}^{+} = \{ t \mid t \in T^{PS} \wedge r(t) = r \wedge (st_t \le \tau \le end_t) \}$ şi

$M^{-}_{r,\tau} = \left\{ t \middle| t \notin T^{PS} \wedge r(t) = r \wedge \left(est_t \le \tau \le lft_t \right) \right\}$.

Algoritmul poate fi rulat odată în fiecare nod cercetat cu ferestrele de timp ale task-ului actual trasate la rezolvarea restricţiilor. Metoda este bazată pe algoritmul de planificare după regula priorităţii LFT [3] şi stă la baza strategiei de repartizare a timpilor. Ea a fost modificată ca să genereze planuri partial completabile atunci când este incapabilă să găsească un plan complet consistent. Algoritmul atribuie timpii de start taskurilor într-o ordine cronologică potrivit priorităţii şi adaugă task-urile procesate în T^{PS}. Pseudocodul algoritmului [7] este următorul:

1 PROCEDURE FACCPS( )
2 $U := \left\{ t \middle| t \in T : st_t \text{ nu este mărginit} \right\}$;
3 Atât timp cât ($U \neq \Phi$)
4 Alege un task $t \in U$ şi un timp de start τ utilizând regula LFT;
5 Scoate *t* din *U*;
6 Dacă $\tau + d_t \le lft_t$ atunci
7 $st_t := \tau$;
8 Adaugă *t* în T^{PS};
9 altfel
10 FAOT(*t*);
1 PROCEDURE FAOT(*t*)
2 Dacă $t \in T^{PS}$ atunci
3 Scoate *t* din T^{PS};
4 Pentru toate task-urile $t' \in T^{PS} : (t' \rightarrow t) \in C$
5 Dacă $end_{t'} > est_t$ atunci
6 FAOT(t');
7 Pentru toate task-urile $t' \in T^{PS} : (t' - - \succ t) \in C$
8 Dacă $st_{t'} > est_t$ atunci
9 FAOT(t');
10 Pentru toate task-urile $t' \in T^{PS} : r(t') = r(t)$
11 $I := [st_{t'}, end_{t'}] \cap [est_t, lft_t]$; ($I$ este intervalul în care *t* şi *t'* pot fi procesate concurent)
12 Dacă $\exists \tau \in I : \left| M^{+}_{r(t),\tau} \right| + \left| M^{-}_{r(t),\tau} \right| > q(r(t))$ atunci
13 FAOT(t');

3. Probleme de transfer şi de afectare în programarea operativă a producţiei

Aceste probleme apar frecvent la repartizarea sarcinilor pe parteneri, muncitori sau utilaje. Problema afectării constă în acţiunea de a aloca, a atribui, a pune în legătură pe x… cu y… respectându-se anumite condiţii. Soluţionarea se poate face prin algoritmul lui H. Kuhn, numit metoda ungară, sau prin metoda ''Branch and Bound'' (algoritmul lui Little care construieşte o arborescenţă cu submulţimea cuplajelor care se bucură de o anumită proprietate, se încearcă minimizarea mulţimii sumelor valorilor arcelor din cuplajele maximale din graful ce modelează problema afectării).

Am abordat problema transferurilor (o problema de programare liniară) prin metode bazate pe grafuri (avand în particular problema lui Hitchcock pentru existenţa într-un graf a unui flux

compatibil cu surplusurile date). Modelarea cu o reţea de transport permite în baza teoremei lui Gale găsirea unui flux maximal care să satureze arcele de ieşire. O altă soluţie se bazează pe teorema lui Hoffman şi pe algoritmul lui Ford-Fulkerson. Problema transferului se enunţă astfel: fiind date surplusurile $\sigma_i\,(i=1,2,\ldots,n)$ localizate în vârfurile $x_i \in X$ şi numerele $v_j\,(j=1,2,\ldots,m)$ afectate arcelor $u_j \in U$ (graful $G=(X,U)$ modelează transferul) se cere potenţialul $p=(p_1,p_2,\ldots,p_n)$ care satisface condiţiile: (i) $p_k - p_i \le v_j$ pentru $u_j=(x_i,x_k)\in U$ şi (ii) produsul scalar $<\sigma, p>$ să fie minim. Pentru $u_j=(x_i,x_k)\in U$ notăm numărul v_j cu v_k^i pentru a evidenţia vârfurile. Potenţialul p se zice că este compatibil dacă $p_k - p_i < v_k^i$. Graful G poate fi transformat într-o reţea de transport $R=(X_R,U_R)$ astfel:

- se adaugă o intrare $a \notin X$ şi o ieşire $b \notin X$;
- se introduc arcele de intrare (a,x_i), unde $x_i \in X$ are $\sigma_i>0$, notăm cu P mulţimea lor;
- se introduc arcele de ieşire (x_i,b), unde $x_i \in X$ are $\sigma_i<0$, notăm cu Q mulţimea lor;
- $X_R = X \cup \{a,b\}$, $U_R = U \cup P \cup Q$, punem pe fiecare arc $u_j \in U_R$ capacitatea

$$c(u_j)=\begin{cases} c_j=\sigma_i & \text{când } u_j \in P \\ c_j^{'}=-\sigma_i & \text{când } u_j \in Q \\ +\infty & \text{când } u_j \in U \end{cases}$$

Fie φ fluxul care extinde în R pe cel din G, unde $\varphi_j = y_k^i$ pentru $u_j=(x_i,x_k)\in U$. Prin urmare avem $\varphi_j = c_j$ când $u_j=(a,x_i)\in P$ şi $\varphi_j = c_j^{'}$ când $u_j=(x_i,b)\in Q$. Reţeaua se alege încât fluxul căutat să-i satureze arcele de intrare şi de ieşire, deci $\sum_{u_j\in P} c_j = \sum_{u_j\in Q} c_j^{'}$.

Problema transferului în forma ei canonică se enunţă astfel: fiind dată o reţea de transport cu intrarea a şi ieşirea b cu capacităţile c_j pe arcele de intrare şi $c_j^{'}$ pe arcele de ieşire, se caută un flux maximal φ astfel încât: (i) $\varphi_j \ge 0, \forall\ j$; (ii) $\varphi_j = c_j$ când $u_j \in \omega_a^+$ şi $\varphi_j = c_j^{'}$ când $u_j \in \omega_b^-$; (iii) $\sum_{u_j\in\omega_{x_i}^+} \varphi_j - \sum_{u_j\in\omega_{x_i}^-} \varphi_j = 0,\ i=1,2,\ldots,n$; (iv) $\sum_{u_j\in U} v_j\cdot\varphi_j$ să fie minimă când se cunosc numerele v_j afectate arcelor $u_j \in U$.

Prin suprimarea arcelor pentru care $p_k - p_i < v_k^i$ se obţine din R o reţea parţială R'.

Propoziţia 1. Dacă potenţialul compatibil p, în reţeaua parţială asociată R', face ca fluxul maximal să nu satureze arcele de ieşire, atunci se poate găsi în G un alt potenţial compatibil p' astfel încât $<p',\sigma> < <p,\sigma>$. Dacă în R' fluxul maximal saturează arcele de ieşire, atunci produsele scalare $<\sigma, p>$ şi $<v,\varphi>$ sunt minime.

Bazat pe acest rezultat, următorul algoritm (A1) rezolvă problema generală de transfer:

- Pasul 1. Se introduce în G o tensiune ale cărei componente verifică $\pi_j \le v_j$, pe cât posibil să egaleze numerele v_j, $j=1,2,\ldots,m$ şi se deduce din ea un potenţial p.
- Pasul 2. Se transformă G într-o reţea de transport R în care se suprimă arcele cu $\pi_j < v_j$, obţinându-se reţeaua R'.
- Pasul 3. În R' se caută fluxul maximal φ.

- Pasul 4. Dacă φ nu saturează arcele de ieşire, se îmbunătăţeşte potenţialul şi se reia de la Pasul 2, altfel în G avem că φ este soluţia problemei de transfer şi $<\sigma, p>$ este soluţia problemei conexe de potenţial.

