CAPITOLUL I

PROGRAMARE ÎN NUMERE ÎNTREGI

Capitolele 1 şi 2 ale cursului de Cercetări Operaţionale din anul III au ca suport Notele de curs ale Domnului Profesor Vasile Nica şi lucrarea "Capitole Speciale ale Cercetărilor Operaţionale" a aceluiaşi autor, editată de Centrul de Învăţământ Economic Deschis la Distanţă din Academia de Studii Economice Bucureşti.

1.1 Definirea unei probleme de programare liniară în numere întregi

Să se determine $x_1, x_2, \ldots, x_n$ care maximizează funcţia:

$$f = c_1x_1 + c_2x_2 + \ldots + c_nx_n \qquad (1)$$

cu satisfacerea restricţiilor:

$$\begin{cases} a_{11}x_1 + a_{12}x_2 + \ldots + a_{1n}x_n = b_1 \\ a_{21}x_1 + a_{22}x_2 + \ldots + a_{2n}x_n = b_2 \\ \quad\vdots \\ a_{m1}x_1 + a_{m2}x_2 + \ldots + a_{mn}x_n = b_m \end{cases} \qquad (2)$$

a restricţiilor de nenegativitate:

$$x_j \geq 0 \text{ , oricare ar fi } j = 1,\ldots, n \qquad (3)$$

şi a condiţiilor de integritate:

$$x_j \text{ , întregi } j = 1, ..., n \text{ ,} \qquad (4)$$

În (1) şi (2) coeficienţii b_1 ,…, b_n şi a_{11}, a_{12}, … , a_{mn} se vor presupune în mod constant *întregi*. Cerinţa nu este deloc restrictivă pentru că, în mod obişnuit, datele, oricare ar fi problema, sunt numere raţionale.

Problemele (1)–(4) = (**I**nteger **L**inear **P**rogramming) se prezintă în forma *standard*. Considerarea aproape în exclusivitate a acestei forme nu constituie o restrângere a generalităţii noastre pentru că oricare ar fi inecuaţia (cu coeficienţi întregi!), ea poate fi transformată în egalitate prin introducerea unei variabile de abatere în maniera binecunoscută din programarea liniară. Este evident că şi variabilele de abatere vor satisface restricţiile de nenegativitate.

Ignorând condiţia de integritate, primele trei condiţii formează o problemă standard de programare liniară denumită în continuare (LP).

Pentru comoditatea scrierii adaptăm notațiile matriciale uzuale:

$$\textbf{(ILP)}\begin{cases}(\max) f = c\cdot x\\ A\cdot x = b\\ x \geq 0\\ x\ \hat{I}NTREG\end{cases} \qquad \textbf{(LP)}\begin{cases}(\max) f = c\cdot x\\ A\cdot x = b\\ x \geq 0\end{cases}$$

Să punem în evidență mulțimea de Soluții Admisibile ale celor două probleme:

$$\left.\begin{array}{l} A_{ILP} = \{\, x \in R^n \,\mathbb{R}\, A\cdot x = b\,,\, x \geq 0\,,\, x \text{ întreg}\}\\ A_{LP} \ = \{\, x \in R^n \,\mathbb{R}\, A\cdot x = b\,,\, x \geq 0\,\} \end{array}\right\} \Rightarrow A_{ILP} \subset A_{LP}$$

De aici rezultă că (LP) este o RELAXARE a lui (ILP).

Să presupunem pentru moment că ambele probleme au **S**oluții **O**ptime finite. În mod constant vom nota cu $\boldsymbol{x^0}$ soluția optimă a programului ILP (Soluție Optimă Întreagă), în timp ce soluția optimă a programului LP este $\boldsymbol{x^*}$ (**Soluție Optimă Neîntreagă – Fracționară**).

avem $\boldsymbol{f(x^0) \leq f(x^*)}$

Este posibil ca x^* să aibă toate componentele întregi $\Rightarrow \boldsymbol{x^* = x^0}$. Condiția de integritate (4) complică enorm rezolvarea programului (ILP). Vom sublinia acest lucru printr-o paralelă între trăsăturile cele mai importante ale Mulțimilor de Soluții Admisibile ale celor două probleme: (LP) şi (ILP).

Exemplu numeric:

$$\textbf{(ILP)}\begin{cases}(\max) f = 3x_1 - x_2\\ 3x_1 - 2x_2 \leq 3\\ 5x_1 + 4x_2 \geq 10\\ 2x_1 + \ x_2 \leq 5\\ x_{1,2} \geq \text{Ø}\\ x_{1,2} \in \mathbf{Z}\end{cases} \qquad \textbf{(LP)}\ \text{relaxata}\begin{cases}(\max) f = 3x_1 - x_2\\ 3x_1 - 2x_2 \leq 3\\ 5x_1 + 4x_2 \geq 10\\ 2x_1 + \ x_2 \leq 5\\ x_{1,2} \geq \text{Ø}\end{cases}$$

Figura 1 pune în evidență mulțimea soluțiilor admisibile A_{LP} ca fiind poligonul convex ABCD. Soluția optimă x^* se află într-un punct extrem al acestui poligon. Reprezentând grafic curbele de nivel ale funcției obiectiv se deduce că: $x^* \equiv A$ adică $\begin{bmatrix} x_1^* = 13/7 \\ x_2^* = 9/7 \end{bmatrix}$, $f(x^*) = 30 / 7$.

