

CAPITOLUL III

PROBLEME DE OPTIMIZARE DE DIMENSIUNI MARI

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

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

- numărul de variabile și numărul de restricții;
- complexitatea expresiilor funcției obiectiv și a restricțiilor;
- performanțele echipamentului de calcul: hardware și software.

Utilizarea modelelor matematice în studiul unor situații reale - în special din domeniul economic - a condus la programe matematice care suprasolicită și 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** așezate "**pe diagonală**";
- număr **foarte mare** de variabile și **relativ puține** restricții sau invers, multe restricții și puține 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 restricții și 10^6 variabile, matricea coeficienților ar avea 10^{10} intrări și în cazul unei densități de 100% ar necesita 10^{10} date numerice de adunat, sortat și prelucrat!!

Pentru programele neliniare, complexitatea și 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 știut 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**:

$$\left\{ \begin{array}{l} \sum_{j=1}^n a_{ij}x_j = b_i \quad i = 1, \dots, m \\ 0 \leq x_j \leq u_j \quad j = 1, \dots, n \\ (\max)f = \sum_{j=1}^n c_j x_j \end{array} \right.$$

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

$$\left\{ \begin{array}{l} \sum_{j=1}^n a_{ij}x_j = b_i \quad i = 1, \dots, m \\ x_j + x_{n+j} = u_j \quad j = 1, \dots, n \\ x_j \geq 0 \quad j = 1, \dots, 2n \\ (\max)f = \sum_{j=1}^n c_j x_j \end{array} \right.$$

Rezulta un program cu $m+n$ restricții și $2n$ variabile, ale cărei baze erau matrici de ordinul $m+n$.

Totuși 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** (și 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 și transmite nivelului inferior noi parametri.

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

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

§3. Descompunere în programarea liniară. Principiu

Considerăm un program liniar în formă standard:

$$(P) \left\{ \begin{array}{l} Tx = t \\ x \geq 0 \\ (\max)f = cx \end{array} \right. \quad \text{în care:}$$

$T \equiv$ matrice $m \times n$;

$x \equiv$ coloana celor n variabile;

$t \equiv$ coloana celor m termeni liberi;

$c \equiv$ linia celor n coeficienți din funcția obiectiv.

Pentru moment, nu vom face nici o ipoteză asupra "mărimii" programului (P) sau asupra structurii sale. Știm 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

$$\begin{array}{c} T \\ \hline \begin{array}{|c|} \hline A \\ \hline M \\ \hline \end{array} \\ \hline t \\ \begin{array}{|c|} \hline b \\ \hline d \\ \hline \end{array} \end{array}$$

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

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

Este bine cunoscut faptul că A este o mulțime **convexă** și chiar **poliedrală** (adică o intersecție finită de semispații din \mathbb{R}^n). Ea are un număr **finit** de **vârfuri** v^1, v^2, \dots, 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 și mărginite A se scrie ca o **combinație convexă** a vârfurilor ei:

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

unde:

$$\lambda_1 \geq 0, \lambda_2 \geq 0, \dots, \lambda_s \geq 0 \text{ și } \lambda_1 + \lambda_2 + \dots + \lambda_s = 1$$

Înlocuind $x = \sum_{k=1}^s \lambda_k v^k$ în blocul $Mx = d$ și 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:

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

programul (P) se scrie echivalent:

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

Dacă $(\lambda_k^*, k = 1, \dots, s)$ 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).

Figura 3.2

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

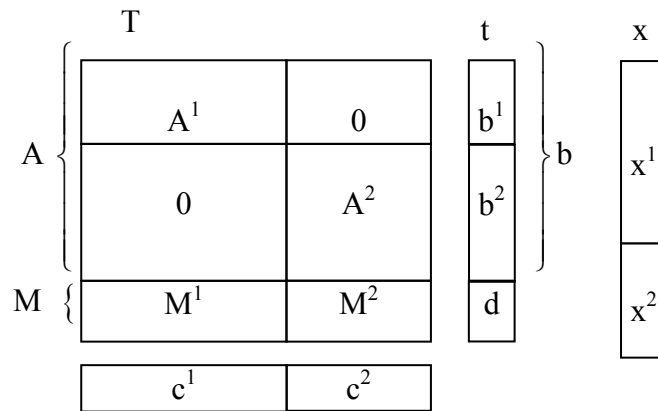


Figura 3.3

Programul (P) are deci forma:

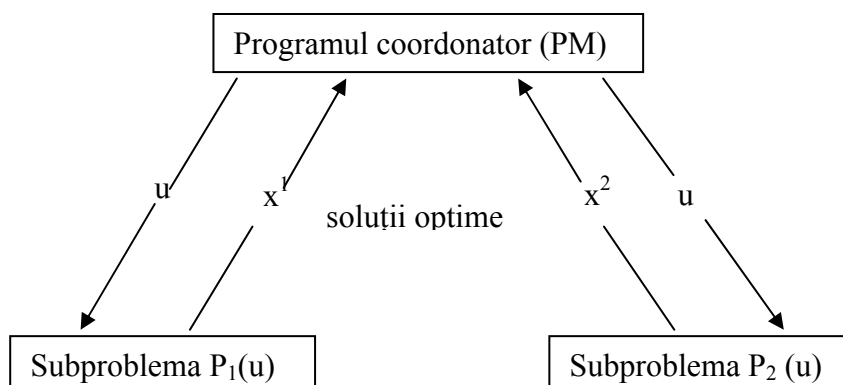
$$(P) \begin{cases} Tx = t \\ x \geq 0 \\ (\max) f = cx \end{cases} \Leftrightarrow \begin{cases} A^1 x^1 = b^1 \\ A^2 x^2 = b^2 \\ M^1 x^1 + M^2 x^2 = d \\ x^1 \geq 0, x^2 \geq 0 \\ (\max) f = c^1 x^1 + c^2 x^2 \end{cases}$$

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

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

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 și 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 și 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 și 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 și 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:

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

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

$$A^1 x^1 \leq b^1 \quad A^2 x^2 \leq b^2 \quad (4.1)$$

și în plus:

$$x^1 \geq 0 \quad x^2 \geq 0 \quad (4.2)$$

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:

$$M^1x^1 + M^2x^2 \leq d \quad (4.3)$$

Venitul total pe economie are expresia:

$$f = c^1x^1 + c^2x^2 \quad (4.4)$$

Reunind (4.1 - 4.4) obținem programul liniar:

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

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

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

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

dar nu rezolvă problema repartizării resurselor comune deoarece, dacă \bar{x}^1 și \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 și nici agenții nu dețin informații complete asupra programului!

Vom presupune că:

- între autoritatea centrală și agenți există o **cooperare** în sensul unui schimb de informații privind "intențiile" de acțiune;
- autoritatea centrală își asumă rolul de **arbitru** în următorul sens: ea "anunță" un sistem de prețuri pe resursele comune iar agenții iau aceste prețuri ca date și își diminuează veniturile cu valoarea resurselor comune solicitate. Fie u vectorul (linie) al acestor prețuri. Atunci:
 - agentul 1, pentru susținerea unui program posibil x^1 (posibil $\equiv A^1x^1 \leq b^1, x^1 \geq 0$), 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 ($A^2x^2 \leq b^2, x^2 \geq 0$), 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:

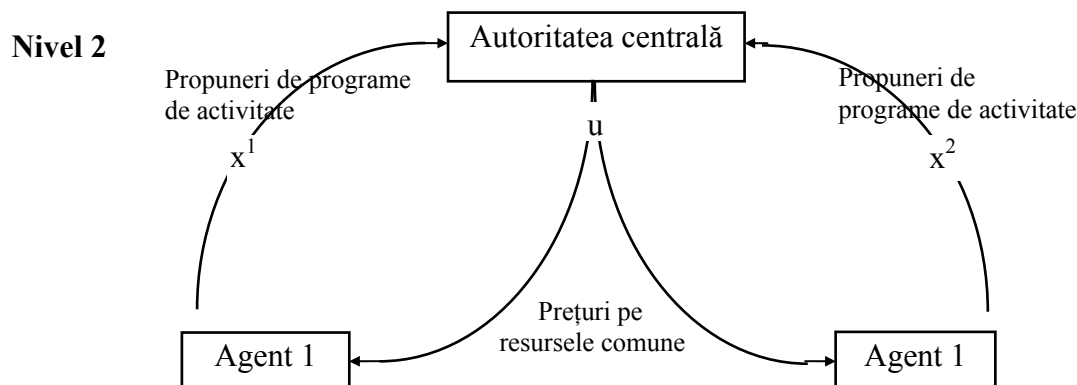
$$P_1(u) \begin{cases} A^1x^1 \leq b^1 \\ x^1 \geq 0 \\ (\max) \tilde{f}_1 = (c^1 - uM^1)x^1 \end{cases} \quad P_2(u) \begin{cases} A^2x^2 \leq b^2 \\ x^2 \geq 0 \\ (\max) \tilde{f}_2 = (c^2 - uM^2)x^2 \end{cases}$$