Problema transferurilor se poate aduce la o problemă Hitchcock ce se modelează printr-un graf simplu (mulţimea vârfurilor $X \cup Y$, X şi Y disjuncte, şi mulţimea arcelor $U \subset \{(x,y) \mid x \in X, y \in Y\}$). Orice submulţime a lui U care nu omite nici un vârf al grafului se numeşte acoperire a grafului. O problemă o constituie găsirea unei acoperiri minimale (după cardinalitate, număr de arce din acoperire). Soluţiile problemei lui Hitchcock sunt printre acoperirile cu un anumit număr de arce. Problema elaborării simultane a unui număr de proiecte de către un număr de institute, cu minimizarea duratei totale de elaborare, revine la soluţionarea unei probleme Hitchcock. Pentru aceasta se utilizează teorema lui Konig-Hall privind cuplajul arcelor într-un graf. Teorema lui Konig-Ore dă numărul arcelor unui cuplaj maximal (care este egal cu numărul vârfurilor unui suport minimal; unde suport este orice submulţime de noduri astfel aleasă încât fiecare arc al grafului simplu să aibă cel puţin o extremitate în ea). Dăm câteva exemple de probleme de alocare:

1. Într-un mediu de producţie distribuită compus din m unităţi $M_1, M_2, ..., M_m$ trebuie să se execute n procese de fabricatie $L_1, L_2, ..., L_n$. Fiecare unitate poate executa numai anumite lucrări. Se cere să se facă o repartiţie a fiecărei unităţi la unul dintre procesele pentru care este specializată, astfel încât să se poată realiza cât mai multe procese. Pot să apară următoarele cazuri: a) $m=n$, b) $m>n$, c) $m<n$.

În cazul a) fiecare unitate va fi repartizată la realizarea unui proces şi fiecărui proces i se va afecta o unitate. Dacă numărul unităţilor este strict mai mare decât numărul proceselor ($m>n$), vor exista unităţi cărora nu le-a fost repartizat niciun proces. Dacă numărul unităţilor este strict mai mic decât numărul proceselor ($m<n$) şi dacă sunt îndeplinite condiţiile din teorema König-Hall atunci este posibil să repartizăm fiecare unitate la un proces şi numai la unul, pentru care este specializată. În acest caz vor exista unele procese cărora nu li se va putea afecta unităţi.

Într-adevăr, dacă este îndeplinită condiţia $|A| \le |\Gamma A|$ din teorema König-Hall, atunci $\delta_0 = 0$, deci $\max_{C \in a} |C| = |X|$, relaţie ce axprimă faptul că fiecare unitate poate fi repartizată la un proces. Dacă condiţia $|A| \le |\Gamma A|$ nu este îndeplinită, $\delta_0 > 0$, deci nu se pot repartiza proceselor decât un număr de $|X| - \delta_0$, conform teoremei König-Ore.

2. Problema alocării lucrărilor $L = \{L_1, L_2, ..., L_m\}$ partenerilor $P = \{p_1, p_2, ..., p_n\}$.

O astfel de problemă poate fi modelată după cum urmează:

Fie $G \subset L \times P$, pentru fiecare $(i, j) \in G$ ataşăm o constantă v_{ij} (cost, profit, valoare etc.) şi o variabilă y_{ij}, $y_{ij} = \begin{cases} 1, & \text{dacă partenerul } p_i \text{ realizează lucrarea } L_j, \\ 0, & \text{în caz contrar.} \end{cases}$

Modelul matematic constă în restricţiile:

$$\sum_{j=1}^{n} y_{ij} \le 1, \quad i = \overline{1,m},$$

$$\sum_{i=1}^{m} y_{ij} \le 1, \quad j = \overline{1,n},$$

$$\sum_{(i,j) \in G} y_{ij} = \min(m, n)$$

$\sum_{(i,j)\in G} y_{ij} \cdot v_{ij}$ să fie minimă

Se caută pentru y_{ij} valorile care satisfac aceste condiţii. Considerăm că relaţia definită de *G* este surjectivă. Soluţia căutată se află pe arcele unui cuplaj în graful simplu ataşat (L,P,G). Dacă $m < n$, din condiţia 3 rezultă că numărul arcelor cuplajului este *m* (cuplaj al lui *L* în *P*), deci este cuplaj maximal. Dacă $m > n$ din condiţia 3 rezultă că numărul arcelor cuplajului este *n* (cuplajul este al lui *P* în *L*), deci cuplaj maximal. Dacă $m = n$, cuplajul este al lui *L* pe *P*, deci este maximal. Aşadar, problema alocării revine la alegerea dintre cuplajele maximale existente în graf a celor pentru care suma numerelor v_{ij} ataşate arcelor lor să fie minimă. Dacă în locul condiţiei 4 avem: 4') $\sum_{(i,j)\in G} y_{ij} \cdot v_{ij}$ să fie maximă

Punând $V = \max_{(i,j)\in G} v_{ij}$ şi $t_{ij} = V - v_{ij}$, atunci $\sum_{(i,j)\in G} y_{ij} \cdot v_{ij} = V \cdot \sum_{(i,j)\in G} y_{ij} - \sum_{(i,j)\in G} y_{ij} \cdot t_{ij} =$ $V \cdot \min(m,n) - \min \sum_{(i,j)} y_{ij} \cdot t_{ij}$, deci problema de maxim se reduce la problema de minim. În cazul $m = n$ condiţiile 1 şi 2 devin egalităţi.