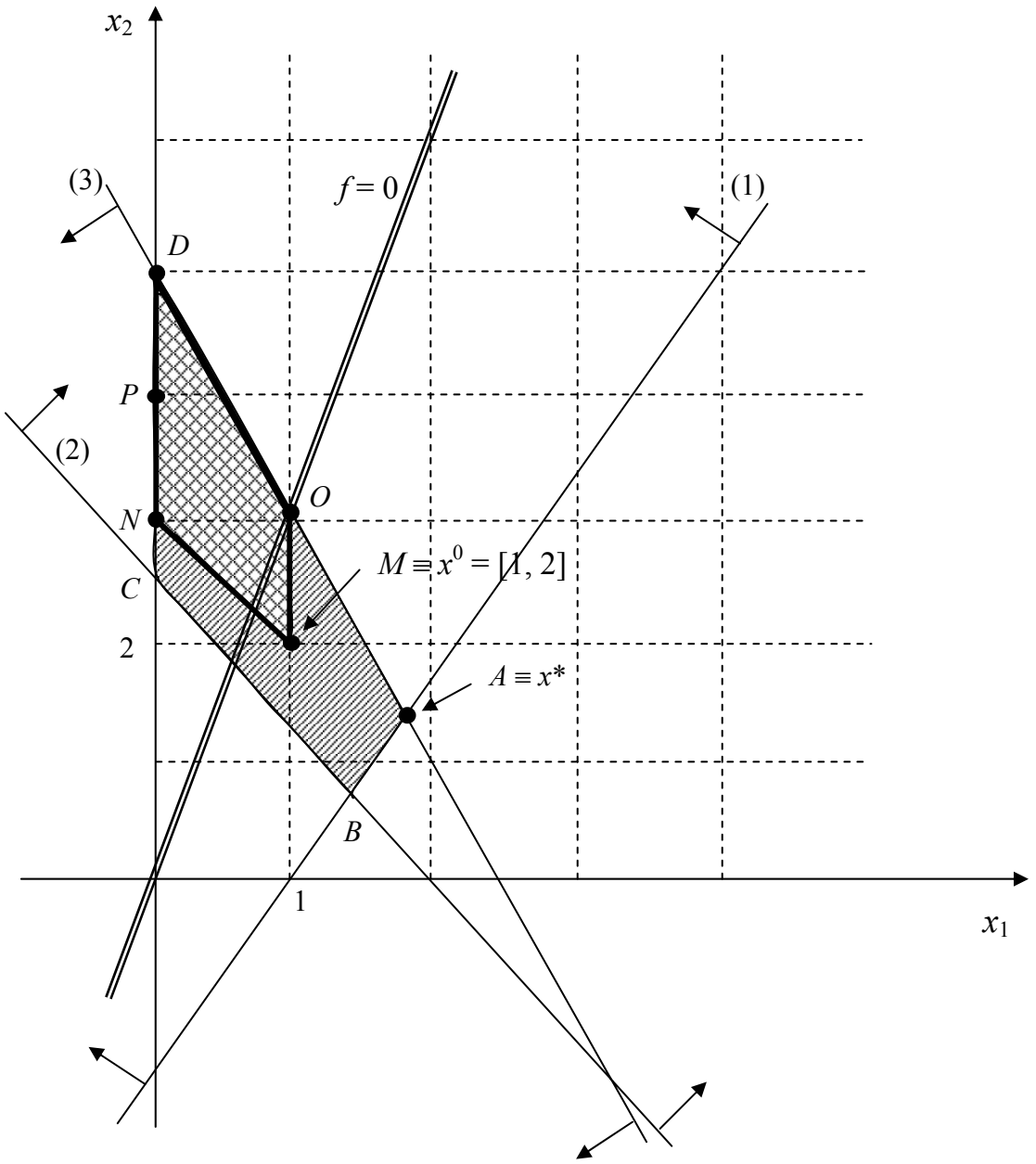


Figura 1

Prin inspectarea directă se conclude că A_{ILP} este formată din punctele M(1,2), N(0,3), P(0,4), D(0,5), O(1,3). Valoarea cea mai mare a funcției obiectiv se atinge în (1, 2) $\Rightarrow x^0 \equiv M$ avem $x^0{}_1=1$, $x^0{}_2 = 2$ cu funcția obiectiv $f(x^0) = 1$.

Observație:

Simpla rotunjire a componentelor optimului fracționar nu conduce în mod obligatoriu la optimul întreg. În cazul de față prin rotunjire se obține punctul $x(2,1)$ care nu este o soluție admisibilă.

Figura 1 pune în evidență şi un alt aspect interensant. Să considerăm ***cea mai mică mulțime convexă care conține soluții admisibile întregi ale problemei ILP***. Nu este greu de văzut că această mulțime este reprezentată de poligonul MNDO. Vârfurile acestui poligon sunt, prin construcție, soluții admisibile întregi şi dacă maximizăm pe f pe această mulțime regăsim optimul întreg x^0 !

Prin urmare putem rezolva o problemă de programare liniară întreagă ca o problemă de programare liniară obişnuită cu condiția să ştim să descriem în limbaj ecuațional **ÎNFĂŞURĂTOAREA CONVEXĂ** a mulțimii soluțiilor sale întregi!

Reîntorcându-ne la cazul general dar inspirându-ne tot timpul din acest exemplu notăm următoarele:

1) În timp ce A_{LP} formează o mulțime convexă şi închisă în $\boldsymbol{R}^n$ (n = numărul variabilelor problemei), mulțimea A_{ILP} este doar o muțime discretă putând avea chiar un număr finit de elemente. ILP poate să nu aibă soluții admisibile (întregi) chiar dacă LP are!