Agenții comunică autorității centrale propunerile optime \tilde{x}^1 și \tilde{x}^2 . În principiu, autoritatea centrală analizează oportunitatea luării în considerare a acestor propuneri pentru maximizarea venitului pe economie și poate decide modificarea sistemului de prețuri, mărind prețurile resurselor comune "intens" solicitate.

Noile prețuri sunt comunicate agenților; aceștia vor căuta noi soluții care să le maximizeze veniturile nete "corectate". Evident, prin creșterea prețurilor 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 \equiv schimburi de informații" între agenți și autoritatea centrală, vor rezulta soluțiile x^{1*} și x^{2*} care maximizează venitul total pe economie. În general, x^{1*} și x^{2*} nu coincid cu una sau alta dintre propunerile agenților (propuneri făcute în cadrul dialogului sus amintit) ci sunt **combinații convexe** ale acestora. Tot odată va rezulta și un sistem u^* de prețuri pe resursele comune în raport cu care x^{1*} și x^{2*} maximizează veniturile nete individuale ale agenților! Spunem că tripletul (x^{1*}, x^{2*}, u^*) reprezintă un **echilibru** pentru economia (liniară) considerată.

Dialogul dintre autoritatea centrală și agenți poate fi reprezentat astfel:



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. Algoritm 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 cunoaște 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} și 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**:

$$\begin{aligned}\bar{\gamma}_k &= \pi \begin{bmatrix} 1 \\ Q^k \end{bmatrix} - \gamma_k = [\pi_0, u] \begin{bmatrix} 1 \\ Q^k \end{bmatrix} - \gamma_k = \pi_0 + uQ^k - \gamma_k = \\ &= \pi_0 + uMv^k - cv^k = \pi_0 - (c - uM)v^k \quad k = 1, \dots, s\end{aligned}$$

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

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

Se observă că $\max_{k=1, \dots, s} (c - uM)v^k$ 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 cunoașterea apriorică** a vârfurilor v^1, \dots, 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 $P(u)$ și \tilde{f}^* valoarea funcției obiectiv \tilde{f} în v^* . v^* este unul din vârfurile mulțimii A și putem scrie:

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

astfel că:

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

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 $Q^* = Mv^*$, $\gamma_* = cv^*$ și 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 părăsește baza actuală și 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 cunoașterea 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 ceea ce privește 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 $Mx = d$ și/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 și x_2 nivelele de operare ale celor două activități. Resursele specifice controlate de către cei doi agenți limitează nivelele activităților după cum urmează:

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

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

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

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

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

Partiționăm sistemul restricțiilor în două blocuri:

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

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

Principiul de descompunere propune rezolvarea programului:

$$\left\{ \begin{array}{l} \sum_{k=1}^s \lambda_k = 1 \\ \sum_{k=1}^s \lambda_k Q^k \leq \begin{bmatrix} 9 \\ 7 \end{bmatrix} = d \\ \lambda_k \geq 0 \quad k = 1, \dots, s \\ (\max) f = \sum_{k=1}^s \gamma_k \lambda_k \end{array} \right.$$

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:

$$A = \left\{ x = (x_1, x_2) \mid \begin{cases} Ax \leq b \\ x \geq 0 \end{cases} \Leftrightarrow 0 \leq x_1 \leq 4, 0 \leq x_2 \leq 3 \right\}$$

Este evidentă mărginirea mulțimii A .

