C A P I T O L U L III

PROBLEME DE OPTIMIZARE DE DIMENSIUNI MARI

§1. Problema dimensiunii în rezolvarea efectivă a problemelor de optimizare practice

Principala cauză generatoare de dificultăţi în rezolvarea problemelor de optimizare reale este **dimensiunea**: o asemenea problemă este pur şi simplu "**prea mare**". În programarea matematică, mărimea unei probleme este o chestiune relativă, depinzând de mulţi parametri cum ar fi:

- numărul de variabile şi numărul de restricţii;
- complexitatea expresiilor funcţiei obiectiv şi a restricţiilor;
- performanţele echipamentului de calcul: hardware şi software.

Utilizarea modelelor matematice în studiul unor situaţii reale - în special din domeniul economic - a condus la programe matematice care suprasolicită şi cele mai puternice calculatoare.Din fericire, marea majoritate a problemelor de optimizare "mari" au o "structură specială" care, în programarea liniară de exemplu, înseamnă:

- **densitate mică** a constantelor numerice **nenule**;
- **gruparea elementelor nenule în blocuri** aşezate "**pe diagonală**";
- număr **foarte mare** de variabile şi **relativ puţine** restricţii sau invers, multe restricţii şi puţine variabile.

Trebuie spus că dacă un program liniar mare nu are o structură specială, sarcina colectării datelor este practic aproape imposibil de realizat; astfel pentru un program cu 10^4 restricţii şi 10^6 variabile, matricea coeficienţilor ar avea 10^{10} intrări şi în cazul unei densităţi de 100% ar necesita 10^{10} date numerice de adunat, sortat şi prelucrat!!

Pentru programele neliniare, complexitatea şi structura specială se caracterizează mult mai greu.

§2. Clasificarea metodelor de rezolvare a programelor liniare mari

În principiu metodele de rezolvare a programelor mari se împart în două categorii:

- metode **directe**: acestea specializează o procedură generală adaptând-o la specificul unei anumite clase de probleme de optimizare.

Exemplul tipic îl constituie algoritmul simplex; este ştiut că principala problemă de calcul care apare în aplicarea lui o constituie **modul de manipulare a inversei bazei curente**. În cazul unei "structuri speciale" este posibil ca dimensiunea acestei matrici să se reducă semnificativ.

Să considerăm cazul unui program liniar cu **variabile superior mărginite**:

$$\begin{cases} \sum_{j=1}^{n} a_{ij} x_j = b_i \quad i = 1,\dots,m \\ 0 \le x_j \le u_j \quad j = 1,\dots,n \\ (\max) f = \sum_{j=1}^{n} c_j x_j \end{cases}$$

Abordarea "clasică" presupunea transformarea condițiilor de limitare superioară în egalități:

$$\begin{cases} \sum_{j=1}^{n} a_{ij} x_j = b_i \quad i = 1,\dots,m \\ x_j + x_{n+j} = u_j \quad j = 1,\dots,n \\ x_j \ge 0 \quad j = 1,\dots,2n \\ (\max) f = \sum_{j=1}^{n} c_j x_j \end{cases}$$

Rezulta un program cu m+n restricții şi 2n variabile, ale cărui baze erau matrici de ordinul m+n.

Totuşi forma particulară a restricțiilor de limitare superioară a putut fi exploatată eficient într-o specializare a algoritmului simplex în care inversa bazei curente are dimensiunea egală cu numărul m al restricțiilor propriu zise.

- metode **indirecte**, bazate pe **descompunerea** problemei mari în subprobleme **mai mici**, **interconectate**. Subproblemele pot fi rezolvate **independent** (şi dacă este posibil chiar **simultan**) dar faptul că ele interacționează implică existența unui mecanism (problemă) de **coordonare**. Astfel, rezolvarea problemei originale mari se face "**la două nivele**":

  - la primul nivel - inferior - se rezolvă subproblemele; rezultatele sunt "comunicate":
  - la al doilea nivel - superior - care "analizează" aceste rezultate şi transmite nivelului inferior noi parametri.

  La nivelul unu are loc o reluare a calculelor (reoptimizare); noile rezultate sunt trimise nivelului superior care le analizează ş.a.m.d.

Important este faptul că acest proces iterativ este **convergent** în sensul că într-un număr **finit** de paşi (≡ dialoguri între cele două nivele), nivelul coordonator "anunță" găsirea soluției optime.

§3. Descompunere în programarea liniară. Principiu

Considerăm un program liniar în formă standard:

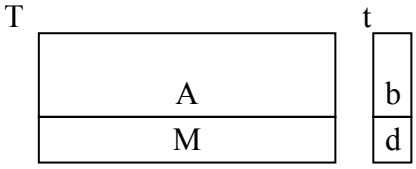
$$(P)\begin{cases} Tx = t \\ x \ge 0 \\ (\max) f = cx \end{cases} \quad \text{în care:}$$

T ≡ matrice m×n;
x ≡ coloana celor n variabile;
t ≡ coloana celor m termeni liberi;
c ≡ linia celor n coeficienți din funcția obiectiv.

Pentru moment, nu vom face nici o ipoteză asupra "mărimii" programului (P) sau asupra structurii sale. Ştim că (P) poate fi rezolvat "dintr-o dată" cu ajutorul algoritmului simplex. Ne propunem să arătăm cum poate fi rezolvat (P) prin **descompunerea** sa în mai multe subprobleme **mai mici**, **intercorelate**.

Vom împărți sistemul Tx=t al restricțiilor în două blocuri (partiționarea este deocamdată arbitrară):

- blocul Ax=b cu $m_1 < m$ restricții;
- blocul Mx=d cu $m_2 = m - m_1$ restricții



Considerăm mulțimea **soluțiilor admisibile** ale sistemului liniar Ax=b:

$$A = \{x \in R^n \mid Ax = b\ ,\ x \geq 0\}$$

Este bine cunoscut faptul că A este o mulțime **convexă** şi chiar **poliedrală** (adică o intersecție finită de semispații din R^n). Ea are un număr **finit** de **vârfuri** $v^1, v^2, ..., v^s$ care se identifică cu **soluțiile admisibile de bază** ale sistemului Ax = b.

Pentru simplitatea expunerii, vom presupune în continuare că mulțimea A este **mărginită** (această ipoteză este îndeplinită în mai toate cazurile practice).

Un rezultat clasic al analizei convexe arată că orice punct al mulțimii poliedrale şi mărginite A se scrie ca o **combinație convexă** a vârfurilor ei:

$$(\forall) x \in A \quad x = \lambda_1 v^1 + \lambda_2 v^2 + ... + \lambda_s v^s$$

unde:

$$\lambda_1 \geq 0, \lambda_2 \geq 0, ..., \lambda_s \geq 0 \text{ şi } \lambda_1 + \lambda_2 + ... + \lambda_s = 1$$

Înlocuind $x = \sum_{k=1}^{s} \lambda_k v^k$ în blocul Mx = d şi în funcția obiectiv f = cx obținem:

$$Mx = d \quad \rightarrow \quad \sum_{k=1}^{s} \lambda_k (Mv^k) = d$$

$$f = cx \quad \rightarrow \quad f = \sum_{k=1}^{s} \lambda_k (cv^k)$$

Cu notațiile:

$$Q^k = Mv^k \quad , \quad \gamma_k = cv^k \quad k = 1, ..., s$$

programul (P) se scrie echivalent:

$$(PM) \begin{cases} \sum_{k=1}^{s} \lambda_k = 1 \\ \sum_{k=1}^{s} \lambda_k Q^k = d \\ \lambda_k \geq 0 \quad k = 1, ..., s \\ (\max) f = \sum_{k=1}^{s} \gamma_k \lambda_k \end{cases} \text{ în care variabilele sunt scalarii } \lambda_1, \lambda_2, ..., \lambda_s$$

Dacă $(\lambda_k^*\ , k = 1, ..., s)$ este o soluție optimă a programului (PM) atunci $x^* = \sum_{k=1}^{s} \lambda_k^* v^k$ este o soluție optimă a programului original (P).

Programul (PM) se numeşte **program coordonator** (master program) şi are următoarele proprietăţi:

- are mai puţine restricţii decât (P): doar $1+m_2$ faţă de $m_1 + m_2$;
- are în general un număr foarte mare de variabile, câte una pentru fiecare vârf al mulţimii A şi, după cum se ştie numărul acestor vârfuri estte de obicei "impresionant";
- rezolvarea programului (PM) necesită - cel puţin la prima vedere - cunoaşterea vârfurilor $v^1,v^2,...,v^s$ fără de care nu se pot evalua colanele Q^k şi scalarii $\gamma_1,\gamma_2,...,\gamma_s$. Or, cunoaşterea apriorică a vârfurilor $v^1,v^2,...,v^s$ este o sarcină extrem de grea şi practic imposibil de făcut în mai toate cazurile!

Din fericire, rezolvarea programului (PM) nu necesită cunoaşterea apriorică a vârfurilor $v^1,v^2,...,v^s$. După cum vom vedea în secţiunea 5, pe parcursul aplicării algoritmului simplex acestui program, vârfurile mulţimii A absolut necesare în optimizare vor fi generate (calculate) "**la cerere**" prin rezolvarea unor programe liniare de forma:

$$P(u)\begin{cases} Ax = b \\ x \geq 0 \\ (\max)\tilde{f} = (c - uM)x \end{cases}$$

în care u este un vector linie ale cărui componente se stabilesc şi se modifică în funcţie de stadiul rezolvării programului (PM).

În esenţă, rezolvarea programului original (P) s-a redus la:

- rezolvarea programului coordonator (PM);

şi la:

- rezolvarea mai multor probleme de forma P(u)

toate de dimensiuni mai mici decât cele ale programului (P); vezi figura 3.1

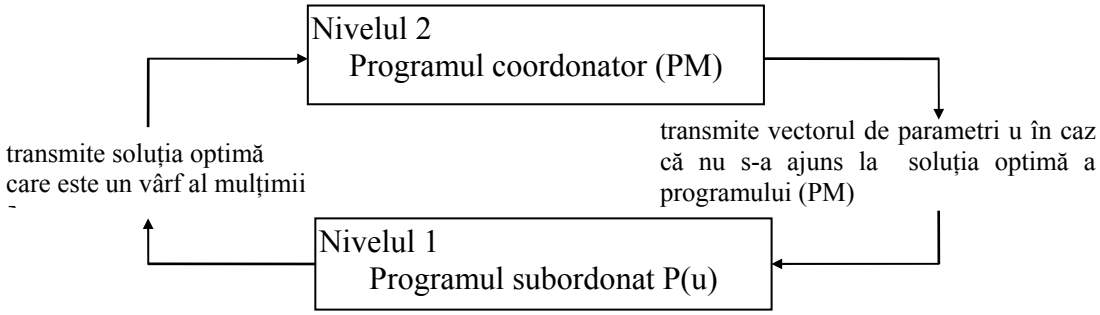


Figura 3.1

Cele de mai sus constituie esenţa **principiului de descompunere Dantzig - Wolfe**. El poate fi folosit atunci când programul (P) are un număr **foarte mare** de restricţii. Descompunerea devine şi mai atractivă în cazul în care submatricea A are o structură **diagonală**

$$A = \begin{bmatrix} A^1 & & & 0 \\ & A^2 & & \\ & & \ddots & \\ 0 & & & A^p \end{bmatrix}$$

Figura 3.2

în care $A^1, A^2, ..., A^p$ sunt matrici de diferite dimensiuni.Pentru simplificarea expunerii să presupunem că A are doar două blocuri A^1 şi A^2. Elementele constitutive ale programului (P) pot fi partiționate astfel:

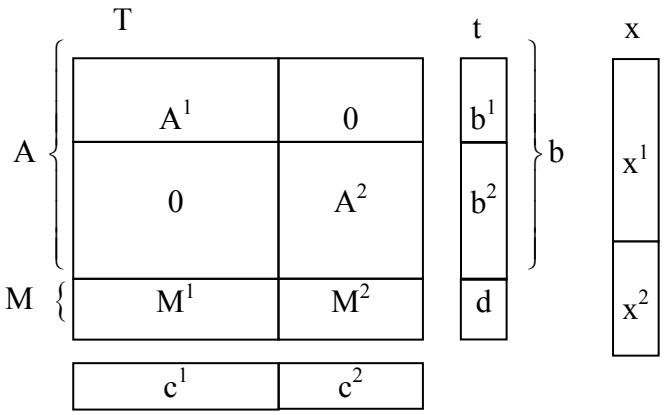


Figura 3.3

Programul (P) are deci forma:

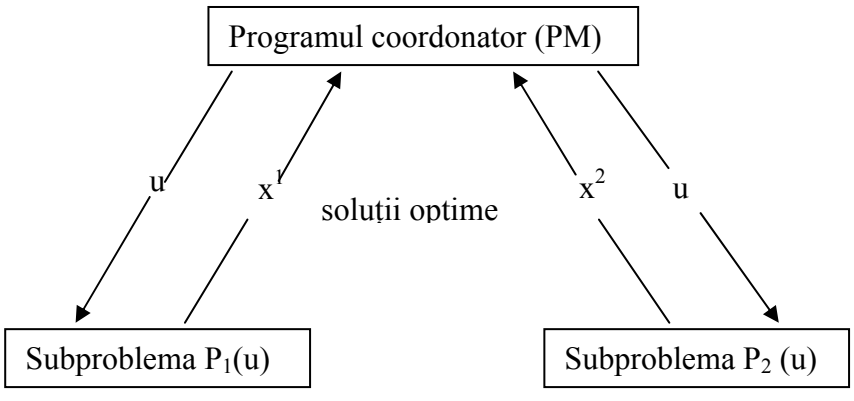
$$(P)\begin{cases} Tx = t \\ x \geq 0 \\ (\max) f = cx \end{cases} \Leftrightarrow \begin{cases} A^1 x^1 = b^1 \\ A^2 x^2 = b^2 \\ M^1 x^1 + M^2 x^2 = d \\ x^1 \geq 0\,,\; x^2 \geq 0 \\ (\max) f = c^1 x^1 + c^2 x^2 \end{cases}$$

Se observă uşor că în această situație, problema P(u) "se sparge" în două subprobleme independente (care pot fi rezolvate simultan!):

$$P_1(u)\begin{cases} A^1 x^1 = b^1 \\ x^1 \geq 0 \\ (\max) \widetilde{f}_1 = (c^1 - uM^1)x^1 \end{cases} \qquad P_2(u)\begin{cases} A^2 x^2 = b^2 \\ x^2 \geq 0 \\ (\max) \widetilde{f}_2 = (c^2 - uM^2)x^2 \end{cases}$$

Schema de rezolvare "la două nivele" a programului (P) din figura 3.1 devine:

Nivel 2



Nivel 1

Figura 3.4

§4. O interpretare economică a principiului de descompunere

Să considerăm o **economie** cu mai mulți **agenți**. Fiecare agent operează un număr de **activități** de pe urma cărora obține un **venit**. Operarea activităților implică utilizarea unor **resurse** disponibile în cantități **limitate**. Firește, obiectivul fiecărui agent este **maximizarea** venitului propriu.

Este clar că dacă fiecare agent ar deține controlul asupra tuturor resurselor necesare lui atunci maximizarea venitului **la scara întregii economii** s-ar reduce la maximizarea venitului fiecărui agent în parte.

În realitate lucrurile stau altfel.

Fiecare agent deține controlul asupra anumitor resurse: capacități proprii de producție, forța de muncă angajată, resurse financiare proprii, unele materii prime utilizate în exclusivitate.Acestea vor fi numite **resurse specifice**.

Pe lângă resursele specifice, fiecare agent utilizează şi alte resurse care nu sunt la dispoziția sa exclusivă; aceste resurse sunt procurate de pe piață, la concurență cu ceilalți agenți, datorită faptului că sunt disponibile în cantități limitate. Acestea vor fi denumite **resurse comune**.

În acest context, problema principială care se pune este de a stabili cum vor fi repartizate resursele comune între agenți astfel încât, la scara întregii economii, venitul să fie maxim.

Într-o economie **centralizată,** repartizarea resurselor comune este făcută de **stat** care indică fiecărui agent ce şi cât să producă.

Ne propunem să arătăm cum se face repartizarea comune într-o economie **descentralizată** în care autoritatea centrală nu mai deține controlul asupra acțiunilor agenților.

Ne vom situa în cazul ideal al unei economii liniare, caracterizate prin următoarele ipoteze:

- pentru fiecare activitate, consumurile de resurse şi venitul sunt direct proporționale cu nivelul la care este operată activitatea;
- nivelul de operare al unei activități poate fi reprezentat de orice număr real nenegativ;
- veniturile agenților nu se condiționează reciproc şi sunt egale cu suma veniturilor activităților fiecăruia. Venitul la scara întregii economii este suma veniturilor agenților.

Pentru simplitatea expunerii vom presupune că în economia studiată există numai doi agenți.

Introducem notațiile:

x^1, x^2 ≡ vectorii (coloană) nivelelor de operare ale activităților celor doi agenți;
b^1 , b^2 ≡ vectorii (coloană) cantităților disponibile din resursele specifice;
A^1, A^2 ≡ matricile consumurlor unitare de resurse specifice;
M^1, M^2 ≡ matricile consumurilor unitare din resursele comune;
d ≡ vectorul (coloană) cantităților disponibile de resurse comune;
c^1, c^2 ≡ vectorii (linie) veniturilor unitare corespunzătoare celor doi agenți.

Evident, nivelele de operare ale activităților agenților sunt condiționate de disponibilele de resurse specifice:

$$A^1x^1 \le b^1 \qquad A^2x^2 \le b^2 \tag{4.1}$$

şi în plus:

$$x^1 \geq 0 \qquad x^2 \geq 0 \tag{4.2}$$

Vectorii x^1, x^2 care satisfac relațiile 4.1 - 4.2 se vor numi **programe posibile de activitate**.

Un cuplu de programe posibile (x^1, x^2) devine **realizabil** numai dacă necesarul de resurse comune se încadrează în disponibilul dat adică:

$$M^1x^1 + M^2x^2 \leq d \tag{4.3}$$

Venitul total pe economie are expresia:

$$f = c^1x^1 + c^2x^2 \tag{4.4}$$

Reunind (4.1 - 4.4) obținem programul liniar:

$$(P)\begin{cases} A^1x^1 \leq b^1 \\ A^2x^2 \leq b^2 \\ M^1x^1 + M^2x^2 \leq d \\ x^1 \geq 0\,,\ x^2 \geq 0 \\ (\max) f = c^1x^1 + c^2x^2 \end{cases} \tag{4.5}$$

care modelează problema repartizării resurselor comune în vederea maximizării venitului pe întreaga economie.

Observăm că matricea programului (4.5) are **structura bloc diagonală cu restricții de cuplare**, identică cu structura pe care s-a prezentat, în secțiunea precedentă, principiul de descompunere Dantzig - Wolfe.

Din punct de vedere formal, problema repartizării resurselor comune într-o economie descentralizată înseamnă rezolvarea programului (P) în condițiile în care nici agenții nici autoritatea centrală nu au informații complete asupra acestuia. Astfel:

agentul 1 controlează (cunoaşte) b^1, A^1, M^1, c^1;
agentul 2 controlează (cunoaşte) b^2, A^2, M^2, c^2;
autoritatea centrală controlează (cunoaşte) d.

Maximizarea venitului fiecărui agent, ținând seama numai de resursele sale specifice, revine formal la rezolvarea programelor:

$$(P_1)\begin{cases} A^1x^1 \leq b^1 \\ x^1 \geq 0 \\ (\max) f_1 = c^1x^1 \end{cases} \qquad (P_2)\begin{cases} A^2x^2 \leq b^2 \\ x^2 \geq 0 \\ (\max) f_2 = c^2x^2 \end{cases}$$

dar nu rezolvă problema repartizării resurselor comune deoarece, dacă $\bar{x}^1$ şi $\bar{x}^2$ sunt soluțiile optime ale programelor din (4.6), este posibil ca:

$$M^1\bar{x}^1 + M^2\bar{x}^2 \leq d$$

În continuare vom arăta - în principiu - cum poate fi rezolvat programul (P) din (4.5) în situația în care nici autoritatea centrală şi nici agenții nu dețin informații complete asupra programului!

Vom presupune că:

- între autoritatea centrală şi agenţi există o **cooperare** în sensul unui schimb de informaţii privind "intenţiile"de acţiune;
- autoritatea centrală îşi asumă rolul de **arbitru** în următorul sens: ea "anunţă" un sistem de preţuri pe resursele comune iar agenţii iau aceste peţuri ca date şi îşi diminuează veniturile cu valoarea resurselor comune solicitate.Fie u vectorul (linie) al acestor preţuri.Atunci:
- agentul 1, pentru susţinerea unui program posibil x^1 (posibil $\equiv A^1x^1 \le b^1$, $x^1 \ge 0$),va trebui să "plătească" valoarea uM^1x^1 astfel că venitul său "net" va fi:

$$\tilde{f}_1 = c^1x^1 - uM^1x^1 = (c^1 - uM^1)x^1$$

- analog, venitul agentului 2, rezultat din programul posibil x^2 ($A^2x^2 \le b^2$, $x^2 \ge 0$), va fi:

$$\tilde{f}_2 = c^2x^2 - uM^2x^2 = (c^2 - uM^2)x^2$$

Maximizarea acestor venituri nete înseamnă rezolvarea programelor modificate:

$$P_1(u)\begin{cases} A^1x^1 \le b^1 \\ x^1 \ge 0 \\ (\max)\tilde{f}_1 = (c^1 - uM^1)x^1 \end{cases} \qquad P_2(u)\begin{cases} A^2x^2 \le b^2 \\ x^2 \ge 0 \\ (\max)\tilde{f}_2 = (c^2 - uM^2)x^2 \end{cases}$$

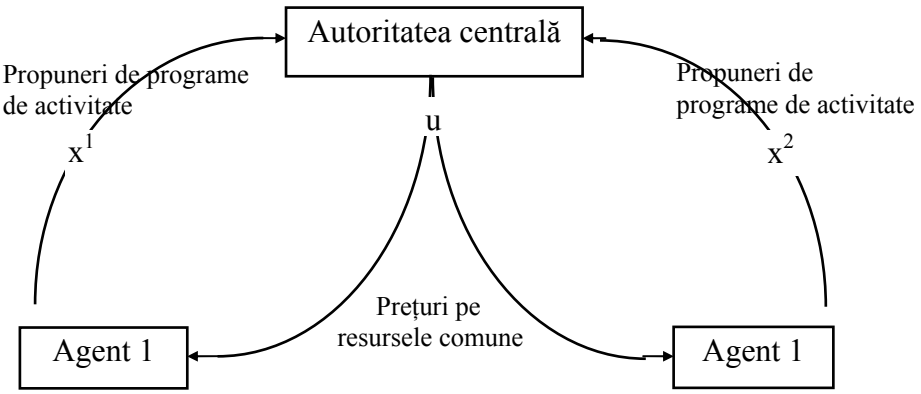
Agenţii comunică autorităţii centrale propunerile optimale $\tilde{x}^1$ şi $\tilde{x}^2$. În principiu,autoritatea centrala analizează oportunitatea luării în considerare a acestor propuneri pentru maximizarea venitului pe economie şi poate decide modificarea sistemului de preţuri, mărind preţurile resurselor comune "intens" solicitate.

