

Sistemul de restricții (1) realizează legătura dintre resurse și activități prin intermediul celor m restricții liniare.

Modelul problemei de programare liniară conține restricții de tipul (1) precum și un criteriu de "performanță" care să permită evaluarea eficienței fiecărei activități. În funcție de scopul urmărit, putem alege drept criteriu de eficiență un indicator care măsoară efortul, unul care măsoară rezultatul sau un indicator exprimat ca raport între rezultat și efort (sau efort pe rezultat).

Este evident că eficiența maximă înseamnă minimizarea efortului și maximizarea rezultatului, iar conceptul de optim se definește, în acest caz, ca un program $x \in R^n$ care minimizează sau maximizează o funcție obiectiv și, în același timp, satisface toate restricțiile tehnico-economice.

Presupunând că fiecare componentă a vectorului linie $c = (c_1, c_2, \dots, c_n)$ măsoară eficiența unei unități din rezultatul activității A_j , atunci se poate introduce funcția liniară:

$$f(x) = c_1 \cdot x_1 + c_2 \cdot x_2 + \dots + c_n \cdot x_n$$

care evaluează performanța oricărui program x .

Sintetizând, obținem următorul program de programare liniară:

$$\text{optim}_x [f(x)] \quad (1)$$

$$\left. \begin{array}{l} \sum_{j=1}^n a_{ij} \cdot x_j \leq b_i \quad i \in I_1 \\ \sum_{j=1}^n a_{kj} \cdot x_j \geq b_k \quad k \in I_2 \end{array} \right\} \text{unde } I_1 \cup I_2 = \{1, 2, \dots, m\} \quad (2)$$

$$x_j \geq 0 \quad j = 1, n \quad (3)$$

Relațiile (1), (2) și (3) constituie împreună modelul general al unei probleme de programare liniară, având fiecare un rol specific:

1. relația (1), unde $f(x) = \sum_{j=1}^n c_j \cdot x_j$ este denumită funcția obiectiv de eficiență a problemei, evaluează eficiența/permanța fiecărei variante de program x ;
2. relațiile (2) de tipul $\sum_{j=1}^n a_{ij} \cdot x_j \leq b_i$ reprezintă restricții de tip resurse; iar restricțiile de tipul $\sum_{j=1}^n a_{kj} \cdot x_j \geq b_k$ se referă la restricții tehnico-economice de tip calitativ (și ca urmare indicatorul b_k este limita inferioară impusă "rețetei optime");
3. relația (3) $x_j \geq 0 \quad j = 1, \dots, n$, numită condiția de nenegativitate a variabilelor, asigură obținerea unei soluții realizabile din punctul de vedere al logicii economice.

După cum s-a arătat la început, structura concretă a unei aplicații în economie este determinată în primul rând de obiectivul urmărit.

Astfel, în cazul *problemei determinării structurii sortimentale optime a producției*, se cunosc cantitățile disponibile (cantitățile de care se poate face rost pe perioada analizată) din fiecare materie primă $\{b_i, i = 1, \dots, m\}$, coeficienții tehnologici $\{a_{ij}, i = 1, \dots, m, j = 1, \dots, n\}$ (a_{ij} reprezintă cantitatea din materia primă i necesară fabricării unei unități din produsul de tipul j), cantitățile maxime $\{\bar{x}_j, j = 1, \dots, n\}$ și minime $\{\underline{x}_j, j = 1, \dots, n\}$ ce pot fi produse din fiecare sortiment în perioada analizată și profiturile unitare $\{p_j, j = 1, \dots, n\}$ ale fiecărui tip de produs. Se cere găsirea acelor cantități x_j care trebuie fabricate din fiecare tip de produs astfel încât să se obțină profitul maxim, în condițiile nedepășirii disponibilurilor din fiecare resursă.

Pentru simplificarea modelului, se presupune că prețul unui bun nu depinde de cantitatea produsă din acesta sau din celelalte, consumul din fiecare materie primă este direct proporțional cu cantitatea produsă și, pentru fiecare bun, consumurile dintr-o resursă sau alta nu se condiționează reciproc.

Problema matematică echivalentă este:

$$\begin{cases} \max_{x_j, j=1, n} (p_1 x_1 + p_2 x_2 + \dots + p_n x_n) \\ a_{i1} x_1 + a_{i2} x_2 + \dots + a_{in} x_n \leq b_i & i = 1, \dots, m \\ \underline{x}_j \leq x_j \leq \bar{x}_j & j = 1, \dots, n \\ x_j \geq 0 & j = 1, \dots, n \end{cases}$$

În unele probleme, în loc de profiturile p_j se cunosc veniturile unitare v_j sau costurile unitare c_j sau alt criteriu de eficiență, scopul fiind maximizarea venitului, minimizarea costurilor respectiv optimul indicatorului de eficiență respectiv, sau toate la un loc. De asemenea pot lipsi condițiile de limitare a producției sau pot exista și alte condiții.

La o **problemă de programare operativă a producției** restricțiile se referă la o serie de mașini (utilaje) cu care se execută produsele dorite, b_i fiind disponibilul de timp al utilajului i pe perioada analizată iar a_{ij} timpul necesar prelucrării unui produs de tipul j pe utilajul i , scopul fiind maximizarea producției.

Ca urmare, modelul are forma:

$$\begin{cases} \max_{x_j (j=1, \dots, n)} (x_1 + x_2 + \dots + x_n) \\ a_{i1} x_1 + a_{i2} x_2 + \dots + a_{in} x_n \leq b_i & i = 1, \dots, m \\ x_j \geq 0 & i = 1, \dots, n \end{cases}$$

Dacă se dorește obținerea unui meniu (rețete furajere), care să asigure necesarurile $\{b_i, i = 1, \dots, m\}$ dintr-un număr de m substanțe esențiale organismului, având la dispoziție un număr de n alimente, cunoscându-se cantitățile $\{a_{ij}, i = 1, \dots, m, j = 1, \dots, n\}$ din fiecare substanță pe care le conține o unitate de măsură din fiecare aliment și costurile $\{c_j, j = 1, \dots, n\}$ unei unități de măsură din fiecare aliment, putem scrie modelul:

$$\begin{cases} \min_{x_j (j=1, \dots, n)} (c_1 x_1 + c_2 x_2 + \dots + c_n x_n) \\ a_{i1} x_1 + a_{i2} x_2 + \dots + a_{in} x_n \geq b_i & i = 1, \dots, m \\ x_j \geq 0 & j = 1, \dots, n \end{cases}$$

Variabilele x_j reprezintă, în acest caz, cantitatea din fiecare aliment ce va intra în meniu iar $f(x) = \sum_{j=1}^n c_j \cdot x_j$ este costul total al rețetei definită de vectorul x .

Vom încheia exemplificarea cu prezentarea *modelului amestecului optim de produse petroliere*. Presupunem că o rafinărie dispune de n tipuri de benzine, prin amestecarea acestora urmând să obțină o benzină cu m caracteristici impuse și la un preț minim posibil. O serie de caracteristici trebuie să fie îndeplinite cu o limită inferioară (de exemplu cifra octanică), altele cu o limită superioară (de exemplu densitatea sau temperatura de fierbere) și altele cu egalitate (de exemplu cantitatea necesară), aceste limite fiind $b_i^1, i = 1, \dots, m_1$, $b_i^2, i = 1, \dots, m_2$ respectiv $b_i^3, i = 1, \dots, m_3$ cu $m_1 + m_2 + m_3 = m$. De asemenea, trebuie ținut cont de cantitățile disponibile din fiecare benzină $D_i, i = 1, \dots, n$. Se cunosc caracteristicile fiecărei benzine disponibile, date într-o matrice $A \in M_{m \times n}$, unde a_{ij} este valoarea caracteristicii i pentru benzina j și prețurile unitare ale fiecărei benzine, notate $p_j, j = 1, \dots, n$. Dacă notăm cu $x_j, j = 1, \dots, n$, cantitățile din fiecare benzină care vor forma amestecul optim, atunci avem de rezolvat problema:

$$\min_{x_j} \left(\sum_{j=1}^n p_j \cdot x_j \right)$$

$$\begin{cases} \sum_{j=1}^n a_{ij} \cdot x_j \geq b_i^1 & i = 1, \dots, m_1 \\ \sum_{j=1}^n a_{ij} \cdot x_j \leq b_i^2 & i = m_1 + 1, \dots, m_1 + m_2 \\ \sum_{j=1}^n a_{ij} \cdot x_j = b_i^3 & i = m_1 + m_2 + 1, \dots, m \\ x_j \leq D_j & j = 1, \dots, n \\ x_j \geq 0 & j = 1, \dots, n \end{cases}$$

Programarea matematică

Se observă că toate aceste probleme, cu toate că reprezintă situații economice total diferite, sunt aplicații în sfera activității economice ale următoarei probleme de optimizare:

$$\begin{cases} (\max) \text{ sau } (\min) f(x_1, x_2, \dots, x_n) & 1.1 \\ g_i(x_1, x_2, \dots, x_n) \leq b_i \quad i = 1, \dots, n & 1.2 \\ x_1, x_2, \dots, x_n \in \Omega \subseteq R^n & 1.3 \end{cases}$$

în care funcțiile $f, g_i : R^n \rightarrow R$ pot avea orice formă și proprietăți (liniare, convexe, continui, diferențiabile etc) iar Ω poate fi orice submulțime a lui R^n (continuuă sau discretă, mărginită sau nemărginită, convexă sau neconvexă, finită sau infinită etc) și în care dorim să găsim minimul sau maximum funcției f în variabilele x_i care îndeplinesc restricțiile 1.2 și 1.3. O astfel de problemă se numește **problemă de programare matematică**.