O dată obținută soluția optimă $\{\lambda_k^*, k = 1, \dots, s\}$ 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 formeii standard:

$$(PM) \left\{ \begin{array}{l} \sum_{k=1}^s \lambda_k = 1 \\ \sum_{k=1}^s \lambda_k Q^k + \begin{bmatrix} y_1 \\ y_2 \end{bmatrix} = \begin{bmatrix} 9 \\ 7 \end{bmatrix}, \\ \lambda_k \geq 0 \quad k = 1, \dots, s; \quad y_1, y_2 \geq 0 \\ (\max) f = \sum_{k=1}^s \gamma_k \lambda_k \end{array} \right.$$

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

Programul (PM) are trei restricții și se vede ușor că matricea sa conține coloanele unitare

$\begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix}$ și $\begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}$ corespunzătoare variabilelor y_1, y_2 . Pentru o bază unitară de start ne-ar trebui și

coloana $\begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}$ care nu este tot așa de "vizibilă". Am putea introduce o variabilă artificială în restricția

de convexitate $\sum_{k=1}^s \lambda_k = 1$, dar putem proceda, mai simplu, și astfel:

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

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 variabilelor λ_1, y_1 și y_2 . Toate aceste variabile au coeficienți nuli în funcția obiectiv așa 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 bazeice are componentele:

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

Valoarea funcției obiectiv este:

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

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 și 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 și 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, își vor maximiza veniturile "plătind" pentru resursele comune solicitate. "Propunerile" de programe de activitate sunt "comunicate" de agenți autorității centrale. Aceasta preia propunerile și încearcă, pe

baza lor și a propunerilor “mai vechi”, să construiască o “mixtură” care să se încadreze în disponibilul limitat de resurse comune și 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:

$$P(u = (0,0)) \begin{cases} x_1 \leq 4 \\ x_2 \leq 3 \\ x_1 \geq 0, x_2 \geq 0 \\ (\max)\tilde{f} = 7x_1 + 6x_2 \end{cases}$$

cu soluția optimă evidentă:

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

pe care o trimite “ca propunere de program de activitate” autorității centrale. Deoarece:

$$\pi_0 - \tilde{f}^* = 0 - 46 < 0$$

soluția din tabelul (T₁) nu este optimă; baza curentă trebuie schimbată prin introducerea unei noi coloane din matricea programului (PM). Această coloană se generează astfel:

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

Calculăm:

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

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

Calcululele 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}	$\begin{bmatrix} 1 \\ Q^2 \end{bmatrix}$
	$\begin{bmatrix} 1 \\ 17 \\ 11 \end{bmatrix}$
	$-46 \leftarrow \pi_0 - \tilde{f}^*$

 \Rightarrow (T₂)

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

\uparrow $\underbrace{\hspace{2cm}}_u$
 π_0

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”:

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

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:

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

(La “prețurile actuale” agentul 1 are un venit net pozitiv; el își poate permite să procure resursele R_1 și 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 și R_2 depășește venitul său “brut” astfel că, pentru agentul 2, decizia va fi “să nu facă nimic”.)

Deoarece :

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

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 “îmbunătățește” baza curentă:

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

Intră în bază coloana:

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 și model matematic

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

Presupunem că L și l_1, l_2, \dots, l_m sunt exprimate prin numere **întregi, pozitive** și că $L > l_1 > l_2 > \dots > l_m$. 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 $a = (a_1, a_2, \dots, a_m)$ 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 depășește lungimea suportului, astfel că:

$$l_1 a_1 + l_2 a_2 + \dots + l_m a_m \leq L$$

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

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

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

$$(P) \left\{ \begin{array}{l} \text{Sa se determine } x_1, x_2, \dots, x_n \text{ care minimizeaza functia:} \\ \quad z = x_1 + x_2 + \dots + x_n \\ \text{cu restrictiile:} \\ \quad a^1 x_1 + a^2 x_2 + \dots + a^n x_n = b \Leftrightarrow \sum_{j=1}^n a_{ij} x_j = b_i \quad i = 1, \dots, m \\ \text{si conditiile explicite impuse variabilelor:} \\ \quad x_1, x_2, \dots, x_n \geq 0 \text{ intregi} \end{array} \right.$$