Noile preţuri sunt comunicate agenţilor; aceştia vor căuta noi soluţii care să le maximizeze veniturile nete "corectate". Evident, prin creşterea preţurilor la anumite resurse comune, cererile "excesive" din aceste resurse vor fi "temperate". Formal, cele spuse înseamnă reluarea programelor $P_1(u)$, $P_2(u)$ cu u modificat!

Important este că într-un număr **finit** de asemenea "dialoguri ≡ schimburi de informaţii" între agenţi şi autoritatea centrală, vor rezulta soluţiile x^{1*} şi x^{2*} care maximizează venitul total pe economie. În general, x^{1*} şi x^{2*} nu coincid cu una sau alta dintre propunerile agenţilor (propuneri făcute în cadrul dialogului sus amintit) ci sunt **combinaţii convexe** ale acestora. Tot odată va rezulta şi un sistem u^* de preţuri pe resursele comune în raport cu care x^{1*} şi x^{2*} maximizează veniturile nete individuale ale agenţilor! Spuunem că tripletul (x^{1*}, x^{2*}, u^*) reprezintă un **echilibru** pentru economia (liniară) considerată.

Dialogul dintre autoritatea centrală şi agenţi poate fi reprezentat astfel:

Nivel 2



Nivel 1

Figura 4.1

Comparând această schemă cu cea din figura 3.4 se constată că cele spuse mai sus constituie o **interpretare economică** a principiului de descompunere Dantzig - Wolfe.

§5. Algoritmul de descompunere Dantzig - Wolfe

În această secțiune vom arăta cum se rezolvă efectiv programul principal (PM) din secțiunea 3.

Să admitem pentru moment că am cunoaşte toate constantele programului (PM). Vom aplica acestui program versiunea **revizuită** a algoritmului simplex.

Să presupunem cunoscută o **bază admisibilă** B; în raport cu această bază partiționăm vectorul λ al variabilelor:

$$\lambda = \begin{bmatrix} \lambda^B \\ \lambda^S \end{bmatrix}$$

unde λ^B este vectorul **variabilelor bazice** iar λ^S al celor **nebazice.**

Soluția $\bar{\lambda}$ asociată bazei B va avea forma:

$$\bar{\lambda} = \begin{bmatrix} \bar{\lambda}^B \\ \bar{\lambda}^S \end{bmatrix} \quad \text{cu} \quad \begin{cases} \bar{\lambda}^B = B^{-1}\begin{bmatrix} 1 \\ d \end{bmatrix} \\ \bar{\lambda}^S = 0 \end{cases}$$

B fiind presupusă admisibilă, $\bar{\lambda}^B \geq 0$. Fie:

$$\pi = \gamma^B B^{-1}$$

vectorul **multiplicatorilor simplex** asociați bazei B (γ^B este vectorul format din coeficienții γ_k ai variabilelor bazice λ_k din λ^B)

Valoarea funcției obiectiv f în soluția de bază $\bar{\lambda}$ este:

$$\bar{f} = \pi \begin{bmatrix} 1 \\ d \end{bmatrix}$$

Elementele numerice $B^{-1}, \bar{\lambda}^B, \bar{f}$ şi π sunt reunite în următorul **tabel simplex redus**:

λ^B	$\bar{\lambda}^B$	B^{-1}
f	$\bar{f}$	π

Tabelul 5.1

Din necesități care vor deveni evidente imediat, partiționăm π astfel: $\pi = (\pi_0, u)$

π_0 fiind prima componentă din π iar u reunind pe celelalte.

Testarea optimalității soluției $\bar{\lambda}$ necesită calcularea **costurilor reduse**:

$$\bar{\gamma}_k \underset{df}{=} \pi\begin{bmatrix}1\\ Q^k\end{bmatrix} - \gamma_k = [\pi_0, u]\begin{bmatrix}1\\ Q^k\end{bmatrix} - \gamma_k = \pi_0 + uQ^k - \gamma_k =$$

$$= \pi_0 + uMv^k - cv^k = \pi_0 - (c - uM)v^k \quad k = 1,\ldots,s$$

După cum este bine ştiut, dacă $\bar{\gamma}_k \geq 0 \quad k = 1,\ldots,s$ atunci $\bar{\lambda}$ este o soluție optimă a programului (PM). Pentru a vedea dacă se întâmplă acest lucru evaluăm:

$$\min_{k=1,..,s} \bar{\gamma}_k = \min_{k=1,..,s} [\pi_0 - (c - uM)v^k] = \pi_0 - \max_{k=1,..,s} (c - uM)v^k$$

Se observă că $\max_{k=1,..,s} (c - uM)v^k$ este valoarea maximă a funcției liniare $\tilde{f} = (c - uM)x$ pe vârfurile mulțimii poliedrale mărginite A.

Conform **teoremei centrale a programării liniare** [9] , evaluarea acestui maxim **nu necesită cunoaştera apriorică** a vârfurilor $v^1,\ldots,v^s$; este suficient să se rezolve programul liniar:

$$P(u)\begin{cases} Ax = b \\ x \geq 0 \\ (\max)\tilde{f} = (c - uM)x \end{cases}$$

Fie v^* o soluție optimă a programului P(u) şi $\tilde{f}^\bullet$ valoarea funcției obiectiv $\tilde{f}$ în v^*. v^* este unul din vârfurile mulțimii A şi putem scrie:

$$\tilde{f}^* = (c - uM)v^* = \max_{k=1,..,s} (c - uM)v^k$$

astfel că:

$$\min_{k=1,..,s} \bar{\gamma}_k = \pi_0 - \tilde{f}^*$$

Dacă $\pi_0 - \tilde{f}^* \geq 0$ (în fapt $\pi_0 - \tilde{f}^* = 0$!) atunci soluția $\bar{\lambda}$ este optimă.

Dacă $\pi_0 - \tilde{f}^* < 0$ se calculează $Q^* = Mv^*$, $\gamma_* = cv^*$ şi se introduce în baza curentă coloana $\begin{bmatrix}1\\ Q^*\end{bmatrix}$.

După evaluarea coloanei $B^{-1}\begin{bmatrix}1\\ Q^*\end{bmatrix}$ se determină coloana care părăseşte baza actuală şi se pivotează tabelul simplex redus curent, intrându-se într-o nouă iterație.

Din descrierea făcută rezultă clar că ameliorarea soluției admisibile de bază $\bar{\lambda}$ - presupusă dată - nu a necesitat cunoaşterea de la început a tuturor vârfurilor mulțimii A ; vârful necesar în procesul de optimizare s-a obținut rezolvând un program liniar de forma P(u) cu un vector u adecvat.

În ceeace priveşte obținerea unei soluții de bază **inițiale** $\bar{\lambda}$ pentru programul (PM), aceasta se poate obține în maniera uzuală. Se pleacă de la o **bază unitară** ale cărei coloane corespund:

- (unele) unor (eventuale) **variabile de abatere** existente în blocul Mx = d;

- (altele) unor **variabile artificiale** introduse în anumite ecuații ale blocului $Mx = d$ şi/sau în ecuația de convexitate $\sum_{k=1}^{s} \lambda_k = 1$.

În cazul în care au fost efectiv folosite variabile artificiale, într-o primă fază se va minimiza suma w a acestora. Coloanele care vor intra în bază se vor genera după schema generală de mai sus, funcția f fiind înlocuită cu funcția w care se minimizează!!

Exemplu 5.1 Se consideră o economie liniară descentralizată cu doi agenți, fiecare oprând câte o activitate. Fie x_1 şi x_2 nivelele de operare ale celor două activități. Resursele specifice controlate de către cei doi agenți limitează nivelele activităților după cum urmează:

$$x_1 \le 4 \quad , \quad x_2 \le 3$$

Susținerea activităților necesită două resurse comune R_1 şi R_2 al căror disponibil este limitat şi controlat de o "autoritate centrală". La un nivel de operare egal cu unitatea, vectorii consumurilor din resursele R_1 şi R_2 sunt: $\begin{bmatrix} 2 \\ 2 \end{bmatrix}$ pentru prima activitate şi $\begin{bmatrix} 3 \\ 1 \end{bmatrix}$ pentru a doua. Vectorul cantităților disponibile din resursele R_1 şi R_2 este $\begin{bmatrix} 9 \\ 7 \end{bmatrix}$. În fine, veniturile unitare sunt de 7 u.m. (unități monetare) în prima activitate şi de 6 u.m. în a doua.

Fiecare agent caută să obțină un venit cât mai mare, dar ei nu dețin controlul asupra resurselor comune. Pe de altă parte, economia fiind descentralizată, autoritatea centrală nu poate impune agenților nivelele la care să opereze activitățile proprii.

Obiectivul urmărit este maximizarea venitului total pe economie.

Formal, problema constă în rezolvarea programului liniar:

$$(P)\begin{cases} x_1 \le 4 \\ x_2 \le 3 \\ 2x_1 + 3x_2 \le 9 \\ 2x_1 + x_2 \le 7 \\ x_1 \ge 0 \, , \, x_2 \ge 0 \\ (\max 0f = 7x_1 + 6x_2 \end{cases}$$

în situația încare nici agenții, nici autoritatea centrală nu dețin "informații complete" asupra programului (P).

După cum am văzut, rezolvarea este posibilă prin **cooperarea** dintre agenți şi autoritatea centrală, **pe baza algoritmului de descompunere Dantzig - Wolfe**.

Partiționăm sistemul restricțiilor în două blocuri:

$$\begin{cases} x_1 \le 4 \\ x_2 \le 3 \end{cases} \Leftrightarrow \quad Ax \le b$$

$$\begin{cases} 2x_1 + 3x_2 \le 9 \\ 2x_1 + x_2 \le 7 \end{cases} \Leftrightarrow \quad Mx \le d \quad \text{unde} \quad M = \begin{bmatrix} 2 & 3 \\ 2 & 1 \end{bmatrix}, d = \begin{bmatrix} 9 \\ 7 \end{bmatrix}$$

Principiul de descompunere propune rezolvarea programului:

$$\begin{cases} \sum_{k=1}^{s} \lambda_k = 1 \\ \sum_{k=1}^{s} \lambda_k Q^k \le \begin{bmatrix} 9 \\ 7 \end{bmatrix} = d \\ \lambda_k \ge 0 \quad k = 1,\dots,s \\ (\max) f = \sum_{k=1}^{s} \gamma_k \lambda_k \end{cases}$$

în care:

$$Q^k = Mv^k \quad \gamma_k = cv^k = [7,6]v^k \quad k = 1,\dots,s$$

unde $v^1,\dots,v^s$ sunt vârfurile mulțimii poliedrale:

$$A = \left\{ x = (x_1, x_2) \middle| \begin{cases} Ax \le b \\ x \ge 0 \end{cases} \Leftrightarrow \quad 0 \le x_1 \le 4 \,.\, 0 \le x_2 \le 3 \right\}$$

Este evidentă mărginirea mulțimii A .