Funcția (1.1) se numește funcția obiectiv a problemei de optimizare. În aplicațiile economice, ea reprezintă criteriul de performanță urmărit: maximizarea beneficiului, maximizarea producției marfă, minimizarea costului producției, maximizarea gradului de încărcare al utilajelor sau minimizarea timpului de staționare al acestora, maximizarea veniturilor etc.

Inegalitățile (1.2), în care g_i sunt funcții de n variabile iar b_i sunt constante, se numesc restricții ale problemei de optimizare. Ele traduc în limbaj matematic condițiile de natură economică sau tehnologică în care se desfășoară procesul economic modelat, cum ar fi: nedepășirea stocurilor disponibile de resurse (materii prime, capacități de producție, forță de muncă, fonduri bănești, timp etc), îndeplinirea sau depășirea unor indicatori economici (producția fizică, netă) etc.

Condițiile (1.3), impuse "direct" variabilelor, depind de natura problemei studiate. De cele mai multe ori, Ω este mulțimea vectorilor $x = (x_1, \dots, x_n) \in R^n$ cu toate componentele nenegative și acest fapt se justifică prin aceea că, în general, x_i reprezintă "nivelele" unor activități de producție, nivele care nu pot fi negative. În unele probleme de optimizare, cum ar fi cele de croire sau ordonanțare, variabilele desemnează mărimi indivizibile și deci nu pot lua decât valori întregi. Mulțimea Ω va fi formată în acest caz din vectori x cu toate componentele întregi și nenegative. În alte probleme, ca de pildă cele de afectare, portofoliu etc, variabilele nu pot lua decât valorile 0 sau 1 (variabile bivalente) și ca atare Ω va fi formată din vectorii $x = (x_1, \dots, x_n)$ cu toate componentele 0 sau 1. Caracteristic acestor tipuri de probleme este faptul că Ω devine o submulțime discretă din R^n , de unde și numele generic de probleme de optimizare discretă dat acestora.

O soluție a unei astfel de probleme are deci trei proprietăți:

P1. verifică restricțiile 1.2

P2. verifică restricțiile "directe" 1.3

P3. optimizează funcția obiectiv pe mulțimea vectorilor care îndeplinesc condițiile P1. și P2.

În teoria programării matematice se folosesc, pentru claritatea și simplitatea expunerii, următoarele noțiuni:

soluție a problemei de optimizare	= un vector $x \in R^n$ care verifică restricțiile 1.2
soluție admisibilă a problemei de optimizare	= un vector $x \in R^n$ care verifică restricțiile 1.2 și 1.3 \Leftrightarrow o soluție care verifică restricțiile 1.3
soluție optimă a problemei de optimizare	= un vector $x \in R^n$ care verifică restricțiile 1.2 și 1.3 și optimizează funcția obiectiv pe mulțimea tuturor vectorilor cu această proprietate \Leftrightarrow o soluție care verifică restricțiile 1.3 și optimizează funcția obiectiv pe mulțimea tuturor soluțiilor cu această proprietate \Leftrightarrow o soluție admisibilă care optimizează funcția obiectiv pe mulțimea soluțiilor admisibile

Izvorâtă din studiul problemelor extremale ale analizei matematice clasice, programarea matematică a cunoscut în ultimele decenii o dezvoltare spectaculoasă, explicabilă numai în parte prin progresele înregistrate în matematică. Într-adevăr, această dezvoltare a fost puternic stimulată de creșterea vertiginoasă a complexității activităților economice, care a furnizat probleme cu structuri din ce în ce mai complicate, ca și de rapida perfecționare a mijloacelor automate de calcul, singurele capabile să testeze eficiența practică a metodelor teoretice elaborate. La rândul ei, programarea matematică, prin rezultatele ei, a adus un considerabil aport la perfecționarea metodelor de conducere în economie și a impulsionat cercetările - în plan teoretic - privind modelarea sistemelor economice complexe, studiul și interpretarea legilor și proceselor economice.

În ceea ce privește rezolvarea acestor probleme, este evident că varietatea extremă a acestora face imposibilă găsirea unui algoritm practic care să le poată rezolva pe toate, dar putem găsi (sau ne putem aștepta să găsim), pentru fiecare caz particular, un algoritm de rezolvare care să-și tragă eficacitatea tocmai din folosirea tuturor particularităților cazului respectiv. În prezent există sute de algoritmi de rezolvare, cercetarea problemei evoluând din ce în ce mai mult spre testarea și îmbunătățirea acestora decât spre găsirea unora noi.

Problema de programare liniară

Definiția 1. Dacă într-o problemă de programare matematică funcția obiectiv f și funcțiile g_i , din restricțiile 1.2, sunt funcții liniare atunci problema se numește problemă de programare liniară. O problemă de programare liniară este, deci, un caz particular al problemelor de programare matematică și, ținând cont de forma oricărei funcții liniare, rezultă că forma generală a oricărei probleme de programare liniară este:

$$\begin{cases} \max(\min)f = c_1 \cdot x_1 + c_2 \cdot x_2 + \dots + c_n \cdot x_n \\ a_{i1} \cdot x_1 + a_{i2} \cdot x_2 + \dots + a_{in} \cdot x_n = b_i \quad i = 1, \dots, n \\ x_1, x_2, \dots, x_n \begin{matrix} \leq 0 \\ \geq 0 \\ \text{oarecare} \end{matrix} \end{cases}$$

unde c_j (coeficienții funcției obiectiv), a_{ij} (coeficienții restricțiilor) și b_i (termenii liberi) sunt constate reale.

Forma conică și forma standard a unei probleme de programare liniară

Cu toate că problema de programare liniară este cea mai simplă dintre problemele de programare matematică, ea este încă destul de complicată, prin faptul că orice restricție poate avea trei forme diferite iar obiectivul poate fi minimizarea sau maximizarea funcției f . Este puțin probabil că există un algoritm și simplu și aplicabil la toate cazurile.

Din acest motiv este mult mai simplu să găsim o anumită formă (cât mai simplă) cu proprietatea că pentru orice problemă P , există o alta problemă P' de această formă, echivalentă cu problema inițială P (spunem că două probleme sunt echivalente dacă există un izomorfism între mulțimile soluțiilor celor două probleme și un homeomorfism între funcțiile lor obiectiv) și să dispunem de un algoritm care să rezolve problemele de această formă și de o regulă prin care să găsim soluția problemei inițiale P din soluția problemei P' , găsită cu acest algoritm.

În acest sens (dar și din cauza frecvenței apariției lor în practică) s-au evidențiat două forme ale problemelor de programare liniară, **forma canonică** și **forma standard**.

Definiția 2. Spunem că o problemă este la **forma canonică** dacă are una din următoarele două forme:

Forma canonică de minim	Forma canonică de maxim
$\begin{cases} (\min)f = c_1 \cdot x_1 + c_2 \cdot x_2 + \dots + c_n \cdot x_n \\ a_{i1} \cdot x_1 + a_{i2} \cdot x_2 + \dots + a_{in} \cdot x_n \geq b_i \quad i = 1, \dots, n \\ x_1, x_2, \dots, x_n \geq 0 \end{cases}$	$\begin{cases} (\max)f = c_1 \cdot x_1 + c_2 \cdot x_2 + \dots + c_n \cdot x_n \\ a_{i1} \cdot x_1 + a_{i2} \cdot x_2 + \dots + a_{in} \cdot x_n \leq b_i \quad i = 1, \dots, n \\ x_1, x_2, \dots, x_n \geq 0 \end{cases}$

Această formă este cel mai des întâlnită în practică (vezi problema determinării structurii sortimentale optime a producției sau problema dietei) și, în plus, restricțiile economice sunt în general inegalități și foarte rar egalități. De asemenea, această formă este invariantă la anumite transformări (vezi problema duală) și asigură existența soluției (orice problemă la această formă, care are termenii liberi și coeficienții restricțiilor pozitivi, are soluție).