Deoarece printre rețetele a^1, a^2, \dots, a^n se numără și rețetele **unitare**:

$$(1, 0, 0, \dots, 0), (0, 1, 0, \dots, 0), \dots, (0, 0, 0, \dots, 1)$$

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

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

$$w(a) = L - (l_1 a_1 + l_2 a_2 + \dots + l_m a_m)$$

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

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

unde A^1, A^2, \dots, A^N este (sub)lista rețetelor maximale, iar y_1, y_2, \dots, y_N sunt multiplicățile 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 aceeași 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 rețere pot fi croite "în exces".

Programele întregi (P) și (P') sunt în esență echivalente în sensul că au același 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 rețere cu lungimile $l_1 = 11, l_2 = 7, l_3 = 4$ din suporturi cu lungimea $L = 19$. În următorul tabel sunt indicate toate rețetele de croire și 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 și nemaximale - conduce la modelul:

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

- 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 și cu un efort computațional rezonabil - o soluție "bună" nu neapărat optimală. Așa 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, \dots, 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 și ne semnificative.

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 situațiile reale) și care - în cazul rezolvării "obișnuite" - 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:

$e^1 = (1, 0, \dots, 0)^T$, $e^2 = (0, 1, \dots, 0)^T$, ..., $e^m = (0, 0, \dots, 1)^T$ cu tabelul redus:

e^1	b_1	1			
e^2	b_2		1		
\vdots	\vdots			\ddots	
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:

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

$$r_1 = \left\lfloor \frac{L}{l_1} \right\rfloor, \quad r_2 = \left\lfloor \frac{L}{l_2} \right\rfloor, \quad \dots, \quad r_m = \left\lfloor \frac{L}{l_m} \right\rfloor$$

și cu tabelul simplex redus:

K^1	b_1/r_1	$1/r_1$		
K^2	b_2/r_2		$1/r_2$	
\vdots	\vdots			\ddots
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**, $\bar{x} = \begin{bmatrix} \bar{b} \\ 0 \end{bmatrix} \leftarrow \begin{matrix} x^B \\ x^S \end{matrix}$ soluția asociată bazei B. Presupunem disponibil tabelul simplex corespunzător; vezi tabelul 6.4

Reamintim că:

$$\begin{aligned} \bar{b} &= B^{-1} \cdot b \\ \pi &= c^B \cdot B^{-1} = [1, 1, \dots, 1] \cdot B^{-1} \\ \bar{f} &= c^B \cdot B^{-1} \cdot b = \pi \cdot b \end{aligned}$$

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

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

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

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

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

$$R(\pi) \begin{cases} \max & \pi_1 a_1 + \pi_2 a_2 + \dots + \pi_m a_m \\ & l_1 a_1 + l_2 a_2 + \dots + l_m a_m \leq L \\ & a_1, a_2, \dots, a_m \geq 0 \text{ intregi} \end{cases}$$

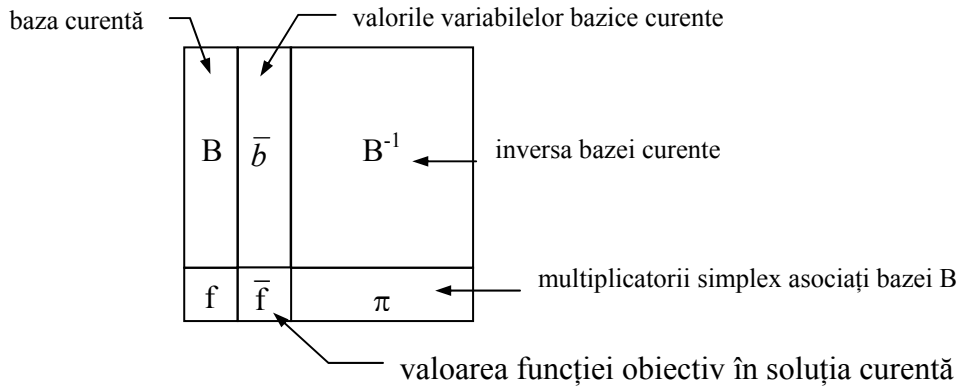
Dacă maximul funcției obiectiv din $R(\pi)$ este ≤ 1 este clar că $\bar{c}_j = \pi \cdot a^j - 1 \leq 0$ pentru toți $j = 1, \dots, n$ și soluția \bar{x} asociată bazei B este optimă.