O dată obținută soluția optimă $\left\{ \lambda_k^* , k = 1,\dots,s \right\}$ a programului propus, soluția optimă a programului original (P) va rezulta din formula:

$$x^* = \sum_{k=1}^{s} \lambda_k^* v^k$$

Aplicăm algoritmul simplex revizuit formei standard:

$$(PM) \begin{cases} \sum_{k=1}^{s} \lambda_k = 1 \\ \sum_{k=1}^{s} \lambda_k Q^k + \begin{bmatrix} y_1 \\ y_2 \end{bmatrix} = \begin{bmatrix} 9 \\ 7 \end{bmatrix} \\ \lambda_k \ge 0 \quad k = 1,\dots,s \; ; \; y_1, y_2 \ge 0 \\ (\max) f = \sum_{k=1}^{s} \gamma_k \lambda_k \end{cases}$$

Variabilele de abatere y_1 , y_2 arată ce cantități din resursele comune R_1 , R_2 rămân nefolosite.

Programul (PM) are trei restricții şi se vede uşor că matricea sa conține coloanele unitare $\begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix}$ şi $\begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}$ corespunzătoare variabilelor y_1 , y_2. Pentru o bază unitară de start ne-ar trebui şi

coloana $\begin{bmatrix}1\\0\\0\end{bmatrix}$ care nu este tot aşa de "vizibilă". Am putea introduce o variabilă artificială în restricția de convexitate $\sum_{k=1}^{s}\lambda_k = 1$, dar putem proceda, mai simplu, şi astfel:

Se observă că vectorul nul $\begin{bmatrix}0\\0\end{bmatrix}$ este unul din vârfurile $v^1,...,v^s$ ale mulțimii A (nu întotdeauna este aşa!); putem presupune că $v^1 = \begin{bmatrix}0\\0\end{bmatrix}$. Atunci $Q^1 = Mv^1 = \begin{bmatrix}0\\0\end{bmatrix}$ şi $\gamma_1 = cv^1 = 0$.

În concluzie, matricea programului (PM) conține coloana unitară $\begin{bmatrix}1\\0\\0\end{bmatrix} = \begin{bmatrix}1\\Q^1\end{bmatrix}$.

Astfel, pentru (PM) dispunem de baza **unitară** corespunzătoare varibilelor λ_1, y_1 şi y_2. Toate aceste variabile au coeficienți nuli în funcția obiectiv aşa că :
$\gamma^B = (0,0,0)$. În consecință, vectorul multiplicatorilor simplex asociați bazei unitare indicate este:

$$\pi = \gamma^B B^{-1} = \gamma^B = [0,0,0]$$

din care rezultă $\pi_0 = 0$, u = [0,0]. Vectorul valorilor variabilelor bazice are componentele:

$$\overline{\lambda}^B = B^{-1}\begin{bmatrix}1\\9\\7\end{bmatrix} = \begin{bmatrix}1=\lambda_1\\9=y_1\\7=y_2\end{bmatrix}$$

Valoarea funcției obiectiv este:

$$\bar{f} = \gamma^B\overline{\lambda}^B = \pi\begin{bmatrix}1\\9\\7\end{bmatrix} = 0$$

Toate aceste elemente formează tabelul simplex redus de start:

(T_1)	λ_1	1	1	0	0
	y_1	9	0	1	0
	y_2	7	0	0	1
	f	0	0	0	0

Tabelul 5.2

Considerarea vârfului $v^1 = \begin{bmatrix}0\\0\end{bmatrix}$ sugerează că, la inițierea dialogului între agenți şi autoritatea centrală, se pleacă de la situația în care cele două activități nu sunt operate: $x_1 = 0$, $x_2 = 0$.
$y_1 = 9$, $y_2 = 7$ arată că resursele R_1 şi R_2 nu sunt deocamdată solicitate.

Am văzut în secțiunea 4 că vectorul u are semnificația de **sistem de prețuri pe resursele comune**.Aceste prețuri sunt "anunțate" de către autoritatea centrală agenților, care la rândul lor, îşi vor maximiza veniturile "plătind" pentru resursele comune solicitate. "Propunerile" de programe de activitate sunt "comunicate" de agenți autorității centrale. Aceasta preia propunerile şi încearcă, pe

baza lor şi a propunerilor "mai vechi", să construiască o "mixtură" care să se încadreze în disponibilul limitat de resurse comune şi să conducă la un venit total cât mai mare.

Iterația 1 Autoritatea centrală anunță sistemul de prețuri u = (0,0) , altfel spus "oferă" resursele comune "pe gratis".

Agenții rezolvă programul:

$$P(u=(0,0))\begin{cases} x_1 \le 4 \\ x_2 \le 3 \\ x_1 \ge 0\ ,\ x_2 \ge 0 \\ (\max)\widetilde{f} = 7x_1 + 6x_2 \end{cases}$$

cu soluția optimă evidentă:

$$x_1^\bullet = 4\ ,\ x_2^\bullet = 3\ ,\ \widetilde{f}^\bullet = (\max)\widetilde{f} = 7\cdot 4 + 6\cdot 3 = 46$$

pe care o trimit "ca propunere de program de activitate" autorității centrale. Deoarece:

$$\pi_0 - \widetilde{f}^\bullet = 0 - 46 < 0$$

soluția din tabelul (T_1) nu este optimă; baza curentă trebuie schimbată prin introducerea unei noi coloane din matricea programului (PM). Această coloană se generează astfel:

Vectorul $\begin{bmatrix} x_1^\bullet = 4 \\ x_2^\bullet = 3 \end{bmatrix}$ - notat în teoria premergătoare cu v^* - este un alt vârf al mulțimii A , să zicem v^2.

Calculăm:

$$v^2 = \begin{bmatrix} 4 \\ 3 \end{bmatrix} \ \Rightarrow\ Q^2 = Mv^2 = \begin{bmatrix} 2 & 3 \\ 2 & 1 \end{bmatrix}\cdot\begin{bmatrix} 4 \\ 3 \end{bmatrix} = \begin{bmatrix} 17 \\ 11 \end{bmatrix}\ ,\ \ \gamma_2 = cv^2 = [7,6]\cdot\begin{bmatrix} 4 \\ 3 \end{bmatrix} = 46$$

Coloana care intră în bază va fi: $\begin{bmatrix} 1 \\ Q^2 \end{bmatrix} = \begin{bmatrix} 1 \\ 17 \\ 11 \end{bmatrix}$.

Calculele uzuale ale unei iterații din algoritmul simplex revizuit sunt indicate mai jos

	$\begin{bmatrix} 1 \\ Q^2 \end{bmatrix}$	$\rightarrow$ 1 17 11
λ_1	1	1 0 0
y_1	9	0 1 0
y_2	7	0 0 1
f	0	0 0 0

$B^{-1}\begin{bmatrix} 1 \\ Q^2 \end{bmatrix}$

1
17
11

$-46 \leftarrow \pi_0 - \widetilde{f}^\bullet$

$\Rightarrow$ (T_2)

λ_1	8/17	1	-1/17	0
λ_2	9/17	0	1/17	0
y_2	20/17	0	-11/17	1
f	414/17	0	46/17	0

π_0 u

Tabelul 5.3

Iterația 2 Autoritatea centrală anunță noul sistem de prețuri: u = (46/17 , 0).
După cum se vede, resursa R_2 este încă oferită "pe gratis", deoarece $y_2 = 20/17$ arată că "mixtura":

$$\tilde{x} = \lambda_1 v^1 + \lambda_2 v^2 = \frac{8}{17}\begin{bmatrix}0\\0\end{bmatrix} + \frac{9}{17}\begin{bmatrix}4\\3\end{bmatrix} = \begin{bmatrix}x_1 = 36/17\\ x_2 = 21/17\end{bmatrix}$$

nu utilizează integral această resursă.

Calculăm vectorul veniturilor unitare "nete":

$$c - uM = [7,6] - \left[\frac{46}{17},0\right]\begin{bmatrix}2 & 3\\ 2 & 1\end{bmatrix} = \left[7 - \frac{92}{17}, 6 - \frac{138}{17}\right] = \left[\frac{27}{17}, \quad -\frac{36}{17}\right]$$

Agenții vor avea de rezolvat programul:

$$P(u = (\frac{46}{17},0)) \begin{cases} x_1 \leq 4 \\ x_2 \leq 3 \\ x_1 \geq 0\,,\ x_2 \geq 0 \\ (\max)\tilde{f} = \frac{27}{17}x_1 - \frac{36}{17}x_2 \end{cases}$$

a cărui soluție optimă este:

$$x_1^{\bullet} = 4\,,\ x_2^{\bullet} = 0\,,\ \tilde{f}^{\bullet} = (\max)\tilde{f} = \frac{27}{17}\cdot 4 = \frac{108}{17}$$

(La "prețurile actuale" agentul 1 are un venit net pozitiv; el își poate permite să procure resursele R_1 şi R_2 în cantitățile necesare pentru operarea activității sale la nivelul maxim posibil 4. Pentru agentul 2, resursa R_1 este "prea scumpă": oricât de mic ar fi nivelul de operare al activității proprii, "costul" resurselor R_1 şi R_2 depăşeşte venitul său "brut" astfel că, pentru agentul 2, decizia va fi "să nu facă nimic".)

Deoarece :

$$\pi_0 - \tilde{f}^{\bullet} = 0 - \frac{108}{17} < 0$$

soluția din tabelul (T_2) adică "mixtura" $\tilde{x}$ descrisă mai sus, nu este optimă.

Noua propunere a agenților $\begin{bmatrix}x_1 = 4\\ x_2 = 0\end{bmatrix}$ este un alt vârf, să zicem v^3, al mulțimii `A` , vârf care va produce coloana ce "îmbunătățeşte" baza curentă:

$$v^3 = \begin{bmatrix}4\\0\end{bmatrix} \quad \Rightarrow \quad Q^3 = Mv^3 = \begin{bmatrix}2 & 3\\ 2 & 1\end{bmatrix}\cdot\begin{bmatrix}4\\0\end{bmatrix} = \begin{bmatrix}8\\8\end{bmatrix} \quad , \quad \gamma_3 = cv^3 = [7,6]\cdot\begin{bmatrix}4\\0\end{bmatrix} = 28$$

Intră în bază coloana:

$$\begin{bmatrix} 1 \\ Q^3 \end{bmatrix} = \begin{bmatrix} 1 \\ 8 \\ 8 \end{bmatrix}$$

Pivotăm tabelul curent (T_2):

	$\begin{bmatrix} 1 \\ Q^3 \end{bmatrix}$	1	8	8	$B^{-1}\begin{bmatrix} 1 \\ Q^3 \end{bmatrix}$
λ_1	8/17	1	-1/17	0	9/17
λ_2	9/17	0	1/17	0	8/17
y_2	20/17	0	-11/17	1	48/17
f	414/17	0	0	0	$-108/17 \leftarrow \pi_0 - \tilde{f}^\bullet$

$\Rightarrow$ (T_3)

λ_1	1/4	1	1/16	-3/16
λ_2	1/3	0	1/6	-1/6
λ_3	5/12	0	-11/48	17/48
f	27	0	5/4	9/4
		π_0	u	

Tabelul 5.4

Iterația 3 Noul sistem de prețuri pe resursele comune anunțat de autoritatea centrală va fi:

$$u = (5/4 , 9/4)$$

Veniturile unitare "nete" ale agenților devin:

$$c - uM = [7,6] - \left[\frac{5}{4}, \frac{9}{4}\right]\begin{bmatrix} 2 & 3 \\ 2 & 1 \end{bmatrix} = [7,6] - [7,6] = [0,0]$$

Astfel, funcția obiectiv a programului P(u = (5/4 , 9/4)) este constantă: $\tilde{f} = (c - uM)x = 0$ şi în consecință valoarea ei maximă va fi $\tilde{f}^\bullet = 0$. Deoarece $\pi_0 - \tilde{f}^\bullet = 0 - 0 = 0$ soluția din tabelul (T_3) este soluția optimă a programului (PM). Soluția optimă a programului original (P) este "mixtura convexă" a celor trei "propuneri" v^1, v^2, v^3:

$$x^\bullet = \lambda_1^\bullet v^1 + \lambda_2^\bullet v^2 + \lambda_3^\bullet v^3 = \frac{1}{4}\begin{bmatrix} 0 \\ 0 \end{bmatrix} + \frac{1}{3}\begin{bmatrix} 4 \\ 3 \end{bmatrix} + \frac{5}{12}\begin{bmatrix} 4 \\ 0 \end{bmatrix} = \begin{bmatrix} x_1^\bullet = 3 \\ x_2^\bullet = 1 \end{bmatrix}$$

iar venitul maxim total are valoarea (max)f = 27.