Definiția 3. Spunem că o problemă este la **forma standard** dacă are forma:

$$\begin{cases} (\min) \text{ sau } (\max) f = c_1 \cdot x_1 + c_2 \cdot x_2 + \dots + c_n \cdot x_n \\ a_{i1} \cdot x_1 + a_{i2} \cdot x_2 + \dots + a_{in} \cdot x_n = b_i \quad i=1, \dots, n \\ x_1, x_2, \dots, x_n \geq 0 \end{cases}$$

Această formă, deși nenaturală din punct de vedere economic, este cea care se pretează cel mai bine la rezolvarea cu metodele algebrei clasice.

Propoziție. Pentru orice problemă de programare liniară P există o problemă la forma canonică PFC și o problemă la forma standard PFS echivalente cu P.

Demonstrație. Într-adevăr:

a) orice problemă de maxim poate fi transformată în una de minim și reciproc folosind relația:

$$\max f(x) = - \min f(-x)$$

b) orice restricție de tipul " \leq " poate fi transformată într-o restricție de forma " \geq " și reciproc folosind relația:

$$\alpha \leq \beta \Leftrightarrow -\alpha \geq -\beta$$

c) orice restricție inegalitate poate fi transformată în egalitate, prin introducerea unei variabile suplimentare nenegative și folosind relațiile:

$$\alpha \leq \beta \Leftrightarrow \begin{cases} \alpha + x = \beta \\ x \geq 0 \end{cases} \quad \text{și} \quad \alpha \geq \beta \Leftrightarrow \begin{cases} \alpha - x = \beta \\ x \geq 0 \end{cases}$$

Toate variabilele introduse pentru transformarea inegalităților în egalități se numesc **variabile de abatere** sau **variabile de compensare**.

d) orice restricție egalitate poate fi transformată în restricții inegalitate, folosind relația:

$$\alpha = \beta \Leftrightarrow \begin{cases} \alpha \leq \beta \\ \alpha \geq \beta \end{cases}$$

e) orice variabilă cu restricție de semn negativă ($x \leq 0$) poate fi înlocuită cu o variabilă cu restricție de semn pozitivă ($y \geq 0$), folosind relația:

$$x \leq 0 \Leftrightarrow \begin{cases} x = -y \\ y \geq 0 \end{cases}$$

f) orice variabilă fără restricție de semn poate fi înlocuită cu două variabile cu restricție de semn pozitivă, folosind relația:

$$x \text{ oarecare} \Leftrightarrow \begin{cases} x = y - z \\ y \geq 0 \\ z \geq 0 \end{cases}$$

Se demonstrează fără un efort matematic deosebit că dacă P' se obține din P folosind doar

transformările de mai sus (numite și transformări elementare) atunci P și P' sunt două probleme echivalente și între soluțiile lor optime există o bijecție.

Din cele de mai sus rezultă **cum** poate fi adusă orice problemă de programare liniară la forma standard (sau canonică) și **cum** se obține soluția problemei inițiale din soluția problemei la forma standard (sau canonică).

Exemplu: Problemei P de mai jos îi corespunde problema la forma standard PFS alăturată:

P	PFS
$(\min) f = 3x_1 - 2x_2 + 4x_3$ $\begin{cases} 9x_1 + 13x_2 + 15x_3 \leq 16 \\ 19x_1 + 36x_2 + 6x_3 \geq -9 \\ 7x_1 - 7x_2 - 14x_3 \leq 21 \\ 3x_1 + 30x_2 + 5x_3 \leq 31 \\ x_1 \leq 0, x_2 \text{ oarecare}, x_3 \geq 0 \end{cases}$	$(\max) g = +3a_1 + 2y_2 - 2z_2 - 4x_3$ $\begin{cases} -9a_1 + 13y_2 - 13z_2 + 15x_3 + x_4 = 16 \\ -19a_1 + 36y_2 - 36z_2 + 6x_3 - x_5 = -9 \\ -7a_1 - 7y_2 + 7z_2 - 14x_3 + x_6 = 21 \\ -3a_1 + 30y_2 - 30z_2 + 5x_3 + x_7 = 31 \\ a_1, y_2, z_2, x_3, x_4, x_5, x_6, x_7 \geq 0 \end{cases}$

Problema PFS are o singură soluție optimă:

$$a_1 = 3, y_2 = \frac{4}{3}, z_2 = 0, x_3 = 0, x_4 = \frac{77}{3}, x_5 = 0, x_6 = \frac{154}{3}, x_7 = 0$$

care dă un maxim al funcției g egal cu $\frac{35}{3}$.

Acestei soluții îi corespunde singura soluție optimă pentru P:

$$x_1 = -3, x_2 = \frac{4}{3}, x_3 = 0$$

care dă un minim al funcției f, egal cu $-\frac{35}{3}$.

Rezolvarea problemei de programare liniară.

Analiza problemei

Fie o problemă P despre care presupunem (fără a restrânge generalitatea) că a fost adusă la forma standard. De asemenea presupunem (tot fără a restrânge generalitatea) că variabilele problemei au fost numerotate și denumite x_j cu $j = 1, \dots, n$ ($n =$ numărul de necunoscute), coeficienții variabilelor din funcția obiectiv f cu c_j ($c_j =$ coeficientul variabilei x_j), că în ecuații variabilele apar în ordinea indicilor, ecuațiile fiind numerotate de la 1 la m ($m =$ numărul de ecuații) și că am notat cu b_i termenul liber al ecuației i și cu a_{ij} coeficientul variabilei j din ecuația i . În acest caz putem așeza variabilele problemei într-un vector cu n componente, coeficienții funcției obiectiv într-un vector cu n componente, termenii liberi într-un vector cu m componente, coeficienții variabilelor din ecuații într-o matrice cu m linii și n coloane și vom avea:

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

(vom nota cu x^T, c^T și b^T transpușii acestor vectori și cu A^T transpusa matricei A).

Alegerea așezării ca vectori coloană a fost făcută din rațiuni de ușurință a calculelor și a

memorării acestora. După aceste notații, problema poate fi scrisă mult mai simplu:

$$\begin{aligned} (\min) f = c^T \cdot x & \quad (\max) f = c^T \cdot x \\ \begin{cases} A \cdot x = b \\ x \geq 0 \end{cases} & \quad \text{sau} \quad \begin{cases} A \cdot x = b \\ x \geq 0 \end{cases} \end{aligned}$$

Se știe că un sistem de forma $A \cdot x = b$ are soluție doar dacă $\text{rang}(A) = \text{rang}(\bar{A})$, unde \bar{A} este matricea extinsă obținută adăugând matricei A vectorul b , în acest caz sistemul devenind echivalent cu sistemul obținut prin eliminarea restricțiilor care nu corespund minorului principal (dacă sistemul nu are soluție atunci evident nici problema nu are soluții, caz care este total neinteresant).

Presupunem că au fost eliminate aceste restricții. Dacă $\text{rang } A = n$ atunci sistemul are o singură soluție care, dacă este admisibilă, este și soluția optimă căutată, altfel problema nu are soluție. Este evident că și acest caz este la fel de neinteresant ca primul.