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, \dots, 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} și un nou tabel simplex redus în care apare un nou vector π' de multiplicatori simplex. Pentru a testa optimalitatea noii soluții rezolvăm programul $R(\pi')$ etc.

Procesul iterativ se încheie într-un număr **finit** de pași 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 rețelele unicat(6.1) și cu tabelul simplex redus 6.3. Fie B baza curentă și $\pi = c^B \cdot B^{-1}$ vectorul multiplicatorilor simplex corespunzători.

Conținutul unei iterații:

Pasul 1 Se rezolvă problema auxiliară:

$$R(\pi) \left\{ \begin{array}{l} \max \quad \sum_{i=1}^m \pi_i a_i \\ \sum_{i=1}^m l_i a_i \leq L \\ a_i \geq 0 \text{ intregi } \quad i = 1, \dots, m \end{array} \right.$$

(vezi secțiunea următoare în ceea ce 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 rețelele de croire ale problemei...De altfel, diferitele rețele folosite de algoritm vor avea o notare diferită de cea din tabelul 6.1)

Start. Plecăm cu baza formată din rețelele unicat:

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

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

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

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

(celelalte rețete - pe care de fapt nu le știm - 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: $\bar{f} = \pi \cdot b = 1 \cdot 12 + 1/2 \cdot 18 + 1/4 \cdot 30 = 28 1/2$.

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 :

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

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

A^1	\rightarrow	1	1	0	<div style="border: 1px solid black; padding: 2px; display: inline-block;">coloana $B^{-1} \cdot A^1$</div>	
K^1	12	1	0	0	1	\equiv pivot
K^2	9	0	1/2	0	1/2	\Rightarrow
K^3	15/2	0	0	1/4	0	
f	57/2	1	1/2	1/4	1/2	$= (\max)\rho - 1$

Tabelul 6.6a

Tabelul 6.6b

Iterația 2 Rezolvăm problema:

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

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

$A^2 \rightarrow$	0	2	1	coloana $B^{-1} \cdot A^2$		
A^1	12	1	0	0	0	
K^2	3	-1/2	1/2	0	1	\equiv pivot \Rightarrow
K^3	15/2	0	0	1/4	1/4	
f	45/2	1/2	1/2	1/4	1/4	$= (\max)\rho - 1$

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

Tabelul 6.7a

Tabelul 6.7b

Iterația 3 Acum se rezolvă problema:

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

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

$A^3 \rightarrow$	1	0	2	coloana $B^{-1} \cdot A^3$		
A^1	12	1	0	0	1	
A^2	3	-1/2	1/2	0	-1/2	\Rightarrow
K^3	27/4	1/8	-1/8	1/4	5/8	\equiv pivot
f	87/8	5/8	3/8	1/4	1/8	$= (\max)\rho - 1$

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

Tabelul 6.8a

Tabelul 6.8b

Iterația 4 Rezolvăm problema:

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

De această dată $(\max)\rho = 1$ 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 suportilor "consumați" este : $\frac{102}{5} = 20\frac{2}{5}$.

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

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

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 "încapă" în b' ;
- se actualizează b' prin extragerea rețetei găsite și 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 rețere cerute ar fi următoarea:

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

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

Concluzii finale

1. În soluția optimă a problemei relaxate se utilizează numai rețete maxime;
2. După aplicarea euristicii FFD asupra cererii reziduale, pot apărea și 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 întregă superioară a optimului fracționar și numai în rare cazuri este mai mare decât aceasta cu exact o unitate!