§ 6. Metoda generării de coloane pentru problema croirii

După cum am văzut, principiul de descompunere Dantzig - Wolfe rezolvă un program liniar **cu multe restricții** înlocuindu-l cu un altul - numit **program principal** - cu mai puține restricții dar **cu foarte multe coloane (variabile) care nu sunt disponibile** de la început! În orice fază a rezolvării programului principal, un număr relativ mic de coloane sunt cunoscute, coloanele necesare îmbunătățirii soluțiilor intermediare fiind generate la "cerere".

Scopul acestei secțiuni este acela de a arăta cum se aplică tehnica **generării de coloane** în rezolvarea altor probleme de optimizare similare cum este de exemplu **problema croirii** introdusă în capitolul II, §2. În continuare vom vedea că această tehnică necesită luarea în considerare a unei probleme de optimizare foarte simple ca structură dar deosebit de importantă în optimizarea

combinatorială - **problema rucsacului**. Pentru această subproblemă, în secțiunea următoare, va fi prezentată o metodă specifică de rezolvare bazată pe **programarea dinamică.**

6.1 Problema croirii unidimensionale. Enunț şi model matematic

Un număr de m **repere** cu lungimile $l_1,l_2,...,l_m$ trebuiesc croite din **suporți** cu lungimea comună L în cantitățile $b_1,b_2,...,b_m$ .Obiectivul constă în satisfacerea cererilor cu un consum **minim** de suporți.

Presupunem că L şi $l_1,l_2,...,l_m$ sunt exprimate prin numere **întregi , pozitive** şi că $L > l_1 > l_2 > .. > l_m$. Am numit **rețetă de croire** o modalitate de tăiere a unui suport în repere cu lungimile cerute. Formal, o rețetă de croire se identifică cu un vector $a = (a_1,a_2,...,a_m)$ cu componente numere întregi nenegative în care a_i reprezintă numărul reperelor cu lungimea l_i rezultate din tăierea suportului. Suma lungimilor reperelor astfel obținute nu depăşeşte lungimea suportului, astfel că:

$$l_1a_1 + l_2a_2 + ... + l_ma_m \le L$$

Numărul acestor rețete este **finit** şi ordonându-le într-un fel oarecare, de exemplu **lexicografic**,obținem lista:

$$a^1,\ a^2,\ \ldots,\ a^n \quad \text{unde } a^j = (a_{1j}, a_{2j}, \ldots, a_{mj})^T$$

(pentru nevoi ulterioare rețetele vor fi scrise în coloană). Dacă notăm cu x_j numărul de suporți tăiați după rețeta a^j (x^j se mai numeşte şi **multiplicitatea** rețetei a^j) modelul matematic al problemei de croire este:

$$(P)\begin{cases} \text{Sa se determine } x_1, x_2, \ldots, x_n \text{ care minimizeaza functia:} \\ \qquad z = x_1 + x_2 + \ldots + x_n \\ \text{cu restrictiile:} \\ \qquad a^1x_1 + a^2x_2 + \ldots a^nx_n = b \iff \sum_{j=1}^{n} a_{ij}x_j = b_i \quad i = 1,\ldots,m \\ \text{si conditiile explicite impuse variabilelor:} \\ \qquad x_1, x_2, \ldots, x_n \ge 0 \text{ intregi} \end{cases}$$

Deoarece pritre rețetele $a^1,a^2,...,a^n$ se numără şi rețetele **unitare**:

$$(1,0,0,...,0)\ ,\ (0,1,0,...,0)\ ,\ ...,\ (0,0,0,...,1)$$

problema (P) are soluții admisibile (cu componente) întregi şi chiar soluție optimă.

În cazul - frecvent întâlnit în practică - în care ne limităm la utilizarea numai a aşa numitelor **rețete maximale** - adică a acelor rețete $a = (a_1,a_2,...,a_m)$ pentru care **restul**:

$$w(a) = L - (l_1a_1 + l_2a_2 + \ldots + l_ma_m)$$

este **mai mic** decât lungimea celui mai mic reper - problema de croire se formalizează astfel:

$$(P')\begin{cases} \text{Sa se determine } y_1, y_2, \ldots, y_N \text{ care minimizeaza functia:} \\ \qquad Z = y_1 + y_2 + \ldots + y_N \\ \text{cu restrictiile :} \\ \qquad A^1 y_1 + A^2 y_2 + \ldots + A^N y_N \geq b \\ \text{si conditiile explicite impuse variabilelor:} \\ \qquad y_1, y_2, \ldots, y_N \geq 0 \text{ , intregi} \end{cases}$$

unde A^1 , A^2 , ..., A^N este (sub)lista rețetelor maximale, iar $y_1, y_2, \ldots, y_N$ sunt multiplicitățile acestora.

Observație: Dacă în modelul (P) toate restricțiile erau egalități (aceasta însemnând croirea reperelor "exact" în cantitățile cerute) în noul model (P') nu mai putem impune aceeaşi condiție deoarece,prin restrângerea modalităților de croire a unui suport, este posibil ca sistemul $\sum_{j=1}^{N} A^j y_j = b$ să nu aibe soluții întregi nenegative! Iată de ce, pentru a asigura compatibilitatea noului model suntem nevoiți să admitem că anumite repere pot fi croite "în exces".

Programele întregi (P) şi (P') sunt în esență echivalente în sensul că au acelaşi optim întreg iar soluția optimă a programului (P) utilizează cu prioritate rețete maximale; în consecință, în cele ce urmează vom studia programul "mai general" (P).

Exemplul 6.1 Considerăm cazul croirii a trei repere cu lungimile l_1 = 11, l_2 =7 , l_3 =4 din suporți cu lungimea L = 19. În următorul tabel sunt indicate toate rețetele de croire şi sunt puse în evidență rețetele maximale.

Rețeta	$a^1 \equiv A^1$	$a^2 \equiv A^2$	a^3	a^4	$a^5 \equiv A^3$	a^6	$a^7 \equiv A^4$	a^8	a^9	a^{10}	$a^{11} \equiv A^5$	a^{12}	a^{13}	a^{14}
l_1 = 11	1	1	1	1	0	0	0	0	0	0	0	0	0	0
l_2 = 7	1	0	0	0	2	2	1	1	1	1	0	0	0	0
l_3 = 4	0	2	1	0	1	0	3	2	1	0	4	3	2	1
Rest	1	0	4	8	1	5	0	4	8	12	3	7	11	15

Tabelul 6.1

Pentru cererile b_1 =12 , b_2 =18 , b_3 =30 :

- luarea în considerare a tuturor rețetelor de croire - maximale şi nemaximale - conduce la modelul:

$$(P)\begin{cases} x_1 + x_2 + x_3 + x_4 = 12 \\ x_1 + 2x_5 + 2x_6 + x_7 + x_8 + x_9 + x_{10} = 18 \\ 2x_2 + x_3 + x_5 + 3x_7 + 2x_8 + x_9 + 4x_{11} + 3x_{12} + 2x_{13} + x_{14} = 30 \\ x_j \geq 0, \text{intregi} \\ (\min) z = x_1 + x_2 + x_3 + \ldots + x_{14} \end{cases}$$

- având în vedere numai rețetele maximale obținem modelul:

$$(P')\begin{cases} y_1 + y_2 \geq 12 \\ y_1 + 2y_3 + y_4 \geq 18 \\ 2y_2 + y_3 + 3y_4 + 4y_5 \geq 30 \\ y_k \geq 0\text{, intregi} \\ (min)Z = y_1 + y_2 + y_3 + y_4 + y_5 \end{cases}$$

Ca orice problemă de programare în numere întregi ,(P) este **foarte greu de rezolvat.** În marea majoritate a aplicațiilor practice vom fi fericiți să obținem - în timp util şi cu un efort computațional rezonabil - o soluție "bună" nu neapărat optimală. Aşa cum s-a indicat în capitolul I,§1,o asemenea soluție s-ar putea obține rotunjind convenabil soluția optimă a problemei **relaxate** (PL) dedusă din (P) prin **eliminarea cerinței ca variabilele să ia numai valori întregi**. Această tactică conduce la rezultate foarte bune în special în cazurile în care cererile $b_1,b_2,...,b_m$ sunt mari; într-adevăr în aceste cazuri, componentele soluției optime fracționare vor fi suficient de mari astfel că pierderile datorate rotunjirii vor fi mici şi nesemnificative.

În continuare ne vom ocupa de rezolvarea relaxatei (PL) a problemei de croire (P):

$$(PL)\begin{cases} (min)z = \sum_{j=1}^{n} x_j \\ \sum_{j=1}^{n} a^j x_j = b \\ x_j \geq 0 \end{cases}$$

Dificultatea rezolvării acestei probleme rezidă în numărul foarte mare de coloane (rețete) pe care le poate avea (mai cu seamă în situațile reale) şi care - în cazul rezolvării "obişnuite" - ar trebui mai întâi generate. Vom vedea în continuare cum se poate evita acest impediment.