   Exemplu:

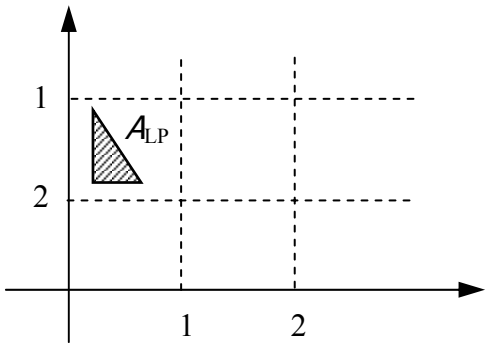


Figura 2

2) Trebuie explicat de la bun început de ce acordăm importanță programării în numere întregi. Este evident faptul că pentru foarte multe probleme variabilele sunt supuse restricțiilor de integritate. Un mod destul de realist de abordare a acestor probleme constă în a rezolva problema fără a ține seama de condițiile de integritate după care rotunjesc componentele fracționare ale soluțiilor optime de aşa manieră încât restricțiile să fie respectate. Această modalitate este frecvent folosită în practică şi este perfect justificată în situația în care valorile permisibile ale variabilelor sunt suficient de mari astfel că efectul rotunjirii este neglijabil. Există însă probleme de programare liniară în numere întregi deosebit de importante în care variabilele iau valori întregi destul de mici adesea **0** sau **1**. Pentru asemenea probleme "distanța" dintre optimul fracționar şi cel întreg s-ar putea să fie atât de mare încât rotunjirea să nu ducă la o soluție acceptabilă.

3) Teoretic, orice problemă de programare liniară în numere întregi este echivalentă cu o problemă de programare liniară uzuală adică în variabile continue dacă mulțimea soluțiilor întregi se înlocuieşte cu înfăşurătoarea sa convexă. Dificultatea abordării problemei din acest unghi rezidă în enorma greutate de a ***descrie ecuațional înfăşurătoarea convexă***.

4) Presupunem că într-o problemă de programare liniară în numere întregi apare restricția:

$$1/2x_1 + 2/3x_2 + 3/4x_3 \le 17/5$$

Transformăm într-o inegalitate cu coeficienți ÎNTREGI prin înmulțirea cu c.m.m.m.c. al numitorilor care este 60 $\Rightarrow 30x_1 + 40x_2 + 45x_3 \le 204 \Rightarrow$ forma STAS:

$30x_1 + 40x_2 + 45x_3 + y = 204$

$x_{1,2,3} \in \mathbf{Z} \Rightarrow y \in \mathbf{Z}$ (ca diferență de numere întregi).

1.2 Exemple

1. Problema croirii

Un număr de repere uni sau bidimensionale trebuie executate în cantități date. Aceste repere se obțin prin decupare din nişte suporți (bare sau foi). Acoperirea unui suport cu diferite repere se poate face în mai multe moduri (bineînțeles că este exclusă orice suprapunere totală sau parțială a reperelor).

O modalitate de acoperire a unui suport poartă numele de REȚETĂ DE CROIRE. După decuparea reperului din suport conform unei rețete rămâne un REST ce nu mai poate fi utilizat. Problema constă în a alege acele rețete de croire prin care **să se obțină reperele în cantitățile dorite cu MINIMUL POSIBIL de rest**.

FORMALIZAREA MATEMATICĂ

Să începem prin a descrie modelul UNIDIMENSIONAL de croire. Reperele vor fi identificate prin nişte numere întregi reprezentând lungimea lor într-o unitate de masură adecvată.

Fie ***m*** = numărul de repere distincte care trebuie tăiate şi

- $l_1, l_2, \ldots, l_m$ = lungimile lor;
- L = lungimea suportului din care se taie reperele.

O rețetă de croire se identifică printr-un vector m dimensional cu componente întregi ($a_1, a_2, \ldots, a_m$), unde a_i = numărul de repere i (deci cu lungimea l_i) ce rezultă prin tăierea conform rețetei.

Evident: $a_1l_1 + a_2l_2 + \ldots + a_ml_m \le L$, iar $L - (a_1l_1 + a_2l_2 + \ldots + a_ml_m)$ = restul rețetei = r_i.

- $\rho_1, \rho_2, \ldots, \rho_n$ = rețetele posibile de croire a reperelor din suportul dat.

$$\rho_j = \begin{pmatrix} a_{1j} \\ a_{2j} \\ \ldots \\ a_{mj} \end{pmatrix}$$

- $b_1, b_2, \ldots, b_m$ = cantitățile în care reperele trebuie executate.
- $r_1, r_2, \ldots, r_n$ = resturile inutilizabile ce rezultă din aplicarea O DATĂ a rețetelor $\rho_1, \rho_2, \ldots, \rho_n$.

În activitatea de producere a cantităților necesare de repere fiecare rețetă va fi aplicată de un număr de ori, posibil zero.

- x_j = numărul de aplicări ale rețetei ρ_j = MULTIPLICITATEA rețetei ρ_j.

Restul total rezultat din aplicarea rețetelor $\rho_1, \ldots, \rho_n$ cu multiplicitățile $x_1, \ldots, x_n$ este dat de expresia:

$$f = r_1x_1 + r_2x_2 + \ldots + r_nx_n \qquad (5)$$

Condiția de realizare a cantităților cerute de repere se transcrie astfel:

$$\begin{cases} a_{11}x_1 + a_{12}x_2 + \ldots + a_{1n}x_n = b_1 \\ a_{21}x_1 + a_{22}x_2 + \ldots + a_{2n}x_n = b_2 \\ \cdot \\ \cdot \\ \cdot \\ a_{m1}x_1 + a_{m2}x_2 + \ldots + a_{mn}x_n = b_m \end{cases} \qquad (6)$$

având în vedere semnificația lor, variabilele $x_1, ..., x_n$ satisfac:

$$x_j \geq 0 \quad , \; j = 1, ..., n \qquad (7)$$
$$x_j = \text{ÎNTREGI} \quad , \; j = 1, ..., n \qquad (8)$$

Problema croirii unidimensionale constă în determinarea multiplicităților x_j care satisfac (6), (7) şi (8) şi care minimizează obiectivul (5).