§7 Programare dinamică

În această secțiune ne vom opri asupra problemei :

$$(R) \begin{cases} \max & \rho = \pi_1 a_1 + \pi_2 a_2 + \dots + \pi_m a_m \\ & l_1 a_1 + l_2 a_2 + \dots + l_m a_m \leq L \\ & a_1, a_2, \dots, a_m \geq 0, \text{ intregi} \end{cases}$$

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

În secțiunea precedentă, (R) a apărut ca **subproblemă** în rezolvarea relaxatei probleme 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 **program 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 și 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 și 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 $k = 1, \dots, m$ și fiecare întreg $\lambda = 0, 1, \dots, L$ considerăm problema:

$$R_k(\lambda) \begin{cases} \max & \pi_1 a_1 + \pi_2 a_2 + \dots + \pi_k a_k \\ & l_1 a_1 + l_2 a_2 + \dots + l_k a_k \leq \lambda \\ & a_1, a_2, \dots, a_k \geq 0, \text{ intregi} \end{cases}$$

al cărei optim îl notăm cu $\rho_k(\lambda)$. Este clar că $R = R_m(\lambda)$ și 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 $0, 1, \dots, L$.

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

- $$\rho_1(\lambda) = \max \{ \pi_1 a_1 \mid l_1 a_1 \leq \lambda \} = \pi_1 \left\lfloor \frac{\lambda}{l_1} \right\rfloor \quad \lambda = 0, 1, \dots, L \quad (7.1)$$

- pentru $k > 1$ avem formula de recurență:

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

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

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

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

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

$$a_m^* = a_m^*(L)$$

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

$$a_k^* = a_k^*(L - S_k)$$

unde :

$$S_k = l_{k+1} a_{k+1}^* + \dots + l_m a_m^*$$

Observații: 1) În termenii problemei rucsacului $\rho_k(\lambda)$ este valoarea maximă a unei încărcări a rucsacului ce nu depășește în greutate plafonul λ și 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 $\rho_1, \rho_2, \dots, \rho_{m-1}, \rho_m$ 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 și decizia inițială, deciziile rămase constituie o strategie optimă în raport cu starea care rezultă din prima decizie.**

Demonstrația formulei (7.2)

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

$$\begin{aligned} \rho_{k-1}(\lambda - l_k \bar{a}_k) &= \pi_1 \bar{a}_1 + \dots + \pi_{k-1} \bar{a}_{k-1} \\ l_1 \bar{a}_1 + \dots + l_{k-1} \bar{a}_{k-1} &\leq \lambda - l_k \bar{a}_k \end{aligned}$$

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

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

În consecință:

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

și cum \bar{a}_k a fost arbitrar aleasă (între 0 și $\left\lfloor \frac{\lambda}{l_k} \right\rfloor$) urmează că:

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

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

$$\begin{aligned} \rho_k(\lambda) &= \pi_1 a_1^\bullet + \dots + \pi_{k-1} a_{k-1}^\bullet + \pi_k a_k^\bullet \\ l_1 a_1^\bullet + \dots + l_{k-1} a_{k-1}^\bullet + l_k a_k^\bullet &\leq \lambda \end{aligned}$$

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

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

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

$$\begin{aligned} \pi_1 a_1^{\bullet\bullet} + \dots + \pi_{k-1} a_{k-1}^{\bullet\bullet} &= \rho_{k-1}(\lambda - l_k a_k^\bullet) > \pi_1 a_1^\bullet + \dots + \pi_{k-1} a_{k-1}^\bullet \\ l_1 a_1^{\bullet\bullet} + \dots + l_{k-1} a_{k-1}^{\bullet\bullet} &\leq \lambda - l_k a_k^\bullet \rightarrow l_1 a_1^{\bullet\bullet} + \dots + l_{k-1} a_{k-1}^{\bullet\bullet} + l_k a_k^\bullet \leq \lambda \end{aligned}$$

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

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

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

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

astfel că:

$$\begin{aligned} \rho_k(\lambda) &= \pi_1 a_1^* + \dots + \pi_{k-1} a_{k-1}^* + \pi_k a_k^* = \rho_{k-1}(\lambda - l_k a_k^*) + \pi_k a_k^* \leq \\ &\leq \max\left\{ \rho_{k-1}(\lambda - l_k a_k) + \pi_k a_k \mid a_k = 0, 1, \dots, \left\lfloor \frac{\lambda}{l_k} \right\rfloor \right\} \end{aligned}$$

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 $\lambda = 0, 1, \dots, 11$