Presupunem deci în continuare că:

$$\text{rang}(A) = m < n$$

Rezolvarea sistemului $A \cdot x = b$ se poate face într-un mod simplu (cel puțin teoretic) folosind algebra matricială astfel:

- împărțim coloanele matricei A în două submatrici: minorul principal (notat cu B , care este o matrice pătratică de dimensiune m și va fi numit **bază a sistemului**) și restul coloanelor (notat cu S , care este o matrice cu m linii și $n - m$ coloane);
- împărțim variabilele problemei în doi vectori: vectorul variabilelor principale (corespunzătoare coloanelor bazei) și vectorul variabilelor secundare (notat cu x_S). Făcând eventual o renumerotare, pentru ușurința expunerii și fiind evident că nu se restrânge generalitatea problemei, presupunem că variabilele principale sunt chiar primele m (această presupunere va fi făcută de câte ori va fi posibil, fără a mai specifica acest lucru).
- aducem succesiv sistemul la forma de mai jos:

$$A \cdot x = b \Leftrightarrow (BS) \cdot \begin{pmatrix} x_B \\ x_S \end{pmatrix} = b \Leftrightarrow B \cdot x_B + S \cdot x_S = b \Leftrightarrow B \cdot x_B = b - S \cdot x_S \Leftrightarrow x_B = B^{-1} \cdot b - B^{-1} \cdot S \cdot x_S$$

- soluția sistemului va fi submulțimea lui R^n :

$$D_B = \{x = \{x_B, x_S\} / x_S \in R^{n-m} \text{ oarecare, } x_B = B^{-1} \cdot b - B^{-1} \cdot S \cdot x_S\}$$

Orice alegere a lui x_S dă o soluție. Dintre toate alegerile posibile este remarcabilă (prin simplitatea ei) soluția $x_S = 0$, care duce la soluția particulară:

$$x^B = \left(\begin{array}{c} B^{-1}b \\ 0 \\ 0 \\ \vdots \\ 0 \end{array} \right) \left. \vphantom{\begin{array}{c} B^{-1}b \\ 0 \\ 0 \\ \vdots \\ 0 \end{array}} \right\} n - m \text{ zerouri}$$

numită **soluția de bază asociată bazei B** . Deci x^B este $x_B = B^{-1} \cdot b$ la care se adaugă $n-m$ zerouri. Cu toate acestea, vor fi ambele numite soluție de bază, rezultând din context de care este vorba.

Observație: Este evident că fiecărui minor principal al sistemului (= minor de dimensiune $m =$ bază) îi corespunde o unică soluție de bază. O soluție de bază care are toate componentele nenule strict pozitive se va numi **soluție de bază admisibilă** iar o soluție

optimă care este de bază se va numi **soluție optimă de bază**. Se observă că o soluție de bază are cel mult m componente diferite de 0. Din cauza importanței lor în rezolvarea problemei, vom evidenția soluțiile de bază care au mai puțin decât m componente nenule, numite **soluții degenerare** și pe cele care au fix m elemente nenule, numite **soluții nedegenerate**.

D_B este izomorfă cu R^n , adică are tot atâtea elemente câte puncte sunt într-un spațiu cu $n - m$ dimensiuni. "Alegându-le" din acestea doar pe cele cu toate elementele pozitive, găsim mulțimea în care vom "căuta" vectorul (vectorii) care dă (dau) extremul lui f .

Rezolvarea problemei poate duce la următoarele rezultate:

- R1. Sistemul $A \cdot x = b$ nu are soluții sau nu are soluții admisibile. În acest caz problema nu are soluție.
- R2. Imaginea funcției obiectiv pe mulțimea soluțiilor admisibile este nemărginită (la $+\infty$ într-o problemă de maxim sau la $-\infty$ într-o problemă de minim). În acest caz spunem că problema are optim infinit.
- R3. Imaginea funcției obiectiv pe mulțimea soluțiilor admisibile este mărginită. În acest caz problema are cel puțin o soluție și spunem că problema are optim finit.

Greutatea găsirii soluției problemei constă în imensitatea numărului de soluțiilor posibile ale sistemului și în respectarea condiției de nenegativitate a celor printre care căutăm extremul dorit al funcției f . Este natural ca primele încercări în rezolvare să caute în primul rând o limitare cât mai puternică a locului în care căutăm. Pe baza unor reprezentări geometrice ale problemei au fost descoperite următoarele proprietăți ale problemei, care poartă denumirea de **teoreme fundamentale** ale programării liniare:

Teorema 1. Dacă problema de programare liniară are soluții admisibile atunci are și **soluții de bază admisibile**.

Teorema 2. Dacă problema de programare liniară are soluții optime atunci are și **soluții optime de bază**.

Teorema 3. Mulțimea soluțiilor admisibile (optime) este închisă și convexă. Dacă este și mărginită atunci punctele extreme ale acesteia sunt chiar soluțiile admisibile (optime) de bază ale problemei.

Cele două teoreme realizează efectiv trecerea către o problemă rezolvabilă pe calculator. Într-adevăr, deoarece o bază este un minor de ordinul m al matricii A și unei baze îi corespunde o unică soluție de bază rezultă că sunt cel mult C_n^m soluții de bază, adică un număr finit. În acest moment s-ar părea că nu avem decât să lăsăm calculatorul să calculeze toate soluțiile de bază și valorile funcției obiectiv în cele admisibile găsiind-o prin comparare pe cea care dă minimul sau maximul funcției printre acestea. Totuși, această variantă se dovedește nepractică la o analiză mai atentă, ținând cont de următoarele observații:

1. faptul că numărul soluțiilor de bază este finit ne asigură doar că problema se va termina cândva, ceea ce, din punct de vedere economic, este evident nemulțumitor. Noi dorim ca problema să fie rezolvată în timp util, adică repede. Rezolvând problema ca mai sus vom avea, pentru o problemă cu 50 variabile și 20 restricții, de calculat, listat și comparat C_{50}^{20} soluții de bază, adică în jur de 10^{20} . Presupunând că suntem dotați cu un supercalculator care ar termina un miliard de baze pe secundă, rezolvarea ar dura 3000 ani. De altfel, o problemă ca cea de sus este foarte mică în comparație cu problemele "serioase" ce au peste 1000 de variabile și 100 de restricții. În plus, un calculator ca cel de sus nu există încă, deci în nici un caz nu e disponibil întreprinderilor obișnuite.
2. Cu algoritmul de mai sus vom găsi cea mai bună soluție dintre soluțiile admisibile de bază, fără însă să știm dacă problema admite, de fapt, optim (ar putea să aibă optim

- infinit).
3. Nu vom ști dacă un minor de $m \times m$ este bază decât după ce-i vom calcula determinantul și nu vom ști dacă soluția de bază corespunzătoare este admisibilă decât după ce o vom calcula.
 4. Soluția optimă, odată găsită, nu va fi recunoscută ca atare decât după ce vom calcula toate celelalte soluții de bază, chiar dacă ea a apărut chiar la începutul calculelor.

În concluzie, pentru a fi eficient, un algoritm ar trebui să aibă următoarele proprietăți:

- a) să dispună de un criteriu de recunoaștere a faptului că problema nu are soluții admisibile;
- b) să dispună de un criteriu de recunoaștere a faptului că problema are optim infinit;
- c) să dispună de un criteriu de recunoaștere dacă o soluție este optimă sau nu;
- d) să treacă de la o soluție de bază admisibilă la una cel puțin la fel de bună (o soluție x este mai bună decât o soluție y dacă $f(x) > f(y)$ într-o problemă de maxim și $f(x) < f(y)$ într-o problemă de minim);
- e) să treacă de la o soluție la cea mai bună dintre soluțiile cel puțin la fel de bune posibile ca succesoare;
- f) să nu revină la o soluție deja analizată;
- g) să efectueze un număr de iterații comparabil polinomial cu dimensiunea problemei.
- h) să nu introducă erori de rotunjire (sau nu prea mari);
- i) să fie cât mai simplu de implementat;

Condițiile de mai sus reprezintă de fapt un algoritm ideal. La începuturile cercetării operaționale era suficient, de fapt, și un algoritm care să îndeplinească măcar condițiile b), c), d), e) și h). Acest algoritm a fost dat de G.B. Dantzig, în 1947, care la numit **algoritmul simplex**. El rămâne și în zilele noastre cel mai eficient algoritm în ceea ce privește viteza de lucru, simplitatea și implementarea pe calculator, pentru problemele care apar efectiv în practica economică.

Totuși, el nu îndeplinea condițiile a), f) și g).

Condiția a) este relativ ușor de îndeplinit și ea este acoperită prin introducerea unei faze inițiale suplimentare la algoritmul simplex, rezultând **algoritmul simplex în două faze**.