3. Să presupunem că într-un atelier se urmăreşte să se repartizeze muncitorii la diverse lucrări. Deoarece calificarea şi aptitudinile muncitorilor sunt diferite, se cere să se atribuie o singură lucrare fiecărui muncitor astfel ca timpul total de ore să fie minim. Se introduce o variabilă x_{ij} care ia valoarea unu dacă muncitorul M_i este repartizat la lucrarea L_j şi zero în caz contrar. Dacă se notează t_{ij} timpul necesar muncitorului M_i de a efectua lucrarea L_j modelul matematic al problemei de afectare este:

Să se minimizeze: $\sum_{i=1}^{n} \sum_{j=1}^{m} x_{ij} \cdot t_{ij}$, variabilele x_{ij} fiind supuse următoarelor restricţii:

$$\sum_{i=1}^{n} x_{ij} = 1, \qquad j=(1,\ldots,m),$$

$$\sum_{j=1}^{m} x_{ij} = 1, \qquad i=(1,\ldots,n), \quad \text{şi} \quad x_{ij} \in \{0,1\}$$

Rezolvarea problemei se poate face prin ataşarea unui graf simplu si determinarea unui cuplaj maximal cu lungimea arcelor minimă folosind următorul procedeu euristic (A2).

1). Arcele grafului ce modelează problema se împart în clase cu aceleaşi $v_{i,j}$, ordonate crescător.

2). Pornim cu un arc din prima clasă, excludem arcele adiacente cu el din clasele următoare, apoi continuăm alegerea arcelor după $v_{i,j}$ crescător şi cu excluderea arcelor adiacente din clasele următoare până formăm un cuplaj maximal W_0^S cu *n* arce.

3). Calculăm $S_s = \sum_{(i,j)\in W_0^S} v_{i,j}$ şi reţinem cuplajele cu suma minimă (ele reprezintă soluţiile problemei).

Algoritmul pentru găsirea tuturor cuplajelor maximale (A3):

Pasul 1. Se construieşte arborele conţinând cuplajele maximale ataşat matricei *Y*:

- Din nodul rădăcină (notat cu 0 şi formând nivelul 0) stabilim arcele de legătură cu nodurile nivelulului 1 (arcele (0, *j*), unde nodurile *j* verifică $y_{1,j} = 1$);

- Construcţia arborelui pe nivelele $2,3,...,n$ se face după cum urmează: pentru *i* de la 2 la *n*, pentru fiecare nod *k* de pe nivelul $i-1$ şi pentru fiecare *j* cu $y_{i,j}=1$, care nu este nod pe ramura de la rădăcina 0 la nodul *k*, se adaugă în arbore nodul *j* pe nivelul *i* şi arcul (k, j).

Pasul 2. Se parcurg toate ramurile (rădăcina 0 – nod terminal) arborelui construit la Pasul 1. Fiecare ramură de lungime *n* defineşte un cuplaj maximal format din arcele $(nivel, nod)$, începând cu nivelul 1 şi terminând cu nivelul *n*.

Aşadar, pentru rezolvarea problemei de alocare se realizează graful care modelează problema, se determină toate cuplajele maximale (cu algoritmul A3) şi se selectează cele cu suma valorilor minimă, obţinându-se astfel toate soluţiile problemei.

Prezentăm trei scurte exemple în care soluţiile se obţin utilizând algoritmii A1-A3 descrişi mai sus.