Adunând (5) şi (6) membru cu membru obținem:

$$(\underbrace{\Sigma\, a_{i1} + r_1}_{L})x_1 + (\underbrace{\Sigma\, a_{i2} + r_2}_{L})x_2 + \ldots + (\underbrace{\Sigma a_{in} + r_n}_{L})\, x_m = \Sigma\, b_i + f$$

sau $\boxed{L(\, x_1 + x_2 + \ldots + x_n\,) = \Sigma\, b_i + f}$ (9)

Fie funcția $g = x_1 + x_2 + \ldots + x_n$ (10)
care indică numărul total de suporți tăiați. Relația (9) devine:

$$f = g \cdot L - \Sigma\, b_i \Leftrightarrow g = 1/L(\Sigma b_i + f) \Rightarrow (\min) f = (\min) g. \qquad (11)$$

Relația (11) arată că dacă în locul minimizării restului total se urmăreşte minimizarea numărului de suporți tăiați, se obține o problemă echivalentă în sensul că cele două probleme au aceeaşi soluție optimă.

Problema croirii bidimensionale este formal identică cu cea unidimensională. Particularitatea problemei croirii rezidă în numărul extraordinar de mare al rețetelor posibile de croire – presupuse cunoscute în formalizarea matematică. În realitate, chiar şi în cazul unidimensional mai simplu, rețetele de croire asociate unui set dat de repere şi unui anumit suport din care se croiesc acestea, nu sunt cunoscute. Generarea acestor rețete este o problemă combinatorială delicată.

2. O problemă de repartizare a investițiilor

O firmă are ***n*** proiecte pe care ar dori să le realizeze într-un răstimp de ***m*** ani, dar din cauza bugetului său limitat numai o parte pot fi alese. Proiectul ***j*** aduce firmei un câştig estimat la c_j \$ şi necesită investiții anuale în valoare de a_{ij} \$, cu $i = 1,...,$ m.

Capitalul disponibil pentru anul ***i*** este b_i. Se pune problema de a alege acele proiecte care să aducă firmei un câştig total (maxim) cu condiția nedepăşirii capitalului disponibil anual.

FORMALIZAREA MATEMATICĂ

Introducem variabilele binare $x_j = \begin{cases} 1 \text{ , dacă proiectul } j \text{ este admis;} \\ 0 \text{ , altfel.} \end{cases}$

Obținem o problemă de programare binară:

$$\begin{cases} (\max) f = \Sigma\, c_j x_j \\ \sum_{j=1}^{n} a_{ij}\, x_j \le b_i \ , \ i = 1, ..., m \\ x_j \in [0,1] \end{cases}$$

3. Problema ordonanțării reperelor pe maşini

Ipoteze: n repere ce trebuie prelucrate pe m maşini.

Fiecare reper necesită cel mult o operație pe fiecare maşină. Fie p_{ik} timpul necesar prelucrării reperului i pe maşina k. Variabilele de decizie ale problemei vor fi:

x_{ik} = momentul de începere a prelucrării reperului i pe maşina k. Se presupune că programarea începe de la momentul 0.

Condiții:

1. Nu se pot programa două repere pentru a fi prelucrate pe acelaşi utilaj; aceasta înseamnă că pentru oricare ar fi reperele i, j şi utilajul k: $x_{ik} - x_{jk} \ge p_{jk}$ sau $x_{jk} - x_{ik} \ge p_{ik}$, $i, j = 1,\ldots, n,$ $i \ne j$, $k = 1,\ldots, m$. (12)

2. Pentru fiecare reper operațiile trebuie executate într-o anumită ordine:

fie $r_{ijk} = \begin{cases} 1 \text{ dacă a } j\text{-a prelucrare a reperului } i \text{ se face pe utilajul } k \\ 0 \quad \text{în caz contrar.} \end{cases}$

Atunci momentul de începere al operației j la reperul i este: $\Sigma r_{ijk} x_{ik}$.

Rezultă restricția $\Sigma r_{ijk}(x_{ik} + p_{ik}) \le \Sigma r_{i,j+1,k} \cdot x_{ik}$, $j = 1,\ldots, m\text{-}1$. (13)

Există diverse funcții obiectiv. Cea obişnuită constă în (min) sumei momentelor de începere a ultimei operații la fiecare job adică:

$$(\min) f = \Sigma\Sigma r_{imk} \cdot x_{ik} \ . \qquad (14)$$

Bineînțeles dihotomia (12) se înlocuieşte cu restricții STAS prin folosirea de variabile bivalente.

Problema (12), (13), (14) este o problemă de un tip nou: ea conține atât variabile continue cât şi variabile bivalente. Spunem că este o problemă de programare MIXTĂ.

1.3 Generalități privind metode de rezolvare a problemelor de programare liniară întreagă

A) Metode de tip plan de secțiune;
B) Metode bazate pe enumerarea implicită a soluțiilor întregi;
C) Metode specifice create pentru rezolvarea unor clase de programe specifice.