6.2 Teoria metodei generării de coloane

Vom aplica problemei (PL) versiunea revizuită a algoritmului simplex.. La start, se poate pleca cu baza formată din cele m rețete unitare:
$e^1 = (1,0,...,0)^T$, $e^2 = (0,1,...,0)^T$, ... , $e^m = (0,0,...,1)^T$ cu tabelul redus:

e^1	b_1	1			
e^2	b_2		1		
⋮	⋮			⋱	
e^m	b_m				1
f	Σb_i	1	1	…	1

Tabelul 6.2

Cel mai bine este să se plece cu baza formată din cele m rețete unicat:

$$K^1 = (r_1,0,...,0)^T \text{ , } K^2 = (0,r_2,...,0)^T \text{ , ... , } K^m = (0,0,...,r_m)^T \text{ în care:}$$

$$r_1 = \left\lfloor L/l_1 \right\rfloor \text{ , } r_2 = \left\lfloor L/l_2 \right\rfloor \text{ , ... , } r_m = \left\lfloor L/l_m \right\rfloor$$

şi cu tabelul simplex redus:

K^1	b_1/r_1	$1/r_1$			
K^2	b_2/r_2		$1/r_2$		
⋮	⋮			⋱	
K^m	b_m/r_m				$1/r_m$
f	$\Sigma b_i/r_i$	$1/r_1$	$1/r_2$		$1/r_m$

Tabelul 6.3

Fie B baza admisibilă **curentă,** $\bar{x} = \begin{bmatrix} \bar{b} \\ 0 \end{bmatrix} \begin{matrix} \leftarrow x^B \\ \leftarrow x^S \end{matrix}$ soluția asociată bazei B. Presupunem disponibil tabelul simplex corespunzător; vezi tabelul 6.4

Reamintim că:

$$\bar{b} = B^{-1} \cdot b$$
$$\pi = c^B \cdot B^{-1} = [1,1,...,1] \cdot B^{-1}$$
$$\bar{f} = c^B \cdot B^{-1} \cdot b = \pi \cdot b$$

După cum se ştie, soluția $\bar{x}$ va fi optimă dacă, pentru toate coloanele $a^1, a^2, ..., a^n$ avem:

$$\bar{c}_j = c^B \cdot B^{-1} \cdot a^j - c_j = \pi \cdot a^j - 1 \le 0 \quad j = 1,..., n$$

Pentru a testa îndeplinirea condiției de mai sus este suficient să calculăm:

$$\max_{j=1,..,n} [\pi \cdot a^j = \pi_1 a_1^j + \pi_2 a_2^j + ... + \pi_m a_m^j]$$

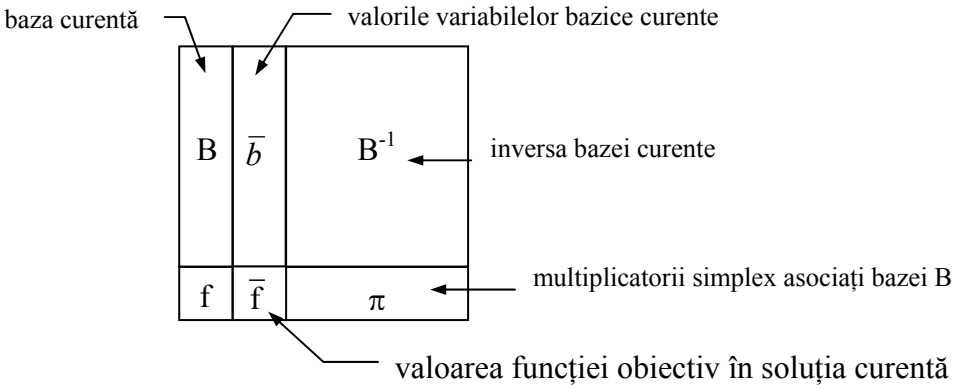
şi cum fiecare a^j este o soluție cu componente întregi nenegative a inecuației $l_1 a_1 + l_2 a_2 + ... + l_m a_m \le L$ va fi suficient să rezolvăm programul auxiliar:

$$R(\pi) \begin{cases} \max \quad \pi_1 a_1 + \pi_2 a_2 + ... + \pi_m a_m \\ l_1 a_1 + l_2 a_2 + ... + l_m a_m \le L \\ a_1, a_2, ..., a_m \ge 0 \text{ intregi} \end{cases}$$

Dacă maximul funcției obiectiv din $R(\pi)$ este ≤ 1 este clar că $\bar{c}_j = \pi \cdot a^j - 1 \le 0$ pentru toți j = *1,...,n* şi soluția $\bar{x}$ asociată bazei B este optimă.
Dacă maximul din $R(\pi)$ este > 1 atunci soluția optimă a^* a programului $R(\pi)$ este o rețetă, fie ea a^k, din lista $a^1, a^2, ..., a^n$ a tuturor rețetelor, cu proprietatea:

$$\bar{c}_k = \pi \cdot a^k - 1 > 0 .$$

Introducem în baza curentă coloana a^k urmând instrucțiunile algoritmului simplex revzuit. Obținem o nouă bază admisibilă B', o nouă soluție $\bar{x}'$ a problemei (PL) în general mai bună decât soluția veche $\bar{x}$ şi un nou tabel simplex redus în care aopare un nou vector π' de multiplicatori simplex.Pentru a testa optimalitatea noii soliuții rezolvăm programul $R(\pi')$ etc.
Procesul iterativ se încheie într-un număr **finit** de paşi cu găsirea soluției optime a problemei (PL).



Tabelul 6.4

6.3 Rezumatul procedurii Generare de Coloane pentru rezolvarea relaxatei problemei de croire

Start Se pleacă cu baza formată din rețetele unicat(6.1) şi cu tabelul simplex redus 6.3. Fie B baza curentă şi $\pi = c^B \cdot B^{-1}$ vectorul multiplicatorilor simplex corespunzători.
Conținutul unei iterații:

Pasul 1 Se rezolvă problema auxiliară:

$$R(\pi)\begin{cases} \max \sum_{i=1}^{m} \pi_i a_i \\ \sum_{i=1}^{m} l_i a_i \le L \\ a_i \ge 0 \text{ intregi} \quad i = 1,\ldots,m \end{cases}$$

(vezi secțiunea următoare în ceeace privește modul algoritmic de rezolvare al problemei $R(\pi)$.

Pasul 2 Dacă maximul funcției obiectiv din $R(\pi)$ este ≤ 1 **stop:** soluția curentă a problemei (PL) este optimă. Altminteri:

Pasul 3 Fie a^* soluția optimă a problemei $R(\pi)$. Se introduce în baza curentă B coloana a^* (reindexată eventual cu numărul de ordine al iterației) urmând instrucțiunile algoritmului simplex revizuit. Se revine la pasul 1 în cadrul unei noi iterații.

Exemplul 6.2 Vom rezolva relaxata (PL) a problemei de croire (P) din exemplul 6.1 (sfătuim cititorul să ignore faptul că am generat deja toate rețetele de croire ale problemei...De altfel, diferitele rețete folosite de algoritm vor avea o notare diferită de cea din tabelul 6.1)

Start. Plecăm cu baza formată din rețetele unicat:

$$K^1 = (1,0,0)^T \, , K^2 = (0,2,0)^T \, , K^3 = (0,0,4)^T$$

$B = [K^1, K^2, K^3] = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 4 \end{bmatrix}$ este o matrice diagonală a cărei inversă este : $B^{-1} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1/2 & 0 \\ 0 & 0 & 1/4 \end{bmatrix}$.

Soluția asociată bazei B = [K^1,K^2,K^3]:

$$B^{-1} \cdot b = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1/2 & 0 \\ 0 & 0 & 1/4 \end{bmatrix} \begin{bmatrix} 12 \\ 18 \\ 30 \end{bmatrix} = \begin{bmatrix} 12 \\ 9 \\ 15/2 \end{bmatrix} \begin{matrix} \leftarrow \text{ multiplicitatea retetei } K^1 \\ \leftarrow \text{ multiplcitatea retetei } K^2 \\ \leftarrow \text{ multiplicitatea retetei } K^3 \end{matrix}$$

(celelalte rețete - pe cqre de fapt nu le ştim - nu se folosesc).
Multiplicatorii simplex asociați bazei [K^1,K^2,K^3]sunt:

$$\pi = c^B \cdot B^{-1} = [1,1,1] \cdot \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1/2 & 0 \\ 0 & 0 & 1/4 \end{bmatrix} = [1, 1/2, 1/4]$$

Valoarea funcției obiectiv în soluția construită este: $\bar{f} = \pi \cdot b = 1 \cdot 12 + 1/2 \cdot 18 + 1/4 \cdot 30 = 28\,1/2 \cdot$
Tabelul simplex redus de start:

K^1	12	1	0	0
K^2	9	0	1/2	0
K^3	15/2	0	0	1/4
f	57/2	1	1/2	1/4

Tabelul 6.5

Iterația 1 Se rezolvă problema :

$$R(\pi) \begin{cases} (\max)\, \rho = 1 \cdot a_1 + 1/2 \cdot a_2 + 1/4 \cdot a_3 = 1/4\,(4a_1 + 2a_2 + a_3) \\ 11a_1 + 7a_2 + 4a_3 \le 19 \\ a_1, a_2, a_3 \ge 0 \text{ intregi} \end{cases}$$

Prin simplă inspecție (în cazul de față) sau aplicând un algoritm adecvat dacă numărul reperelor este mare (vezi secțiunea următoare) se găseşte (max)ρ = 3/2 > 1 şi soluția optimă a^* = (1,1,0) care este o rețetă maximală. Renotăm a^* : $A^1 = (1,1,0)^T$ şi introducem A^1 în baza curentă:

coloana $B^{-1} \cdot A^1$

A^1	→	1	1	0		
K^1	12	1	0	0	(1)	≡ pivot
K^2	9	0	1/2	0	1/2	
K^3	15/2	0	0	1/4	0	
f	57/2	1	1/2	1/4	1/2	=(max)ρ-1

Tabelul6.6a

⇒

A^1	12	1	0	0
K^2	3	-1/2	1/2	0
K^3	15/2	0	0	1/4
f	45/2	1/2	1/2	1/4

Tabelul 6.6b

Iterația 2 Rezolvăm problema:

$$R(\pi)\begin{cases}(\max)\ \rho = \tfrac{1}{2}\cdot a_1 + \tfrac{1}{2}\cdot a_2 + \tfrac{1}{4}\cdot a_3 = \tfrac{1}{4}(2a_1 + 2a_2 + a_3) \\ 11a_1 + 7a_2 + 4a_3 \le 19 \\ a_1, a_2, a_3 \ge 0 \text{ intregi}\end{cases}$$

al cărei optim , (max)ρ = 5/4 > 1 , se atinge pe rețeta maximală $a^* = (0,2,1)^T$,renotată A^2. Introducem A^2 în baza curentă:

coloana $B^{-1}\cdot A^2$

A^2	→	0	2	1	
A^1	12	1	0	0	0
K^2	3	-1/2	1/2	0	(1) ≡ pivot
K^3	15/2	0	0	1/4	1/4
f	45/2	1/2	1/2	1/4	1/4 =(max)ρ-1

⇒

A^1	12	1	0	0
A^2	3	-1/2	1/2	0
K^3	27/4	1/8	-1/8	1/4
f	87/4	5/8	3/8	1/4

Tabelul6.7a Tabelul 6.7b

Iterația 3 Acum se rezolvă problema:

$$R(\pi)\begin{cases}(\max)\ \rho = \tfrac{5}{8}\cdot a_1 + \tfrac{3}{8}\cdot a_2 + \tfrac{1}{4}\cdot a_3 = \tfrac{1}{8}(5a_1 + 3a_2 + 2a_3) \\ 11a_1 + 7a_2 + 4a_3 \le 19 \\ a_1, a_2, a_3 \ge 0 \text{ intregi}\end{cases}$$

Se găseşte (max)ρ = 9/8 > 1 şi soluția optimă $a^* = (1,0,2)^T$,renotată A^3. Introducem A^3 în baza curentă:

coloana $B^{-1}\cdot A^3$

A^3	→	1	0	2	
A^1	12	1	0	0	1
A^2	3	-1/2	1/2	0	-1/2
K^3	27/4	1/8	-1/8	1/4	(5/8) ≡ pivot
f	87/8	5/8	3/8	1/4	1/8 =(max)ρ-1

⇒

A^1	6/5	4/5	1/5	-2/5
A^2	42/5	-2/5	2/5	1/5
A^3	54/5	1/5	-1/5	2/5
f	102/5	3/5	2/5	1/5

Tabelul6.8a Tabelul 6.8b

Iterația 4 Rezolvăm problema:

$$R(\pi)\begin{cases}(\max)\ \rho = \tfrac{3}{5}\cdot a_1 + \tfrac{2}{5}\cdot a_2 + \tfrac{1}{5}\cdot a_3 = \tfrac{1}{5}(3a_1 + 2a_2 + a_3) \\ 11a_1 + 7a_2 + 4a_3 \le 19 \\ a_1, a_2, a_3 \ge 0 \text{ intregi}\end{cases}$$

De această dată (max)ρ = 1 astfel că soluția curentă,înscrisă în tabelul 6. , este optimă.