Exemplul 1. Se consideră graful dat de matricele $Y=\begin{pmatrix}1&1&1&0&1\\1&1&0&1&1\\1&1&1&0&1\\1&1&0&1&1\\1&1&1&1&0\end{pmatrix}$ şi

$V=\begin{pmatrix}5&6&5&\infty&6\\8&4&\infty&5&6\\4&7&3&\infty&4\\3&3&\infty&2&5\\3&7&3&4&\infty\end{pmatrix}$. Aplicând algoritmul A2 obţinem cuplajele maximale

$W_0^1=\begin{pmatrix}0&0&0&0&1\\0&1&0&0&0\\0&0&1&0&0\\0&0&0&1&0\\1&0&0&0&0\end{pmatrix}$, $W_0^2=\begin{pmatrix}0&0&1&0&0\\0&1&0&0&0\\0&0&0&0&1\\0&0&0&1&0\\1&0&0&0&0\end{pmatrix}$, $W_0^3=\begin{pmatrix}0&0&0&0&1\\0&1&0&0&0\\1&0&0&0&0\\0&0&0&1&0\\0&0&1&0&0\end{pmatrix}$,

$W_0^4=\begin{pmatrix}0&0&0&0&1\\0&1&0&0&0\\0&0&1&0&0\\1&0&0&0&0\\0&0&0&1&0\end{pmatrix}$ cu $S_1=S_2=18$, $S_3=19$, $S_4=20$.

Exemplul 2. Se consideră graful dat de matricele $Y=\begin{pmatrix}1&1&0&0&1\\1&1&0&1&0\\0&1&1&0&1\\1&0&1&0&1\\1&0&1&1&0\end{pmatrix}$ şi

$V=\begin{pmatrix}3&4&\infty&\infty&4\\5&2&\infty&3&\infty\\\infty&4&2&\infty&1\\2&\infty&1&\infty&1\\3&\infty&3&2&\infty\end{pmatrix}$. Aplicând algoritmul A3 obţinem cele 12 cuplaje maximale:

$W_0^1 = \{(1,1),(2,2),(3,3),(4,5),(5,4)\}$, cu $S_1 = 10$,

$W_0^2 = \{(1,1),(2,2),(3,5),(4,3),(5,4)\}$, cu $S_2 = 9$,

$W_0^3 = \{(1,1),(2,4),(3,2),(4,5),(5,3)\}$, cu $S_3 = 14$,

$W_0^4 = \{(1,2),(2,1),(3,3),(4,5),(5,4)\}$, cu $S_4 = 14$,

$W_0^5 = \{(1,2),(2,1),(3,5),(4,3),(5,4)\}$, cu $S_5 = 13$,

$W_0^6 = \{(1,2),(2,4),(3,3),(4,5),(5,1)\}$, cu $S_6 = 13$,

$W_0^7 = \{(1,2),(2,4),(3,5),(4,1),(5,3)\}$, cu $S_7 = 13$,

$W_0^8 = \{(1,2),(2,4),(3,5),(4,3),(5,1)\}$, cu $S_8 = 12$,

$W_0^9 = \{(1,5),(2,1),(3,2),(4,3),(5,4)\}$, cu $S_9 = 16$,

$W_0^{10} = \{(1,5),(2,2),(3,3),(4,1),(5,4)\}$, cu $S_{10} = 12$,

$W_0^{11} = \{(1,5),(2,4),(3,2),(4,1),(5,3)\}$, cu $S_{11} = 16$,

$W_0^{12} = \{(1,5),(2,4),(3,2),(4,3),(5,1)\}$, cu $S_{12} = 15$.

Aplicând algoritmul A2 obţinem cuplajul cu suma minimă, adică W_0^2.

Exemplul 3. Pentru problema Dirichlet considerând intrările $S = \begin{pmatrix} 1 & 0 & 1 & 1 & 0 \\ -1 & 1 & 0 & 0 & 0 \\ 0 & -1 & -1 & 0 & 1 \\ 0 & 0 & 0 & -1 & -1 \end{pmatrix}$, $r = \begin{pmatrix} 1 & \frac{1}{2} & 1 & \frac{1}{3} & 1 \end{pmatrix}$ şi $\sigma = (15 \quad 1 \quad -2 \quad -14)$ obţinem potenţialul $p = (4 \quad 3 \quad 2 \quad 0)$, tensiunea $\pi = (1 \quad 1 \quad 2 \quad 4 \quad 2)$ şi fluxul $\varphi = (1 \quad 2 \quad 2 \quad 12 \quad 2)$

4. Concluzii

Dacă în problema Dirichlet sunt date surplusurile doar în anumite vârfuri ale grafului, numărul componentelor date ale vectorilor σ şi p este egal cu puterea mulţimii vârfurilor şi rezistenţele arcelor r_j sunt nenegative pentru orice *j*, atunci problema admite o soluţie unică în grafurile care sunt reţele de transport sau pot fi aduse la forma reţelelor de transport. Mai mult, dacă rezistenţele sau conductanţele sunt de diferite semne, atunci nu se poate stabili apriori existenţa sau unicitatea soluţiei. O concluzie, la acest tip de problemă, este că potenţialele date nu sunt neapărat fixate în vârfurile în care nu se cunosc surplusurile. O altă concluzie este că fluxurile şi tensiunile nu au neapărat toate componentele întregi.