Înainte de a expune bazele teoretice ale acestor metode să examinăm următorul exemplu:

$$(ILP)\ \left\{\begin{array}{l}(\max)f = 3x_1 - x_2 \\ 3x_1 - 2x_2 \le 3 \\ 5x_1 + 4x_2 \ge 10 \\ 2x_1 + x_2 \le 5 \\ x_{1,2} \ge 0 \\ x_{1,2} \text{ întregi}\end{array}\right\} (LP)$$

Ignorăm condiția de integritate şi rezolvăm cu ajutorul simplex-ului problema relaxată (LP). Adăugăm variabilele de abatere şi artificiale necesare obținând:

$$(FBLP)\ \left\{\begin{array}{l}3x_1 - 2x_2 + y_1 = 3 \\ 5x_1 + 4x_2 - y_2 + z = 10 \\ 2x_1 + x_2 + y_3 = 5 \\ (\max)f = 3x_1 - x_2 - M \cdot z, \quad \text{toate variabilele} \ge 0\end{array}\right.$$

B	CB	VVB	x_1	x_2	y_1	y_2	y_3	z
y_1	0	3	**3**	-2	1	0	0	0
z	-M	10	5	4	0	-1	0	1
y_3	0	5	2	1	0	0	1	0
	f	-10M	-3-5M	1-4M	*	M	*	*
x_1	3	1	1	-2/3	1/3	0	0	0
z	-M	5	0	**22/3**	-5/3	-1	0	1
y_3	0	3	0	7/3	-2/3	0	1	0
	f	-5M+3	*	-22/3M-1	5M/3+1	M	*	*
x_1	3	16/11	1	0	2/11	-1/11	0	
x_2	-1	15/22	0	1	-5/22	-3/22	0	
y_3	0	31/22	0	0	-3/22	**7/22**	1	
	F	81/22	*	*	17/22	-3/22	*	
x_1	3	13/7	1	0	1/7	0	2/7	
x_2	-1	9/7	0	1	-2/7	0	3/7	
y_2	0	31/7	0	0	-3/7	1	22/7	
	f	30/7	*	*	5/7	*	3/7	

$x_1 + 1/7y_1 + 2/7y_3 = 13/7$
$x_2 - 2/7y_1 + 3/7y_3 = 9/7$
$-3/7y_1 + y_2 + 22/7y_3 = 31/7$

$x_1 + 1/7y_1 + 2/7y_3 = 1 + 6/7 \Rightarrow 1/7y_1 + 2/7y_3 - 6/7 = 1 - x_1$

$1-x_1 \in Z$ (în orice *x* întreg) $\Rightarrow 1/7y_1+2/7y_3-6/7 \in Z$

Dacă $1-x_1 < 0 \Rightarrow 1- x_1 \le -1 \Rightarrow 1/7y_1 + 2/7y_3 - 6/7 \le -1 \Rightarrow 1/7y_1+2/7y_3 \le -1/7$ care contrazice ipoteza că $y_{1,3} \ge 0$. Deci $\mathbf{1-x_1 \ge 0}$ adică $\mathbf{1/7y_1+2/7y_3-6/7 \ge 0.}$

Soluția optimă fracționară curentă nu verifică această restricție; prin construcție oricare soluție întreagă o verifică.

A) Ideea metodei planului de secțiune constă în a adăuga această restricție la problema curentă şi de a reoptimiza. Prin adăugare mulțimea soluțiilor problemei relaxate LP se micşorează în timp ce mulțimea soluțiilor întregi rămâne aceeaşi. Este posibil ca prin adăugarea de asemenea restricții care taie succesiv porțiuni din poliedrul convex al soluțiilor admisibile ale problemei relaxate, A_{LP}, dar nu alterează mulțimea soluțiilor întregi, **optimul întreg să devină vârf al poliedrului pe care se face reoptimizarea curentă**. În această situație acest optim întreg va fi sigur detectat de metoda Simplex ca fiind soluția optimă a problemei de reoptimizare curente. Teoria ne asigură că este întotdeauna aşa!

Deci, în principiu, metoda planului de secțiune reduce rezolvarea unei (PLI) la o suită de procese de reoptimizare liniară prin adăugare succesivă de noi restricții, neverificate de optimul fracționar curent, dar verificate de orice soluție admisibilă întreagă.

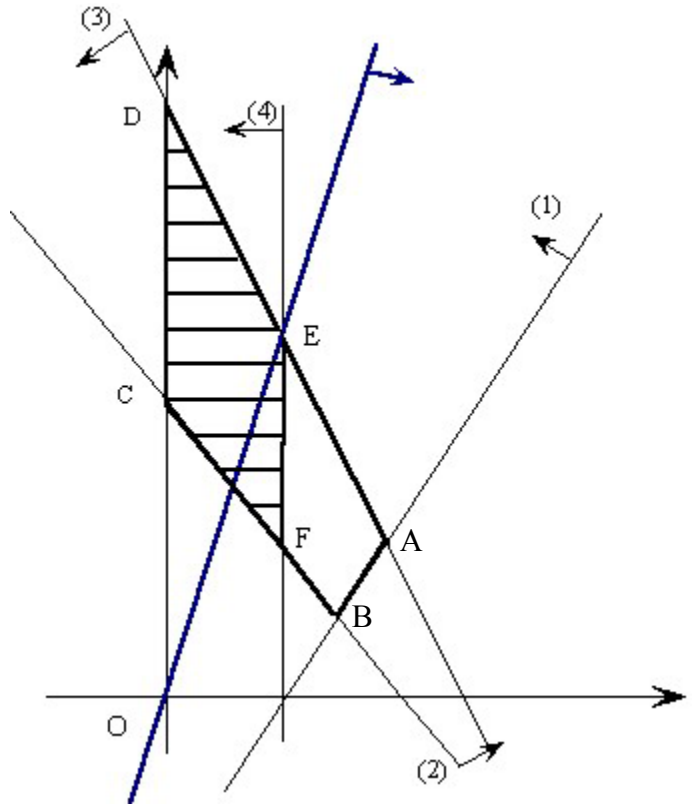
Pentru a vedea efectul introducerii restricției

$$1/7y_1+2/7y_3-6/7 \ge 0$$

asupra mulțimii soluțiilor admisibile ale problemei (LP), observăm că ea este echivalentă cu :

$1-x_1 \ge 0 \Leftrightarrow x_1 \le 1$ restricția (4)

Prin adăugarea ei, poligonul ABCD se reduce la EFCD.