În concluzie, soluția optimă **fracționară** a problemei de croire date utilizează:

- rețeta maximală $A^1 = (1,1,0)^T$ cu multiplicitatea $\frac{6}{5} = 1\frac{1}{5}$;
- rețeta maximală $A^2 = (0,2,1)^T$ cu multiplicitatea $\frac{42}{5} = 8\frac{2}{5}$;
- rețeta maximală $A^3 = (1,0,2)^T$ cu multiplicitatea $\frac{54}{5} = 10\frac{4}{5}$.

Numărul suporților "consumați" este : $\frac{102}{5} = 20\frac{2}{5}$.

Observație: Reîntorcându-ne la problema de croire generală (P) şi la relaxata acesteia se constată imediat că **optimul întreg este cel puțin egal cu rotunjirea superioară a optimului fracționar!**

În cazul de față rezultă că soluția optimă întreagă va utiliza cel puțin $\left\lceil 20\frac{2}{5} \right\rceil = 21$ suporți.

Să vedem acum cum se determină o soluție "bună" pentru problema de croire studiată.

Etapa 1 Se rotunjesc **inferior** multiplicitățile rețetelor din soluția optimă a problemei relaxate (PL):

$$\left\lfloor \frac{6}{5} \right\rfloor = 1 \quad ; \quad \left\lfloor \frac{42}{5} \right\rfloor = 8 \quad ; \quad \left\lfloor \frac{54}{5} \right\rfloor = 10$$

Etapa 2 Se determină cantitățile de repere ce pot fi croite cu rețetele din soluția optimă fracționară dar cu multiplicitățile rotunjite inferior:

$$\bar{b} = 1 \cdot A^1 + 8 \cdot A^2 + 10 \cdot A^3 = \begin{bmatrix} 1 \\ 1 \\ 0 \end{bmatrix} + 8 \cdot \begin{bmatrix} 0 \\ 2 \\ 1 \end{bmatrix} + 10 \cdot \begin{bmatrix} 1 \\ 0 \\ 2 \end{bmatrix} = \begin{bmatrix} 11 \\ 17 \\ 28 \end{bmatrix}$$

Etapa 3 Se determină cantitățile de repere care mai sunt de croit:

$$b' = b - \bar{b} = \begin{bmatrix} 12 \\ 18 \\ 30 \end{bmatrix} - \begin{bmatrix} 11 \\ 17 \\ 28 \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \\ 2 \end{bmatrix}$$

Etapa 4 Pentru "**cererea reziduală**" b' se aplică următoarea euristică, numită FFD (**F**irst **F**it **D**ecreasing):

- se determină prima rețetă în sens lexicografic care "încape" în b';
- se actualizează b' prin extragerea rețetei găsite şi se reia pasul precedent.

În cazul nostru prima rețetă cuprinsă în b' este $(1,1,0)^T = A^1$.Actualizăm cererea reziduală:

$$b' \leftarrow \begin{bmatrix} 1 \\ 1 \\ 2 \end{bmatrix} - \begin{bmatrix} 1 \\ 1 \\ 0 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 2 \end{bmatrix}$$

Au mai rămas două repere cu lungimea $l_3 = 4$ a căror croire necesită consumarea unui suport .

Recapitulând, o soluție "bună" pentru croirea cantităților de repere cerute ar fi următoarea:

- se foloseşte rețeta $A^1 = (1,1,0)$ de 1 + 1 = 2 ori;
- se foloseşte rețeta $A^2 = (0,2,1)$ de 8 ori;
- se foloseşte rețeta $A^3 = (1,0,2)$ de 10 ori;
- se mai taie dintr-un suport două repere cu lungimea $l_3 = 4$ adică se foloseşte rețta nemaximală (0,0,2) .

În total se consumă 2 + 8 + 10 + 1 =21 suporți şi în baza unei observații anterioare soluția construită este chiar optimă!

Concluzii finale

1. În soluția optimă a problemei relaxate se utilizează numai rețete maximale;
2. După aplicarea euristicii FFD asupra cererii reziduale, pot apare şi câteva rețete nemaximale - de regulă una singură;
3. Numeroasele experimente numerice au arătat că optimul întreg al problemei de croire unidimensionale este de regulă egal cu rotunjirea întreagă superioară a optimului fracționar şi numai în rare cazuri este mai mare decât aceasta cu exact o unitate!

§7 Programare dinamică

În această secțiune ne vom opri asupra problemei :

$$(R)\begin{cases}\max \quad \rho = \pi_1 a_1 + \pi_2 a_2 + ... + \pi_m a_m \\ l_1 a_1 + l_2 a_2 + ... + l_m a_m \le L \\ a_1, a_2, ..., a_m \ge 0 \text{ , intregi}\end{cases}$$

în care $L > l_1 > l_2 > ... > l_m$ sunt întregi pozitivi.

În secțiunea precedentă, (R) a apărut ca **subproblemă** în rezolvarea relaxatei problemei de croire unidimensionale prin **metoda generaării de coloane.** Este clar că eficacitatea metodei amintite depinde de performanțele algoritmilor utilizați pentru rezolvarea problemei (R).

(R) este un **progam liniar în numere întregi** foarte simplu, având **o singură restricție**. În literatura de specialitate este cunoscută sub numele de **problema rucsacului** datorită următoarei interpretări:

a_i este numărul pieselor de echipament de greutate l_i şi utilitate π_i care trebuie luate într-o excursie într- un rucsac ce suportă o greutate maximă L. Întrebare: ce piese de echipament trebuie alese şi în câte exemplare vor fi acestea introduse în rucsac astfel încât utilitatea încărcăturii să fie maximă?

Firește, (R) poate fi rezolvată prin metodele specifice programării în numere întregi (plane de secțiune, Branch & Bound etc). Faptul că (R) are o singură restricție permite abordarea ei prin **programare dinamică.** Mai precis, pentru fiecare k = 1,…,m şi fiecare întreg λ =0,1,…,L considerăm problema:

$$R_k(\lambda)\begin{cases}\max \quad \pi_1 a_1 + \pi_2 a_2 + ... + \pi_k a_k \\ l_1 a_1 + l_2 a_2 + ... + l_k a_k \le \lambda \\ a_1, a_2, ..., a_k \ge 0 \text{ , intregi}\end{cases}$$

al cărei optim îl notăm cu $\rho_k(\lambda)$. Este clar că $R = R_m(\lambda)$ şi căaximul funcției obiectiv din R este $\rho_m(L)$.

Observăm că, pentru k fixat ρ_k este o funcție de o singură variabilă ale cărei valori admisibile sunt 0,1,…,L.

Funcțiile $\rho_1, \rho_2, \ldots, \rho_{m-1}, \rho_m$ pot fi determinate astfel:

- $$\rho_1(\lambda) = \max\{\pi_1 a_1 \,|\, l_1 a_1 \le \lambda\} = \pi_1 \left\lfloor \frac{\lambda}{l_1} \right\rfloor \qquad \lambda = 0,1,\ldots,L \tag{7.1}$$

- pentru k > 1 avem formula de recurență:

$$\rho_k(\lambda) = \max\{\rho_{k-1}(\lambda - l_k a_k) + \pi_k a_k \mid a_k = 0,1,\ldots,\left\lfloor \frac{\lambda}{l_k} \right\rfloor\} \tag{7.2}$$

- pentru k = m este suficient să găsim numai valoarea funcției ρ_m în L:

$$\rho_m(L) = \max\{\rho_{m-1}(L - l_m a_m) + \pi_m a_m \mid a_m = 0,1,\ldots,\left\lfloor \frac{\lambda}{l_m} \right\rfloor\}$$

Relația (7.2) arată că funcțiile $\rho_2,\ldots,\rho_{m-1},\rho_m$ rezultă din nişte procese de optimizare **unidimensionale.**

Să presupunem cunoscute funcțiile $\rho_1, \rho_2, \ldots, \rho_{m-1}$ şi valoarea $\rho_m(L)$ şi să notăm cu $a_k^\bullet(\lambda)$ valoarea variabilei a_k care – pentru λ dat – realizează maximul din formula de recurență (7.2). Pentru k = 1 avem $a_1^\bullet(\lambda) = \left\lfloor \frac{\lambda}{l_1} \right\rfloor$. Atunci, o soluție optimă a problemei (R) se găseşte astfel:

$$a_m^\bullet = a_m^\bullet(L)$$

Pentru $k = m-1,\ldots,2,1$:

$$a_k^\bullet = a_k^\bullet(L - S_k)$$

unde :

$$S_k = l_{k+1} a_{k+1}^\bullet + \ldots + l_m a_m^\bullet$$

Observații: 1) În termenii problemei rucsacului $\rho_k(\lambda)$ este valoarea maximă a unei încărcări a rucsacului ce nu depăşeşte în greutate plafonul λ şi care este formată numai din primele k tipuri de echipament.

2) Prin programarea dinamică rezolvarea problemei de **optimizare multidimensională** (R) este înlocuită cu o **secvență de optimizări unidimensionale** bazate pe formula de recurență (7.2).

3) Ecuația funcțională (7.2), prin care funcțiile $\rho_1, \rho_2, \ldots, \rho_{m-1}, \rho_m$ se deduc **una din alta** formalizează – în cazul problemei (R) – **principiul central al programării dinamice** datorat lui R. BELLMAN : **O strategie (secvențială) optimă are proprietatea că oricare ar fi starea inițială şi decizia inițială, deciziile rămase constituie o strategie optimă în raport cu starea care rezultă din prima decizie.**

Demonstrația formulei (7.2)

Fie $\overline{a}_k$ o valoare întreagă , $0 \le \overline{a}_k \le \left\lfloor \frac{\lambda}{l_k} \right\rfloor$, dată variabilei a_k . Fie $(\overline{a}_1, \overline{a}_2, ..., \overline{a}_{k-1})$ o soluție optimă a problemei $R_{k-1}(\lambda - l_k \overline{a}_k)$. Prin urmare:

$$\rho_{k-1}(\lambda - l_k \overline{a}_k) = \pi_1 \overline{a}_1 + ... + \pi_{k-1} \overline{a}_{k-1}$$
$$l_1 \overline{a}_1 + ... + l_{k-1} \overline{a}_{k-1} \le \lambda - l_k \overline{a}_k$$

Deoarece $l_1 \overline{a}_1 + ... + l_{k-1} \overline{a}_{k-1} + l_k \overline{a}_k \le \lambda$ rezultă că $(\overline{a}_1, ..., \overline{a}_{k-1}, \overline{a}_k)$ este o soluție admisibilă a problemei $R_k(\lambda)$ care dă funcției obiectiv valoarea:

$$\pi_1 \overline{a}_1 + ... + \pi_{k-1} \overline{a}_{k-1} + \pi_k \overline{a}_k = \rho_{k-1}(\lambda - l_k \overline{a}_k) + \pi_k \overline{a}_k$$

În consecință:

$$\rho_k(\lambda) \ge \rho_{k-1}(\lambda - l_k \overline{a}_k) + \pi_k \overline{a}_k$$