Algoritmii folosiţi pentru rezolvarea problemelor de transfer, de alocare şi problemelor conexe de potenţial-tensiune sunt utili şi în problemele de planificare a producţiei care vizează ordonarea în timp a sarcinilor. O organizaţie virtuală (O V) este creată pentru realizarea anumitor proiecte. Astfel, OV are un timp de acţiune limitat, iar obiectivele ce trebuie realizate sunt clar definite. In cadrul OV partenerii de proiect formează o echipă, dar au, totodată, un grad mai mare de autonomie în raport cu restul organizaţiei. Termenul de „echipă" este impregant cu valori pozitive. Colaborarea, cooperarea şi angajarea sunt trăsături tipice ale echipei. O bună planificare a producţiei integrate în OV scade costurile de producţie şi creşte competitivitatea pe piaţă a producţiei.

BIBLIOGRAFIE

1. **BRUCKER, P., A. DREXL, R. MOHRING, K. NEUMANN, E. PESCH**: Resource-constrained project scheduling: Notation, classification, models, and methods, European Journal of Operational Research, 112(1):3–41.
2. **CIOBANU, E**.: Creşterea eficacităţii si performanţei grupurilor, publicată în volumul ”Sesiunea Anuală de comunicări ştiinţifice a Universităţii Europene „Drăgan” din Lugoj”, Editura Nagard, Lugoj, 2008, ISBN 978-973-7690-63-0 .
3. **DAVIS, E., J. PATTERSON**: A Comparision of Heuristic and Optimum Solutions in Resource-constrained Project Scheduling, Management Science, 21:944–955, 1975.
4. **DEMEULEMEESTER, E. L., W. S**. **HERROELEN**: A Branch-and-bound Procedure for the Multiple Resource-constrained Project Scheduling Problem, Management Science, 38(12):1803–1818, 1992.
5. **GHAZANFARI, M., B. A. MURTAGH, P. MATHEW**: Planning Horizon Policies in the Modelling of Hierarchical Production Planning, The Proceedings of IASTED International Conference on Modelling, Simulation and Optimization, Gold Coast, Australia, 1996.
6. **IONESCU, T.**: Grafuri, Aplicaţii, Editura Didactică şi Pedagogică, Bucureşti, 1975.
7. **KOVACS, A**.: Novel Models and Algorithms for Integrated Production Planning and Scheduling, Ph. D. Thesis, Budapest University of Technology and Economics, 2005
8. **MARKUS, A., J. VANCZA, T. KIS, A**. **KOVACS**: Project Scheduling Approach to Production Planning, CIRP Annals – Manufacturing Technology, 52(1):359–362, 2003.
9. **MATEIA**, **N. A.:** Modele şi algoritmi pentru programarea operativă a producţiei în procese discrete, Teză de doctorat, ASE Bucureşti, 2009.
10. **MIRCEA I., R. ŞERBAN, D. TODOSE**: Linear programming models applied in selection of partners in a virtual organization, în Proc. of the 4th International Conference of ASECU – Development: Cooperation and Competitiveness, Academy of Economic Studies, 22-24 May 2008, Bucharest, 2008.
11. **MIRCEA I., R. ŞERBAN, M. COVRIG, R. R**. **ŞERBAN**: Production Planning and Scheduling: Models and Algorithms, în Proceedings of the 7th International Symposium of the Romanian Regional Science Association **– „**Territorial Cohesion: growth, convergence, competitiveness**”**, 12-13 iunie 2009, Universitatea de Nord Baia Mare**,** 2009.
12. **RENDI, D.M.**: Metode ale cercetării operaţionale: programare liniară, teoria jocurilor, teoria grafurilor, Editura Orizonturi Universitare, Timişoara, 2002.
13. **TOMESCU, I.**, Combinatorica şi teoria grafurilor, Editura Universităţii din Bucureşti, 1978.