Să procedăm la reoptimizare pentru determinarea soluției optime:

$y_4 = 1/7y_1 + 2/7\ y_3 - 6/7 \Leftrightarrow \ -1/7y_1 - 2/7\ y_3 + y_4 = -6/7$

După reoptimizare se obține optimul fracționar $\mathbf{F} = [x_1 = 1, x_2 = 5/4]$, cu $f = 7/4$

În continuare din ecuațiile deduse din tabelul Simplex optim alegem pe cea al cărei termen liber are partea fracționară maximă.

			3	-1	0	0	0	0
B	CB	VVB	x_1	x_2	y_1	y_2	y_3	y_4
x_1	3	13/7	1	0	1/7	0	2/7	0
x_2	-1	9/7	0	1	-2/7	0	3/7	0
y_2	0	31/7	0	0	-3/7	1	22/7	0
y_4	0	-6/7	0	0	-1/7	0	**-2/7**	1
	f	30/7	*	*	5/7	*	3/7	*
x_1	3	1	1	0	0	0	0	1
x_2	-1	0	0	1	-1/2	0	0	3/2
y_2	0	-5	0	0	**-2**	1	0	11
y_3	0	3	0	0	1/2	0	1	-7/2
	f	3	*	*	1/2	*	*	3/2
x_1	3	1	1	0	0	0	0	1
x_2	-1	5/4	0	1	0	-1/4	0	-5/4
y_1	0	5/2	0	0	1	-1/2	0	-1/2
y_2	0	**7/4**	0	0	0	1/4	1	-3/4
	f	7/4	*	*	*	1/4	*	17/4
y_5		-3/4				-1/4		-1/4

$7/4 = 1/4y_2 + y_3 - 3/4y_4$
$1+3/4 = 1/4y_2 + y_3 - y_4 + 1/4y_4 \Rightarrow$
$1/4y_2 + 1/4y_4 - 3/4 = 1 - y_3 + y_4$

$1/4y_2 + 1/4y_4 - 3/4 \geq 0$

$1 - y_3 + y_4 \geq 0 \Rightarrow$
$1 + 2x_1 + 2x_2 = 5 + 1 - x_1 \geq 0 \Rightarrow$

$x_1 + x_2 \geq 3$

Reoptimizăm:
$y_5 = 1/4y_2 + 1/4y_4 - 3/4 \Rightarrow$
$-1/4y_2 - 1/4y_4 + y_5 = -3/4$

Adăugăm această restricție la tabelul simplex optim și aplicăm simplexul dual

x_1	3	1	
x_2	-1	2	
y_1	0	4	
y_3	0	1	
y_2	0	3	
	f	1	

Într-o singură iterație s-a obținut optimul întreg, punctul $M = [x_1 = 1, x_2 = 2]$, cu $f = 1$.

1.4 Algoritm de tip plan de secțiune pentru rezolvarea problemelor de programare liniară în numere întregi (GOMORY, 1958)

Considerăm problema:

$$(ILP)\begin{cases} A \cdot x = b \\ x \geq 0\ ,\ x \text{ ÎNTREG} \\ (\max) f = c \cdot x \end{cases} \qquad (LP)\begin{cases} A \cdot x = b \\ x \geq 0 \\ (\max) f = c \cdot x \end{cases}$$

coeficienţii lui A şi ai lui *b* sunt întregi.

Fie $A_{ILP} \subset A_{LP}$ mulţimile de soluţii admisibile ale celor două probleme. Vom presupune că (LP) are optim finit x^*. Dacă x^* are toate componentele întregi atunci x^* este şi soluţie optimă a lui (ILP). În caz contrar, x^0 - dacă există – NU se numără printre vârfurile (punctele extreme) ale poliedrului A_{LP} şi deci nu va putea fi detectat de algoritmul Simplex!

Ideea algoritmului este:

Presupunem că x^* nu are toate componentele întregi. Se construieşte o restricţie neverificată de optimul fracţionar x^* dar verificată de orice soluţie admisibilă întreagă. Se adaugă această restricţie problemei originale renotată LP_0 şi se reoptimiozează.

Fie x^{**} soluţia optimă a problemei augmentată notată LP_1. Datorită modului în care a fost definită restricţia suplimentară avem

$$A_{ILP} \subset A_{LP1} \subset A_{LP0} = A_{LP}$$

Dacă nici x^{**} nu are toate componentele întregi se repetă procedeul: se construieşte o nouă restricţie neverificată de x^{**} dar verificată de soluţiile admisibile întregi. Se adaugă restricţia la LP_1 obţinând o problemă de programare liniară, LP_2. Prin construcţie:

$$A_{ILP} \subset A_{LP2} \subset A_{LP1} \subset A_{LP0} = A_{LP}$$

Prin reoptimizare se decide dacă LP_2 are sau nu soluţie optimă. Teoria ne asigură că *în anumite condiţii*, *într-un număr* ***finit*** *de paşi* se ajunge la o problemă de programare liniară, LP_{k-1} a cărei soluţie optimă $x^{k(*)}$ are toate componentele întregi şi deci reprezintă soluţia optimă a lui (ILP).

Din punct de vedere geometric fiecare nouă restricţie îndepărtează o porţiune din mulţimea soluţiilor admisibile ale problemei precedente de unde denumirea de TĂIETURI acordată acestor restricţii suplimentare.