Condiția f) a fost pusă în momentul în care s-a observat că, în condițiile existenței soluțiilor degenerate, se poate ajunge în situația când, de la o soluție dată, nu se poate ajunge decât la una la fel de bună și tot așa, astfel încât, dacă nu se iau măsuri de evitare, se poate ajunge la o soluție care a mai fost, fenomen numit **ciclare**. Astfel de exemple a fost efectiv găsite, unul fiind **exemplul lui Beale** prezentat mai jos:

$$\begin{aligned}
 (\max) f &= \frac{3}{4}x_4 - 20x_5 + \frac{1}{2}x_6 - 6x_7 \\
 \left\{ \begin{array}{l}
 x_1 \quad \quad \quad + \frac{1}{4}x_4 - 8x_5 - x_6 + 9x_7 = 0 \\
 x_2 \quad \quad \quad + \frac{1}{2}x_4 - 12x_5 - \frac{1}{2}x_6 + 3x_7 = 0 \\
 x_3 \quad \quad \quad \quad \quad \quad \quad \quad + x_6 = 1 \\
 x_i \geq 0, \quad i = 1, \dots, 7
 \end{array} \right.
 \end{aligned}$$

Se observă imediat baza admisibilă $B_0 = (a_1, a_2, a_3)$, de la care, aplicând algoritmul sub forma clasică, se vor obține succesiv bazele admisibile $B_1 = (a_4, a_2, a_3)$, $B_2 = (a_4, a_5, a_3)$, $B_3 = (a_6, a_5, a_3)$, $B_4 = (a_6, a_7, a_3)$, $B_5 = (a_6, a_2, a_3)$, $B_6 = (a_4, a_2, a_3) \dots$. Cititorul poate verifica ușor că toate aceste baze au ca soluție de bază soluția $(0, 0, 1, 0, 0, 0, 0)$ și valoarea funcției 0, dar nu îndeplinesc condiția de optim. Pe de altă parte se vede că $B_6 = B_1$ și deci algoritmul va repeta la infinit această succesiune de baze,

neatingând niciodată valoarea maximă $\frac{7}{4}$, corespunzătoare soluției $(\frac{3}{4}, 0, 0, 1, 0, 1, 0)$.

Această situație este îngrijorătoare nu atât din considerente practice imediate (încă nu a fost găsită o problemă din practică la care să apară acest fenomen, toate exemplele existente fiind artificial construite, ca și cel de mai sus) cât din faptul că un algoritm implementat pe calculator s-ar putea să fie pus în fața unei astfel de probleme, situație în care n-am putea rezolva problema nici pe calculator, nici manual, ea fiind prea mare. Situația a fost depășită prin diferite tehnici suplimentare adăugate celei de trecere la o soluție cel puțin la fel de bună (insuficientă, cum s-a văzut), cea mai cunoscută fiind cea care folosește **ordonarea lexicografică** a soluțiilor, care va fi prezentată și în această carte.

În fine, referitor la condiția g), a durat mult timp până s-a demonstrat că algoritmul, sub această formă, **nu este în timp polinomial**, un exemplu fiind clasa de probleme de mai jos, găsită de Klee și Minty în 1972, în care algoritmul trebuie să analizeze 2^n baze (n = numărul de necunoscute) până la găsirea celei optime.

$$(\max)f = \sum_{j=1}^n 10^{n-j} x_j$$

$$\left\{ \begin{array}{l} \left(2 \cdot \sum_{j=1}^{i-1} 10^{i-j} x_j \right) + x_i \leq 100^{i-1} \quad i = 1, \dots, n \\ x_j \geq 0 \quad j = 1, \dots, n \end{array} \right.$$

Pentru o astfel de problemă, la 100 de variabile, algoritmul va avea $2^{100} \cong 10^{30}$ iterații, și chiar la o viteză de un miliard iterații pe secundă (mult peste puterea unui calculator actual) va termina în 10^{13} ani.

Nu se știe încă dacă există sau nu o altă modalitate de trecere de la o bază la alta, folosind tabelele simplex, prin care algoritmul să devină în timp polinomial. Au fost însă găsiți algoritmi care nu folosesc tabelele simplex, primul fiind algoritmul de punct interior al lui Karmakar, despre care s-a demonstrat că lucrează în timp polinomial.

În ceea ce privește erorile de rotunjire, inevitabile când se fac calculele pe un calculator, algoritmul se comportă într-adevăr foarte rău, orice eroare propagându-se imediat la tot tabelul cu efecte foarte mari. Acest lucru poate fi însă depășit aplicând o variantă a algoritmului, numită *varianta revizuită* a algoritmului simplex.

Toate punctele slabe ale algoritmului, amintite mai sus, nu au înmormântat însă algoritmul simplex, deoarece folosirea acestuia aduce informații mult mai ample decât găsirea soluției propriuzise, este mult mai maleabil în cazul modificărilor ulterioare ale datelor problemei (vezi analiza sensibilității soluției la datele problemei), se pretează mult mai bine la interpretări economice, este ușor de scris un program de calculator asociat, este implementat pe foarte multe calculatoare și în plus, așa cum aminteam mai sus, încă **nu a apărut o problemă practică** în fața căruia să clacheze. Nii algoritmi rămân doar ca alternative teoretice sau pentru cazurile în care algoritmul simplex este lent, dar ei nu-l pot înlocui complet.

Fundamentarea matematică a algoritmului simplex

Algoritmul simplex pleacă de la presupunerea că **dispunem deja** de o soluție admisibilă de bază x_B , corespunzătoare unei baze B . De asemenea, presupunem că am rezolvat sistemul $A \cdot x = b$ folosind această bază, rezultând astfel variabilele principale în funcție de cele secundare:

$$x_B = B^{-1} \cdot b - B^{-1} \cdot S \cdot x_S$$

Înlocuind variabilele, cu formula găsită, în funcția obiectiv, obținem:

$$\begin{aligned} f(x) &= c_B^T \cdot (B^{-1} \cdot b - B^{-1} \cdot S \cdot x_S) + c_S^T \cdot x_S = c_B^T \cdot B^{-1} \cdot b - (c_B^T \cdot B^{-1} \cdot S - c_S^T) \cdot x_S = \\ &= f(x^B) - (c_B^T \cdot B^{-1} \cdot S - c_S^T) \cdot x_S = f(x^B) - \Delta \cdot x_S \end{aligned}$$

unde:

- $f(x^B) = c_B^T \cdot B^{-1} \cdot b$ este valoarea funcției obiectiv în soluția de bază
- $c_B^T \cdot B^{-1} \cdot S = z$ este un vector n-m componente $z = (z_{m+1}, z_{m+2}, \dots, z_n)$.
- $\Delta = z - c_S^T$ este un vector n-m componente $\Delta = (\Delta_{m+1}, \Delta_{m+2}, \dots, \Delta_n)$

Obținem următoarea formă a problemei:

$$\begin{cases} (\max)\text{sau}(\min) f = f(x^B) - \sum_{j=m+1}^n \Delta_j \cdot x_j \\ x_i + \sum_{j=m+1}^n a_{ij} \cdot x_j = x_i \quad i = 1, \dots, m \\ x_j \geq 0, \quad j = 1, \dots, n \end{cases}$$

Din motive de organizare și concentrare a datelor problemei, Danzig le-a trecut pe acestea într-un tabel ca cel de mai jos, numit **tabel simplex**.

			c ₁	c ₂	...	c _m	c _{m+1}	...	c _n	
c _B	x _B	x _B	x ₁	x ₂	...	x _m	x _{m+1}	...	x _n	
c ₁	x ₁	B ⁻¹ ·b	I _m				B ⁻¹ ·S			
c ₂	x ₂									
⋮	⋮									
c _m	x _m									
		f(x _B)	0				z _{m+1}	...	z _n	
			0				Δ _{m+1}	...	Δ _n	

Fiecărei soluții de bază îi corespunde un tabel simplex și reciproc, deci, de fiecare dată când vom vorbi de un tabel simplex, vom spune și care este baza asociată.

Din forma funcției obiectiv se vede că:

- într-o problemă de maxim:

$$x^B \text{ este soluție optimă} \Leftrightarrow f(x^B) - \sum_{j=m+1}^n \Delta_j \cdot x_j \leq f(x^B) \text{ oricare ar fi } x_S \geq 0 \Leftrightarrow$$

$$\Leftrightarrow \sum_{j=m+1}^n \Delta_j \cdot x_j \geq 0 \text{ oricare ar fi } x_S \geq 0 \Leftrightarrow \Delta_j \geq 0, j = 1, \dots, n \text{ (toți } \Delta_j \text{ sunt pozitivi)}$$

- într-o problemă de minim:

$$x^B \text{ este soluție optimă} \Leftrightarrow f(x^B) - \sum_{j=m+1}^n \Delta_j \cdot x_j \geq f(x^B) \text{ oricare ar fi } x_S \geq 0 \Leftrightarrow$$

$$\Leftrightarrow \sum_{j=m+1}^n \Delta_j \cdot x_j \leq 0 \text{ oricare ar fi } x_S \geq 0 \Leftrightarrow \Delta_j \leq 0, j = 1, \dots, n \text{ (toți } \Delta_j \text{ negativi)}$$

Pentru a evita complicarea expunerii vom presupune în continuare că *problema este de maxim*, pentru cazul de minim raționamentul fiind analog.

Rezultatul obținut reprezintă criteriul de recunoaștere a optimalității soluției sau, mai simplu, **criteriul de optim**.