şi cum $\overline{a}_k$ a fost arbitrar aleasă (între 0 şi $\left\lfloor \frac{L}{l_k} \right\rfloor$) urmează că:

$$\rho_k(\lambda) \ge \max\{\rho_{k-1}(\lambda - l_k a_k) + \pi_k a_k \mid a_k = 0,1,..., \left\lfloor \frac{\lambda}{l_k} \right\rfloor\}$$

Pe de altă parte fie $(a_1^\bullet, ..., a_{k-1}^\bullet, a_k^\bullet)$ o soluție optimă a problemei $R_k(\lambda)$.Prin urmare:

$$\rho_k(\lambda) = \pi_1 a_1^\bullet + ... + \pi_{k-1} a_{k-1}^\bullet + \pi_k a_k^\bullet$$
$$l_1 a_1^\bullet + ... + l_{k-1} a_{k-1}^\bullet + l_k a_k^\bullet \le \lambda$$

Din $l_1 a_1^\bullet + ... + l_{k-1} a_{k-1}^\bullet \le \lambda - l_k a_k^\bullet$ rezultă că $(a_1^\bullet, ..., a_{k-1}^\bullet)$ este o soluție admisibilă a problemei $R_{k-1}(\lambda - l_k a_k^\bullet)$ şi deci:

$$\pi_1 a_1^\bullet + ... + \pi_{k-1} a_{k-1}^\bullet \le \rho_{k-1}(\lambda - l_k a_k^\bullet)$$

Să arătăm că în ultima relație avem egalitate. Presupunând prin absurd contrariul fie $(a_1^{\bullet\bullet}, ..., a_{k-1}^{\bullet\bullet})$ o soluție optimă a problemei $R_{k-1}(\lambda - l_k a_k^\bullet)$. În consecință vom avea:

$$\pi_1 a_1^{\bullet\bullet} + ... + \pi_{k-1} a_{k-1}^{\bullet\bullet} = \rho_{k-1}(\lambda - l_k a_k^\bullet) > \pi_1 a_1^\bullet + ... + \pi_{k-1} a_{k-1}^\bullet$$

$$l_1 a_1^{\bullet\bullet} + ... + l_{k-1} a_{k-1}^{\bullet\bullet} \le \lambda - l_k a_k^\bullet \quad \rightarrow \quad l_1 a_1^{\bullet\bullet} + ... + l_{k-1} a_{k-1}^{\bullet\bullet} + l_k a_k^\bullet \le \lambda$$

Prin urmare $(a_1^{\bullet\bullet}, ..., a_{k-1}^{\bullet\bullet}, a_k^\bullet)$ este o soluție admisibilă a problemei $R_k(\lambda)$ şi deoarece

$$\pi_1 a_1^{\bullet\bullet} + ... + \pi_{k-1} a_{k-1}^{\bullet\bullet} + \pi_k a_k^\bullet > \pi_1 a_1^\bullet + ... + \pi_{k-1} a_{k-1}^\bullet + \pi_k a_k^\bullet = \rho_k(\lambda)$$

tragem concluzia că $(a_1^\bullet, ..., a_{k-1}^\bullet, a_k^\bullet)$ nu este soluție optimă a problemei $R_k(\lambda)$ contrar ipotezei. În definitiv:

$$\pi_1 a_1^{\bullet}+\ldots+\pi_{k-1}a_{k-1}^{\bullet} = \rho_{k-1}(\lambda - l_k a_k^{\bullet})$$

astfel că:

$$\rho_k(\lambda) = \pi_1 a_1^{\bullet}+\ldots+\pi_{k-1}a_{k-1}^{\bullet} + \pi_k a_k^{\bullet} = \rho_{k-1}(\lambda - l_k a_k^{\bullet}) + \pi_k a_k^{\bullet} \leq$$

$$\leq max\{\rho_{k-1}(\lambda - l_k a_k) + \pi_k a_k \mid a_k = 0,1,\ldots,\left\lfloor \frac{L}{l_k} \right\rfloor\}$$

Egalitatea (7.2) este demonstrată.

Exemplul 7.1 Vom aplica procedura descrisă problemei:

$$(R)\begin{cases} max\ \rho = 16a_1 + 12a_2 + 5a_3 \\ 5a_1 + 4a_2 + 2a_3 \leq 11 \\ a_1, a_2, a_3 \geq 0 \text{ intregi} \end{cases}$$

Iterația 1 Determinăm valorile funcției $\rho_1(\lambda) = \pi_1\left\lfloor \frac{\lambda}{l_1} \right\rfloor = 16\left\lfloor \frac{\lambda}{5} \right\rfloor$ pentru λ = 0,1,...,11

Ele sunt înscrise în tabelul 7.1

λ	0	1	2	3	4	5	6	7	8	9	10	11
$\rho_1(\lambda)$	0	0	0	0	0	16	16	16	16	16	32	32
$a_1^{\bullet}(\lambda)$	0	0	0	0	0	1	1	1	1	1	2	2

Tabelul 7.1

Iterația 2 În continuare calculăm valorile funcției

$$\rho_2(\lambda) = max\{\rho_1(\lambda - l_2 a_2) + \pi_2 a_2 \mid a_2 = 0,1,\ldots,\left\lfloor \frac{\lambda}{l_2} \right\rfloor\} =$$

$$= max\{\rho_1(\lambda - 4a_2) + 12a_2 \mid a_2 = 0,1,\ldots,\left\lfloor \frac{\lambda}{4} \right\rfloor\} \text{ unde } \lambda = 0,1,\ldots,11$$

Astfel, pentru λ = 0,1,2,3 $\rho_2(\lambda) = \rho_1(\lambda)$; pentru λ = 4,5,6,7 vom avea:

$$\rho_2(\lambda) = max\{\rho_1(\lambda - 4a_2) + 12a_2 \mid a_2 = 0,1\} = max\{\rho_1(\lambda),\ \rho_1(\lambda - 4) + 12\}$$

iar pentru λ =8,9,10,11

$$\rho_2(\lambda) = max\{\rho_1(\lambda - 4a_2) + 12a_2 \mid a_2 = 0,1,2\} = max\{\rho_1(\lambda), \rho_1(\lambda - 4) + 12, \rho_1(\lambda - 8) + 24\}$$

Rezultatele sunt afişate în tabelul 7.2

λ	0	1	2	3	4	5	6	7	8	9	10	11
$\rho_2(\lambda)$	0	0	0	0	12	16	16	16	24	28	32	32
$a_2^{\bullet}(\lambda)$	0	0	0	0	1	0	0	0	2	1	0	0

Tabelul 7.2

Iterația 3 În final vom evalua numai:

$$\rho_3(L) = \rho_3(11) = max\{\rho_2(11-2a_3)+5a_3 \mid a_3 = 0,1,\ldots,5 = \left\lfloor \frac{11}{2} \right\rfloor\} =$$

$$max\{\rho_2(11),\rho_2(9)+5,\rho_2(7)+10,\rho_2(5)+15,\rho_2(3)+20,\rho_2(1)+25\} =$$
$$max\{32,28+5,16+10,0+20,0+25\} = 33 \quad \rightarrow \quad a_3^{\bullet} = 1$$

Determinarea soluției optime ("de la sfârşit către început")

Pasul 1 $a_3^{\bullet} = a_3^{\bullet}(11) = 1$;

Pasul 2 $S_2 = l_3 a_3^{\bullet} = 2 \cdot 1 = 2 \quad \rightarrow \quad L - S_2 = 11 - 2 = 9 \quad \rightarrow \quad a_2^{\bullet} = a_2^{\bullet}(9) = 1$;

Pasul 3 $S_1 = l_2 a_2^{\bullet} + l_3 a_3^{\bullet} = l_2 a_2^{\bullet} + S_2 \; 4 \cdot 1 + 2 = 6 \rightarrow L - S_1 = 11 - 6 = 5 \rightarrow$
$\rightarrow a_1^{\bullet} = a_1^{\bullet}(5) = 1$.

Soluția optimă a problemei date este: $a_1^{\bullet} = 1$, $a_2^{\bullet} = 1$, $a_3^{\bullet} = 1 \quad (max)\rho = 33$

§ 8. Întrebări şi probleme

1.Este cunoscut faptul că problemele practice de optimizare de dimensiuni "mari" au o structură "specială". Ce înseamnă această structură specială în programarea liniară?

2.Ce caracteristici are programul principal (P^m) rezultat din aplicarea metodei de descompunere Dantzig - Wolfe? Ce metodă se utilizează pentru rezolvarea lui?

3.Se consideră un program liniar în formă canonică de maximizare a cărui mulțime de restricții a fost partiționată în două blocuri:

$$(P)\begin{cases} Ax \le b \\ Mx \le d \\ x \ge 0 \\ (max) f = cx \end{cases}$$ în notațiile matriciale ale secțiunii 3.

Să presupunem că $\bar{x}$ şi $\bar{u}$ sunt doi vectori nenegativi de dimensiuni convenabile astfel încât:

- $\bar{x}$ este soluția optimă a programului
$$P(\bar{u})\begin{cases} Ax \le b \\ x \ge 0 \\ (max) f' = (c - \bar{u}M)x \end{cases}$$
- $M\bar{x} \le d$
- $\bar{u}(d - M\bar{x}) = 0$

Să se arate că $\bar{x}$ este soluția optimă a programului (P).

.4. [3] Utilizați algoritmul de descompunere Dantzig - Wolfe la rezolvarea următoarelor programe liniare custructură bloc - diagonală şi restricții de cuplare:

$$a)\begin{cases} 2x_1+3x_2 \le 6 \\ 5x_1+x_2 \le 5 \\ 3x_3+4x_4 \ge 12 \\ x_3 \le 4 \\ x_4 \le 3 \\ x_1+4x_2+5x_3+2x_4 \le 7 \\ x_j \ge 0 \quad j=1,\ldots,4 \\ (max) f = x_1+8x_2+5x_3+6x_4 \end{cases} \qquad b)\begin{cases} x_1+3x_2 \le 30 \\ 2x_1+x_2 \le 20 \\ x_3+x_4 \le 15 \\ x_3 \le 10 \\ x_4 \le 10 \\ x_1+2x_2+2x_3+x_4 \le 40 \\ x_j \ge 0 \quad j=1,\ldots,4 \\ (max) f = x_1+x_2+2x_3+x_4 \end{cases}$$

În rezolvarea subproblemelor de la nivelul 1 se poate folosi metoda grafică.

5. Pentru instalația de apă a unui imobil în construcție sunt necesare 80 de țevi de 2m, 40 de țevi de 2,50m şi 30 de țevi cu lungimea de 3,50m. Aceste bucăți se taie din țevi cu lungimea de 9m.

a) Alcătuiți lista rețetelor maximale de croire;

b) Scrieți un program liniar în numere întregi pentru minimizarea numărului de țevi de 9m ce vor fi tăiate;

c) Rezolvați programul relaxat prin metoda generării de coloane;

d) Plecând de la soluția optimă fracționară construiți o soluție "bună" a problemei date;ar putea fi optimă soluția construită de dvs.?

6.Problema rucsacului. Formulare şi model matematic.Descrieți algoritmul de ezolvare al problemei rucsacului prin programare dinamică.

7. Rezolvați problemele de tip rucsac:

$$a)\begin{cases} (max)\rho = 5a_1+12a_2+16a_3 \\ 2a_1+4a_2+5a_3 \le 11 \end{cases} \qquad b)\begin{cases} (max)\rho = a_1+3a_2+5a_3+9a_4 \\ 2a_1+3a_2+4a_3+7a_4 \le 10 \end{cases}$$