Să vedem acum ***modul în care se generează aceste tăieturi***:

Se aplică problemei de programare liniară (LP) = (LP_0) algoritmul Simplex. Fie B baza optimală în raport cu care avem:

$I \subset \{1, 2, \ldots, n\}$= mulţimea indicilor variabilelor bazice;
$J = \{1, 2, \ldots, n\} - I$ = mulţimea indicilor variabilelor nebazice.

Prin înmulţirea sistemului $A \cdot x = b$ la stânga cu B^{-1} explicităm variabilele bazice (x_i), $i \in I$ în funcţie de cele secundare (x_j), $j \in J$

$$x_i + \sum_{j \in J} \bar{a}_{ij} x_j = \bar{b}_i \qquad (15)$$

Termenii liberi ($\bar{b}_i$), $i \in I$ formează coloana VVB a tabelului Simplex optim iar coeficienţii ($\bar{a}_{ij}$), $i \in I$ formează coloana A_j a aceluiaşi tabel.

Valorile $x_i^* = b_i$, $i \in I$, reunite cu $x_j^* = 0$, $j \in J$ formează soluţia optimă x^* a problemei $LP=LP_0$.

Să presupunem că cel puțin una din mărimile $\bar{b}_i$ este fracționară (altminteri x* = x^0 = soluția optimă a programului (ILP)). Fie aceasta $\bar{b}_r$.

Se ştie că orice număr real *a* poate fi pus în forma *a* = [*a*] + {*a*}, unde:

[*a*] = partea întreagă a lui a = cel mai mare număr întreg $\leq$ *a*;

{*a*}= partea fracționară a lui *a* adică *a* - [*a*] $\Rightarrow$ **0 ≤ {*a*} < 1**

Exemplu:

- a = 2,781 $\Rightarrow$ [a] = 2 , {a}= 0,781
- a = -2,3 $\Rightarrow$ [a]= -3, {a}= 0,7

În general, $a \in Z$ dacă {a}= 0.

Aplicăm coeficienților restricției de rang *r*

$$\bar{b}_r = x_r + \sum_{j \in J} \bar{a}_{rj} x_j \qquad (16)$$

descompunerea de mai sus:

$$[\bar{b}_r] + \{\bar{b}_r\} = x_r + \sum_{j \in J} ([\bar{a}_{rj}] + \{\bar{a}_{rj}\}) \cdot x_j \qquad (17)$$

Ipoteza făcută asupra lui b_r implică $0 < \{b_r\} < 1$ (18)

$$[\bar{b}_r] - \sum_{j \in J} [\bar{a}_{rj}] \cdot x_j - x_r = \sum_{j \in J} \{\bar{a}_{rj}\} \cdot x_j - \{\bar{b}_r\} \qquad (19)$$

Fie $\hat{x}$ o soluție admisibilă întreagă a problemei ILP. Atunci membrul stâng al relației (19) calculat în $\hat{x}$ este un număr întreg

$$[\bar{b}_r] - \sum_{j \in J} [\bar{a}_{rj}] \hat{x}_j - \hat{x}_r \in Z \qquad (20)$$

În consecință, şi membrul drept al acestei egalități calculat în aceeaşi soluție este un număr întreg:

$$\sum_{j \in J} \{\bar{a}_{rj}\} \hat{x}_j - \{\bar{b}_r\} \in Z \qquad (21)$$

Practic demonstrăm că

$$\sum_{j \in J} \{\bar{a}_{rj}\} \hat{x}_j - \{\bar{b}_r\} \geq 0 \qquad (22)$$

Demonstrație:

Presupunem prin absurd contrariul:

$$\sum_{j\in J} \{\bar{a}_{rj}\}\hat{x}_j - \{\bar{b}_r\} < 0 \qquad (23)$$

atunci din (19) rezultă:

$$[\bar{b}_r] - \sum_{j\in J} [\bar{a}_{rj}]\,\hat{x}_j - \hat{x}_r < 0$$

şi din (20) deducem:

$$[\bar{b}_r] - \sum_{j\in J} [\bar{a}_{rj}]\,\hat{x}_j - \hat{x}_r \le -1$$

Folosind din nou (19) deducem: $\sum_{j\in J} \{\bar{a}_{rj}\}\hat{x}_j - \{\bar{b}_r\} \le -1$, adică $\sum_{j\in J} \{\bar{a}_{rj}\}\hat{x}_j \le \{\bar{b}_r\} - 1$

Deoarece $\{\bar{a}_{rj}\} \ge 0$, oricare ar fi $j \in J$ membrul stâng este ≥ 0 în timp ce membrul drept este < 0 deoarece $\{\bar{b}_i\} < 1$.

Contradicția obținută demonstrează inegalitatea (22).

Deoarece *x* a fost ales arbitrar urmează că restricția

$$\sum_{j\in J} \{\bar{a}_{rj}\}x_j \ge \{\bar{b}_r\} \qquad (24)$$

este verificată de orice ***soluție admisibilă întreagă***.

P.d.a.p. în soluția optimă neîntreagă x^* avem $x^*_j = 0$, $j\in J$ valori care introduse în (22) duc la $\{\bar{b}_r\} \le 0$ în contradicție cu ipoteza (18) potrivit căreia $\{\bar{b}_r\} > 0$!

În consecință, inegalitatea (24) nu este verificată de optimul fracționar x^*.