Dacă soluția nu este optimă atunci există evident o altă soluție de bază admisibilă mai bună. Se poate demonstra că dacă x și y sunt două soluții admisibile de bază cu $f(x) \leq f(y)$ și numărul de variabile principale prin care diferă cele două este k , cu $1 \leq k \leq m$, atunci există un șir de soluții admisibile de bază: $\{x_1 = x, x_2, \dots, x_k = y\}$ astfel încât:

1. $f(x_i) \leq f(x_{i+1})$ oricare ar fi $1 \leq i < k$
2. x_i diferă de x_{i+1} printr-o singură variabilă, oricare ar fi $1 \leq i < k$

Din cele de mai sus rezultă că e suficient să căutăm o soluție de bază admisibilă mai bună doar printre cele care diferă de cea actuală printr-o singură variabilă.

Trebuie deci să găsim acea variabilă principală care iese din bază și acea variabilă secundară care intră în bază. Rezultatul schimbării trebuie să fie:

1. o soluție de bază (deci coloanele corespunzătoare noii variabile să formeze un minor principal);
2. o soluție admisibilă (adică să aibă toate componentele pozitive);
3. o soluție mai bună;
4. cea mai bună posibilă dintre soluțiile mai bune.

Presupunem că înlocuim variabila principală x_i ($1 \leq i \leq m$) cu variabila secundară x_j ($m+1 \leq j \leq n$). Aceasta este echivalent cu a scoate din ecuația i (singura în care apare x_i) variabila x_j , în funcție de variabila x_i și de celelalte variabile secundare. Este evident că acest lucru (care este echivalent cu faptul că noua soluție trebuie să fie de bază) este posibil doar dacă coeficientul lui x_j din această ecuație este diferit de 0 ($a_{ij} \neq 0$). În acest caz avem succesiv:

$$x_i + \sum_{\substack{k=m+1 \\ k \neq j}}^n a_{ik} \cdot x_k + a_{ij} \cdot x_j = x_i \Leftrightarrow x_j + \sum_{\substack{k=m+1 \\ k \neq j}}^n \frac{a_{ik}}{a_{ij}} \cdot x_k + \frac{1}{a_{ij}} \cdot x_i = \frac{1}{a_{ij}} \cdot x_i \Leftrightarrow$$

$$\Leftrightarrow x_j = \frac{1}{a_{ij}} \cdot x_i - \sum_{\substack{k=m+1 \\ k \neq j}}^n \frac{a_{ik}}{a_{ij}} \cdot x_k - \frac{1}{a_{ij}} \cdot x_i$$

Înlocuind variabila x_j în celelalte ecuații obținem:

$$x_s + \sum_{\substack{k=m+1 \\ k \neq j}}^n a_{sk} \cdot x_k + a_{sj} \cdot \left(\frac{1}{a_{ij}} \cdot x_i - \sum_{\substack{k=m+1 \\ k \neq j}}^n \frac{a_{ik}}{a_{ij}} \cdot x_k - \frac{1}{a_{ij}} \cdot x_i \right) = x_s \Leftrightarrow$$

$$\Leftrightarrow x_s + \sum_{\substack{k=m+1 \\ k \neq j}}^n \left(a_{sk} - a_{sj} \cdot \frac{a_{ik}}{a_{ij}} \right) \cdot x_k - \frac{a_{sj}}{a_{ij}} \cdot x_i = x_s - \frac{a_{sj}}{a_{ij}} \cdot x_i \text{ pentru } s = 1, \dots, m \text{ și } s \neq i$$

Conform formulelor de mai sus rezultă că noua soluție de bază este admisibilă dacă:

$$x_s - \frac{a_{sj}}{a_{ij}} \cdot x_i \geq 0 \text{ oricare ar fi } s = 1, \dots, m \text{ diferit de } i \Leftrightarrow \frac{x_s}{a_{sj}} \geq \frac{x_i}{a_{ij}} \text{ oricare ar fi } s = 1, \dots, m, s \neq i$$

și în urma schimbării noua soluție de bază va fi:

$$\bar{x}_j = \frac{1}{a_{ij}} \cdot x_i \quad \text{și} \quad \bar{x}_s = x_s - \frac{a_{sj}}{a_{ij}} \cdot x_i \quad \text{pentru } s \neq i$$

În concluzie, dacă ne hotărâm să introducem în bază variabila j , **singura variabilă care o poate înlocui (pentru ca noua soluție să fie admisibilă de bază)** nu poate fi decât aceea care corespunde celui a_{ij} **strict pozitiv** din coloana j din matricea $B^{-1} \cdot S$, pentru care se obține valoarea minimă a rapoartelor dintre componentele soluției de bază actuale și componentele coloanei j . Pentru evidențierea acestora, ele se notează cu θ_s și avem:

$$\theta_i = \min_{\substack{1 \leq s \leq m \\ a_{ij} > 0}} \left(\theta_s = \frac{x_s}{a_{sj}} \right)$$

Condiția de mai sus este numită **criteriul de ieșire din bază**.

Mai înainte am văzut ce efect are schimbarea unei variabile din bază asupra sistemului. Este important însă să vedem efectul ei și asupra funcției obiectiv. Valoarea funcției obiectiv în noua soluție de bază va fi:

$$f(\bar{x}^B) = f(x^B) - \Delta_j \cdot \bar{x}_j = f(x^B) - \Delta_j \cdot \frac{1}{a_{ij}} \cdot x_i$$

Pentru ca această soluție să fie cel puțin la fel de bună trebuie ca:

$$f(\bar{x}^B) \geq f(x^B) \Leftrightarrow f(x^B) - \Delta_j \cdot \frac{1}{a_{ij}} \cdot x_i \geq f(x^B) \Leftrightarrow \Delta_j \cdot \frac{1}{a_{ij}} \cdot x_i \leq 0 \quad \begin{matrix} x_i \geq 0 \\ a_{ij} \geq 0 \end{matrix} \Leftrightarrow \Delta_j \leq 0$$

În concluzie, dacă vrem să obținem o soluție strict mai bună vom alege o variabilă x_j corespunzătoare unui $\Delta_j < 0$. Noua soluție va exista efectiv doar dacă pe coloana j din matricea $B^{-1} \cdot S$ există o componentă a_{ij} strict pozitivă și va fi efectiv strict mai bună doar dacă componenta corespunzătoare x_i din soluția de bază este diferită de 0, îmbunătățirea fiind egală cu $-\Delta_j \cdot \frac{1}{a_{ij}} \cdot x_i$.

Situația în care toate componentele coloanei j din matricea $B^{-1} \cdot S$ sunt mai mici sau egale cu 0 este lămurit de următoarea teoremă:

Teoremă: Dacă pe coloana j din matricea $B^{-1} \cdot S$, corespunzătoare unei variabile x_j cu $\Delta_j < 0$, **toate componentele sunt mai mici sau egale cu zero** atunci problema are **optim infinit**.

Demonstrație: Fie o soluție particulară a sistemului, în care variabila secundară $x_j = \lambda > 0$, λ oarecare iar celelalte sunt 0. În acest caz, variabilele principale vor fi: $y_{\lambda i} = x_i - a_{ij} \cdot \lambda$ și vor fi toate pozitive, ținând cont de faptul că toți $a_{ij} \leq 0$ și, deci, soluția va fi admisibilă. Pentru o astfel de soluție, valoarea funcției obiectiv este:

$$f(y_\lambda) = f(x^B) - \Delta_j \cdot \lambda. \quad \text{și avem:} \quad \lim_{\lambda \rightarrow \infty} f(y_\lambda) = +\infty$$

deci imaginea funcției f este nemărginită și afirmația este demonstrată.

Presupunem în continuare că există o componentă a_{ij} strict pozitivă. Deoarece dorim să trecem la cea mai bună dintre bazele posibile, vom alege dintre toate variabilele cu $\Delta_j < 0$, pe cea pentru care se obține:

$$\max_{\substack{m+1 \leq j \leq n \\ \Delta_j < 0}} \left(-\Delta_j \cdot \min_{\substack{1 \leq i \leq m \\ a_{ij} > 0}} \frac{x_i}{a_{ij}} \right)$$

Această condiție este **criteriul de intrare în bază**.

Deoarece calculul necesar găsirii acestuia este laborios, Danzig a preferat înlocuirea acestei alegeri cu alegerea celui j care corespunde celui mai negativ $\Delta_j (= - \max_{\substack{m+1 \leq j \leq n \\ \Delta_j < 0}} (-\Delta_j))$, variantă care

este ușor de aplicat, îmbunătățește soluția și prin care se obține, în marea majoritate a cazurilor, același j .