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
$a_1^*(\lambda)$	0	0	0	0	0	1	1	1	1	1	2	2

Tabelul 7.1

Iterația 2 În continuare calculăm valorile funcției

$$\begin{aligned} \rho_2(\lambda) &= \max\left\{ \rho_1(\lambda - l_2 a_2) + \pi_2 a_2 \mid a_2 = 0, 1, \dots, \left\lfloor \frac{\lambda}{l_2} \right\rfloor \right\} = \\ &= \max\left\{ \rho_1(\lambda - 4a_2) + 12a_2 \mid a_2 = 0, 1, \dots, \left\lfloor \frac{\lambda}{4} \right\rfloor \right\} \text{ unde } \lambda = 0, 1, \dots, 11 \end{aligned}$$

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

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

iar pentru $\lambda = 8, 9, 10, 11$

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

Rezultatele sunt afișate 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
$a_2^*(\lambda)$	0	0	0	0	1	0	0	0	2	1	0	0

Tabelul 7.2

Iterația 3 În final vom evalua numai:

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

$$\max\{\rho_2(11), \rho_2(9) + 5, \rho_2(7) + 10, \rho_2(5) + 15, \rho_2(3) + 20, \rho_2(1) + 25\} =$$

$$\max\{32, 28 + 5, 16 + 10, 0 + 20, 0 + 25\} = 33 \rightarrow a_3^* = 1$$

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

Pasul 1 $a_3^* = a_3^*(11) = 1;$

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

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

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

§ 8. Întrebări și 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:

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

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

- \bar{x} este soluția optimă a programului

$$P(\bar{u}) \begin{cases} Ax \leq b \\ x \geq 0 \\ (max)f' = (c - \bar{u}M)x \end{cases}$$

- $M\bar{x} \leq d$
- $\bar{u}(d - M\bar{x}) = 0$

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 cu structură bloc - diagonală și restricții de cuplare:

$$\begin{array}{l}
 a) \left\{ \begin{array}{l}
 2x_1 + 3x_2 \leq 6 \\
 5x_1 + x_2 \leq 5 \\
 3x_3 + 4x_4 \geq 12 \\
 x_3 \leq 4 \\
 x_4 \leq 3 \\
 x_1 + 4x_2 + 5x_3 + 2x_4 \leq 7 \\
 x_j \geq 0 \quad j = 1, \dots, 4 \\
 (\max)f = x_1 + 8x_2 + 5x_3 + 6x_4
 \end{array} \right.
 \end{array}
 \qquad
 \begin{array}{l}
 b) \left\{ \begin{array}{l}
 x_1 + 3x_2 \leq 30 \\
 2x_1 + x_2 \leq 20 \\
 x_3 + x_4 \leq 15 \\
 x_3 \leq 10 \\
 x_4 \leq 10 \\
 x_1 + 2x_2 + 2x_3 + x_4 \leq 40 \\
 x_j \geq 0 \quad j = 1, \dots, 4 \\
 (\max)f = x_1 + x_2 + 2x_3 + x_4
 \end{array} \right.
 \end{array}$$

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 și 30 de țevi cu lungimea de 3,50m. Aceste bucăți se taie din țevi cu lungimea de 9m.

- Alcătuți lista rețetelor maximale de croire;
- Scrieți un program liniar în numere întregi pentru minimizarea numărului de țevi de 9m ce vor fi tăiate;
- Rezolvați programul relaxat prin metoda generării de coloane;
- 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 și model matematic. Descrieți algoritmul de rezolvare al problemei rucsacului prin programare dinamică.

7. Rezolvați problemele de tip rucsac:

$$\begin{array}{l}
 a) \left\{ \begin{array}{l}
 (\max)\rho = 5a_1 + 12a_2 + 16a_3 \\
 2a_1 + 4a_2 + 5a_3 \leq 11
 \end{array} \right.
 \end{array}
 \qquad
 \begin{array}{l}
 b) \left\{ \begin{array}{l}
 (\max)\rho = a_1 + 3a_2 + 5a_3 + 9a_4 \\
 2a_1 + 3a_2 + 4a_3 + 7a_4 \leq 10
 \end{array} \right.
 \end{array}$$