Vom adăuga restricția (24) la problema LP obținând problema augmentată – cu m+1 restricții – notată LP_1. Pentru reoptimizare transformăm (24) în egalitate introducând o variabilă de abatere x_{n+1}:

$$-\Sigma\{\bar{a}_{rj}\}x_j + x_{n+1} = -\{\bar{b}_r\}$$

adăugăm această restricție la tabelul Simplex curent:

B	CB	VVB	A_r	$A_j(j\in J)$	A_{n+1}
x_r	C_r	$\bar{b}_r$	1	$\bar{a}_{rj}$	
…	…	……	……		……
x_{n+1}	0	$-\{\bar{b}_r\}$	…… 0 ……	$-\{\bar{a}_{rj}\}$ ……	
	f	f	… * ……	Δ_j ……	*

$$\begin{cases} x_i = \bar{b}_i,\ i \in I \\ x_j = 0,\ j \in J \\ x_{n+1} = \{-\bar{b}_r\} \end{cases}$$ constituie o **soluție de bază Dual admisibilă** pentru LP_1.

Utilizând Simplexul dual, după una sau mai multe iterații se ajunge la soluția optimă *x*** a acestei probleme după care se reia procedeul descris anterior.

Restricția de rang *r* (16) din care am derivat tăietura (24) se numeşte RESTRICȚIE GENERATOARE. Ea se caracterizează prin faptul că termenul ei liber este fracționar. În caz că mai multe restricții din (15) au termen liber fracționar, restricția generatoare va fi aleasă astfel încât partea fracționară $\{\bar{b}_r\}$ să fie cât mai mare. Ceea ce este curios este că cu această regulă care s-a dovedit foarte utilă în practică, nu s-a putut dovedi convergența algoritmului. Pentru convergență este necesară aducerea problemei LP la o anume formă şi aplicarea altei reguli privind restricția generatoare. Aceste transformări destul de tehnice dar care nu micşorează generalitatea considerațiilor pot fi ignorate în exemplele ilustrative care sunt de regulă mici ca dimensiune.

Comentariu

Ideea algoritmului şi demonstrația convergenței sunt datorate lui Ralph E. Gomory (1958 – 1960). El a propus doi algoritmi pentru rezolvarea problemelor cu toate variabilele întregi, unul denumit DISCRET celălalt CICLIC. Anterior am expus algoritmul CICLIC. Tot Gomory a propus şi un algoritm pentru rezolvarea problemelor MIXTE în care numai o parte din variabile sunt supuse restricțiilor de integritate.

Experiența numerică acumulată până în prezent degajă un sentiment de **incoerență** în ceea ce priveşte eficacitatea algoritmilor pe o problemă sau alta. Într-o excelentă monografie apărută în 1962 se semnala faptul că o problemă cu 90 de variabile a fost rezolvată foarte rapid în timp ce alta cu numai 12 variabile nu a putut fi rezolvată nici după câteva sute de iterații.

S-a conturat din ce în ce mai mult ideea că – spre deosebire de problemele de programare liniară în care metoda simplex s-a dovedit universală – problemele interne au fiecare structura lor internă pentru care trebuie creat un algoritm particular sau cel puțin specializat unul general.

De asemenea, au fost dezvoltate puternic procedeele care, fără a conduce la soluția optimă, conduc la soluții suboptimale acceptabile.

Exemplu numeric:

$$(ILP) \quad \begin{cases} 2x_1 + x_2 - x_3 \le 4 \\ 4x_1 - 3x_2 \le 2 \\ -3x_1 + 2x_2 + x_3 \le 3 \\ x_{1,2,3} \ge 0, \text{ ÎNTREGI} \\ (\max) f = x_1 - 3x_2 + 3x_3 \end{cases}$$

Rezolvare:

Aducem (ILP) la forma bună în vederea aplicării simplex-ului:

$$(FSILP) = (FBILP): \begin{cases} 2x_1 + x_2 - x_3 + x_4 = 4 \\ 4x_1 - 3x_2 + x_5 = 2 \\ -3x_1 + 2x_2 + x_3 + x_6 = 3 \\ x_{1\ldots6} \geq 0, \text{ ÎNTREGI} \\ (\max) f = x_1 - 3x_2 + 3x_3 \end{cases}$$

B	CB	VVB	x_1	x_2	x_3	x_4	x_5	x_6	
x_4	0	4	2	1	-1	1	0	0	
x_5	0	2	4	-3	0	0	1	0	
x_6	0	3	-3	2	**1**	0	0	1	
	f	0	-1	3	-3	*	*	*	
x_4	0	7	-1	3	0	1	0	1	
x_5	0	2	**4**	-3	0	0	1	0	
x_3	3	3	-3	2	1	0	0	1	
	f	9	-10	9	*	*	*	3	x_7
x_4	0	15/2	0	9/4	0	1	1/4	1	
x_1	1	1/2	1	-3/4	0	0	1/4	0	
x_3	3	9/2	0	-1/4	1	0	3/4	1	
	f	14	*	3/2	*	*	5/2	3	
x_7	0	-1/2	0	**-1/4**	0	0	-1/4	0	1

(min) [3/2 / 1/4 = 6, 5/2 / 1/4 = 10] = 6

Restricția generatoare:
$9/4\ x_2 + x_4 + 1/4x_5 + x_6 = 15/2$
$(2+1/4)x_2 + x_4 + 1/4x_5 = 7+1/2$
$1/4x_2 + 1/4x_5 - 1/2 = 7 - 2x_2 - x_4 - x_6$
$\Rightarrow$ tăietura $1/4x_2 + 1/4x_5 - 1/2 \geq 0$ (aducem la forma STAS utilizând x_7)

Notăm $x_7 = 1/4x_2 + 1/4x_5 - 1/2 \Rightarrow -1/4x_2 - 1/4x_5 + x_7 = -1/2$.
Reoptimizând se obține:

B	CB	VVB	
x_4	0	3	
x_1	1	2	
x_3	3	5	
x_2	-3	2	
	f	11	

Soluția curentă are toate componentele întregi: $x^0_1 = 2$, $x^0_2 = 2$, $x^0_3 = 5$, $f_{max} = 11$.