În acest moment știm cum să găsim o soluție mai bună, dacă ea există și am putea calcula din nou elementele necesare analizării noii soluții, din tabelul simplex, folosind formulele fiecăruia:

$$\begin{aligned} x_B &= B^{-1} \cdot b \\ z &= c_B^T \cdot B^{-1} \cdot A \\ \Delta &= z - c \end{aligned}$$

Acest mod de lucru este efectiv folosit în anumite variante ale algoritmului simplex (vezi forma revizuită) dar el presupune calcularea inversei matricei B , ceea ce necesită foarte mult timp. Acest motiv era suficient de convingător pentru ca Danzig să-și fi pus problema dacă nu cumva noul tabel simplex poate fi calculat pe baza fostului tabel simplex, făcând mult mai puține calcule și utilizând formule ușor de memorat și aplicat. Această întrebare era natural de pus, dacă ținem cont că noua soluție diferă de fosta soluție printr-o singură variabilă. Aceste formule au fost efectiv găsite, ele putând fi obținute urmărind efectul înlocuirii lui x_i cu x_j asupra sistemului:

- noua soluție de bază \bar{x}_B se obține din fosta soluție cu formulele deja găsite mai sus:

$$\begin{aligned} \bar{x}_j &= \frac{1}{a_{ij}} \cdot x_i \quad \text{și} \\ \bar{x}_s &= x_s - \frac{a_{sj}}{a_{ij}} \cdot x_i \quad \text{pentru } s \neq i \end{aligned}$$

- noii coeficienții ai variabilelor în cele m ecuații vor fi:

1. variabilă principală x_s va avea coeficientul 1 în ecuația k și 0 în celelalte ecuații;

2. o variabilă secundară x_k , diferită de x_i , va avea coeficienții $\boxed{a_{sk} - a_{sj} \cdot \frac{a_{ik}}{a_{ij}}}$ în

ecuațiile $s \neq i$ și coeficientul $\boxed{\frac{a_{ik}}{a_{ij}}}$ în ecuația i .

3. noua variabilă secundară x_i va avea coeficienții $\boxed{-\frac{a_{sj}}{a_{ij}}}$ în ecuațiile $s \neq i$ și coeficien-

tul $\boxed{\frac{1}{a_{ij}}}$ în ecuația i .

4. noua valoare a funcției obiectiv va fi: $\boxed{f(\bar{x}^B) = f(x^B) - \Delta_j \cdot \frac{1}{a_{ij}} \cdot x_i}$

5. noii Δ_j îi obținem înlocuind în funcția obiectiv variabila x_j în funcție de x_i :

$$f(x) = f(x^B) - \sum_{\substack{k=m+1 \\ k \neq j}}^n \Delta_k \cdot x_k - \Delta_j \cdot \left(- \sum_{\substack{k=m+1 \\ k \neq j}}^n \frac{a_{ik}}{a_{ij}} \cdot x_k - \frac{1}{a_{ij}} \cdot x_i + \frac{1}{a_{ij}} \cdot x_i \right) =$$

$$\begin{aligned}
 &= f(x^B) - \Delta_j \cdot \frac{1}{a_{ij}} \cdot x_i - \sum_{\substack{k=m+1 \\ k \neq j}}^n \left(\Delta_k - \Delta_j \frac{a_{ik}}{a_{ij}} \right) \cdot x_k - \left(-\frac{\Delta_j}{a_{ij}} \right) \cdot x_i = \\
 &= f(\bar{x}^B) - \sum_{\substack{k=m+1 \\ k \neq j}}^n \left(\Delta_k - \Delta_j \frac{a_{ik}}{a_{ij}} \right) \cdot x_k - \left(-\frac{\Delta_j}{a_{ij}} \right) \cdot x_i \\
 &\text{de unde rezultă că } \boxed{\bar{\Delta}_k = \Delta_k - \Delta_j \frac{a_{ik}}{a_{ij}}} \text{ pentru } k \neq i \text{ și } \boxed{\bar{\Delta}_i = -\frac{\Delta_j}{a_{ij}}}
 \end{aligned}$$

Deși par să fie foarte multe formule complicate, frumusețea algoritmului și meritul lui Danzig constă tocmai în faptul că toate respectă o regulă vizuală ușor de memorat, numită **regula dreptunghiului**, așa cum se va vedea în algoritmul simplex.

ALGORITMUL SIMPLEX

Reamintim că presupunem că problema este la forma standard **de maxim** și că dispunem de o soluție de bază admisibilă.

Pasul 1. Se construiește tabelul simplex corespunzător bazei de care dispunem în ordinea următoare:

1. pe linia a doua se trec toate variabilele într-o ordine oarecare;
2. pe prima linie se trec coeficienții funcției obiectiv, fiecare deasupra variabilei corespunzătoare;
3. se construiește matricea A, fiecare coloană fiind formată din coeficienții unei variabile din toate ecuațiile (ordinea în care se parcurg ecuațiile trebuie să fie aceeași pentru toate variabilele), având grijă ca, coloanele să fie trecute în ordinea în care au fost trecute variabilele pe linia 2;
4. se construiește baza B, ordinea coloanelor fiind cea în care apar ele în matricea A;
5. se calculează B^{-1} ;
6. se calculează $B^{-1} \cdot A$ și se trece în partea centrală a tabelului;
7. se trec variabilele principale în a doua coloană, în așa fel încât, la intersecția liniei și coloanei corespunzătoare acestei variabile, să se afle valoarea 1.
8. se trec în prima coloană coeficienții corespunzători variabilelor principale din funcția obiectiv, fiecare în dreptul variabilei corespunzătoare;
9. se calculează soluția de bază cu formula $B^{-1} \cdot b$, având grijă ca ordinea în care au fost trecuți termenii liberi în vectorul b să fie aceeași cu ordinea în care au fost parcurse ecuațiile la formarea matricii A;
10. se trec în a treia coloană valorile variabilelor principale din soluția de bază, fiecare în dreptul variabilei corespunzătoare;
11. se calculează $f(x_B)$ înmulțind două câte două componentele coloanei 1 cu cele din coloana 3 aflate pe aceeași linie și adunând toate produsele între ele (adică facem produsul scalar dintre c_B și x_B);
12. se calculează pe rând fiecare z_j $j = 1, \dots, n$ un z_j obținându-se înmulțind scalar c_B cu coloana j din $B^{-1} \cdot A$ aflată în centrul tabelului (această linie se calculează și se trece doar în primul tabel, scopul ei fiind calcularea lui Δ);
13. se calculează pe rând fiecare Δ_j $j = 1, \dots, n$ scăzând din linia lui z linia lui c ($\Delta_j = z_j - c_j$)

Pasul 2. Se analizează valorile Δ_j corespunzătoare variabilelor secundare (e ușor de văzut că întotdeauna, cei corespunzători variabilelor principale sunt toți 0, deci neinteresanți).

- dacă toți sunt mai mari sau egali cu 0 atunci soluția actuală este cea optimă. Dacă există $\Delta_j = 0$ în afara bazei, atunci pot apărea două cazuri:

- 1) toate elementele din coloana a_j din $B^{-1} \cdot A$ sunt mai mici sau egale cu 0. Atunci toate soluțiile de forma $x_B - a_j \cdot \lambda$ sunt soluții optime, unde $\lambda > 0$ oarecare;
- 2) există o componentă a_{ij} a coloanei a_j strict pozitivă. Atunci introducând variabila x_j în bază în locul variabilei principale x_i obținem altă soluție de bază optimă.

Dacă apar numai cazuri de tipul 2), obținem toate soluțiile de bază optime, mulțimea tuturor soluțiilor optime fiind formată din toate combinațiile convexe ale acestora. Dacă apare și cazul 1) atunci *mulțimea soluțiilor optime este nemărginită* fiind formată din combinațiile convexe ale soluțiilor de forma $x_B - a_j \cdot \lambda$ unde x_B sunt toate soluțiile optime de bază.

- dacă există $\Delta_j < 0$ atunci îl alegem pe cel mai negativ: $\Delta_k = \min_{\substack{m+1 \leq j \leq n \\ \Delta_j < 0}} \Delta_j$. Variabila x_j

va fi cea care intră în noua bază. Dacă minimum este multiplu atunci alegem, la întâmplare, unul dintre aceștia (cei minimi).

Pasul 3. Se analizează elementele coloanei a_j din $B^{-1} \cdot A$, corespunzătoare variabilei alese la pasul 2.

- Dacă toate sunt mai mici sau egale cu 0 atunci problema are optim infinit și algoritmul ia sfârșit;
- Dacă există componente strict pozitive, pentru acestea se calculează rapoartele $\theta_s = \frac{x_s}{a_{sj}}$. Variabila x_i corespunzătoare raportului minim este cea care va ieși din

bază. Dacă acest minim este multiplu sunt posibile două cazuri:

- 1) minimum este strict pozitiv. În acest caz se alege unul dintre aceștia la întâmplare;
- 2) minimum este 0. Atunci x_i corespunzătorii sunt 0, deci soluția este degenerată și noua soluție este la fel de bună. Această situație poate duce la ciclarea algoritmului și alegerea la întâmplare nu mai este suficientă, fiind nevoie de o regulă de alegere suplimentară care să evite ciclarea. O metodă de alegere se bazează pe faptul că, așa cum vom vedea, întotdeauna prima bază este matricea unitate și în dreptul ei, în toate tabelele simplex, se va afla inversa bazei corespunzător fiecăruia. În acest caz, pentru pozițiile în care s-a obținut minimumul **împărțim prima coloană a lui B^{-1} la coloana a_j** și alegem minimumul dintre aceste rapoarte. Dacă minimumul dintre aceștia este tot multiplu continuăm procedeul, pentru pozițiile ce dau noul minim, cu coloana a doua din B^{-1} și așa mai departe, până minimumul rămâne unic. Nu este posibil să se epuizeze toate coloanele lui B^{-1} și minimumul să rămână multiplu, deoarece, în acest caz, am avea:

$$\frac{b_{i_1 j_k}}{a_{i_1 j}} = \frac{b_{i_2 j_k}}{a_{i_2 j}}, \text{ pentru toți indicii } j_k \text{ ai coloanelor lui } B^{-1}, i_1 \text{ și } i_2 \text{ fiind doi}$$

din indicii care dau același minim până la sfârșit sau altfel scris

$$\frac{b_{i_1 j_k}}{b_{i_2 j_k}} = \frac{a_{i_1 j}}{a_{i_2 j}} \text{ pentru orice } j_k \Rightarrow \text{liniile } i_1 \text{ și } i_2 \text{ din } B^{-1} \text{ sunt proporționale,}$$

fapt ce contrazice faptul că B^{-1} este inversabilă.

Pasul 4. Se calculează componentele tabelului simplex corespunzător noii baze pe baza tabelului actual și folosind următoarele reguli:

1. se încadrează elementul a_{ij} , aflat la intersecția coloanei variabilei care intră în bază

cu linia variabilei care iese din bază, care a fost numit de Danzig **pivot**, într-un dreptunghi

2. coloana pivotului va avea 1 în dreptul pivotului și 0 în rest (inclusiv Δ_j);
3. linia pivotului este linia actuală împărțită la pivot (inclusiv în soluția de bază);
4. restul elementelor se calculează cu regula dreptunghiului (inclusiv soluția de bază, Δ și $f(x_B)$). Regula dreptunghiului se bazează pe observația că în toate formulele prin care se calculează acestea nu apare decât valoarea lor actuală, pivotul și cele două elemente care ar completa dreptunghiul ce are poziția de calculat și pivotul pe diagonală. Regula dreptunghiului se enunță literar astfel: "noua valoare = produsul dintre elementele de pe diagonala pivotului minus produsul dintre cele aflate pe cealaltă diagonală totul împărțit la pivot". (pentru scurtarea timpului de lucru se poate observa că elementele unei linii care are 0 pe coloana pivotului rămân aceleași și de asemenea elementele unei coloane care are 0 pe linia pivotului)

Operația de calculare a noului tabel prin regulile de mai sus poartă denumirea de **pivotare**.

Pasul 5. Se reia algoritmul de la pasul 2.

Exemplu: Fie problema de programare liniară:

$$\begin{aligned}
 (\max) f &= 3x_1 + 2x_2 + x_3 + 4x_4 + 3x_5 + 5x_6 \\
 \begin{cases} x_1 + 2x_2 + x_3 + x_4 + 3x_5 + x_6 \leq 8 \\ 2x_1 + 2x_2 + 3x_3 + x_4 + 3x_5 + 2x_6 \leq 15 \\ 3x_1 + x_2 + 2x_3 + 2x_4 + x_5 + 2x_6 \leq 11 \\ x_i \geq 0, \quad i = 1, \dots, 6 \end{cases}
 \end{aligned}$$

pentru care știm că baza formată din coloanele a_1, a_2, a_3 este admisibilă. Pentru rezolvare vom aduce problema la forma standard de maxim introducând variabilele de abatere x_7, x_8, x_9 obținând:

$$\begin{aligned}
 (\max) f &= 3x_1 + 2x_2 + x_3 + 4x_4 + 3x_5 + 5x_6 \\
 \begin{cases} x_1 + 2x_2 + x_3 + x_4 + 3x_5 + x_6 + x_7 = 8 \\ 2x_1 + 2x_2 + 3x_3 + x_4 + 3x_5 + 2x_6 + x_8 = 15 \\ 3x_1 + x_2 + 2x_3 + 2x_4 + x_5 + 2x_6 + x_9 = 11 \\ x_i \geq 0, \quad i = 1, \dots, 9 \end{cases}
 \end{aligned}$$

Vom așeza în tabelul simplex variabilele în ordinea indicilor și vom avea:

$$c = \begin{pmatrix} 3 \\ 2 \\ 1 \\ 4 \\ 3 \\ 5 \\ 0 \\ 0 \\ 0 \end{pmatrix}, A = \begin{pmatrix} 1 & 2 & 1 & 1 & 3 & 1 & 1 & 0 & 0 \\ 2 & 2 & 3 & 1 & 3 & 2 & 0 & 1 & 0 \\ 3 & 1 & 2 & 2 & 1 & 2 & 0 & 0 & 1 \end{pmatrix}, B = \begin{pmatrix} 1 & 2 & 1 \\ 2 & 2 & 3 \\ 3 & 1 & 2 \end{pmatrix}, c_B = \begin{pmatrix} 3 \\ 2 \\ 1 \end{pmatrix}$$

$$\text{vom calcula matricea } B^{-1} = \begin{pmatrix} \frac{1}{7} & -\frac{3}{7} & \frac{4}{7} \\ \frac{5}{7} & -\frac{1}{7} & -\frac{1}{7} \\ -\frac{4}{7} & \frac{5}{7} & -\frac{2}{7} \end{pmatrix} \text{ și pe baza acesteia soluția de bază } x_B = B^{-1} \cdot b = \begin{pmatrix} 1 \\ 2 \\ 3 \end{pmatrix} \text{ și}$$

matricea $B^{-1} \cdot A = \begin{pmatrix} 1 & 0 & 0 & \frac{6}{7} & -\frac{2}{7} & \frac{3}{7} & \frac{1}{7} & -\frac{3}{7} & \frac{4}{7} \\ 0 & 1 & 0 & \frac{2}{7} & \frac{11}{7} & \frac{1}{7} & \frac{5}{7} & -\frac{1}{7} & -\frac{1}{7} \\ 0 & 0 & 1 & -\frac{3}{7} & \frac{1}{7} & \frac{2}{7} & -\frac{4}{7} & \frac{5}{7} & -\frac{2}{7} \end{pmatrix}$ care se trec în tabelul simplu:

CB	XB	xB	3	2	1	4	3	5	0	0	0
			x1	x2	x3	x4	x5	x6	x7	x8	x9
3	x1	1	1	0	0	$\frac{6}{7}$	$-\frac{2}{7}$	$\frac{3}{7}$	$\frac{1}{7}$	$-\frac{3}{7}$	$\frac{4}{7}$
2	x2	2	0	1	0	$\frac{2}{7}$	$\frac{11}{7}$	$\frac{1}{7}$	$\frac{5}{7}$	$-\frac{1}{7}$	$-\frac{1}{7}$
1	x3	3	0	0	1	$-\frac{3}{7}$	$\frac{1}{7}$	$\frac{2}{7}$	$-\frac{4}{7}$	$\frac{5}{7}$	$-\frac{2}{7}$

și în final se vor calcula valoarea funcției obiectiv în această soluție, z_j și Δ_j :

CB	XB	xB	3	2	1	4	3	5	0	0	0
			x1	x2	x3	x4	x5	x6	x7	x8	x9
3	x1	1	1	0	0	$\frac{6}{7}$	$-\frac{2}{7}$	$\frac{3}{7}$	$\frac{1}{7}$	$-\frac{3}{7}$	$\frac{4}{7}$
2	x2	2	0	1	0	$\frac{2}{7}$	$\frac{11}{7}$	$\frac{1}{7}$	$\frac{5}{7}$	$-\frac{1}{7}$	$-\frac{1}{7}$
1	x3	3	0	0	1	$-\frac{3}{7}$	$\frac{1}{7}$	$\frac{2}{7}$	$-\frac{4}{7}$	$\frac{5}{7}$	$-\frac{2}{7}$
		10	3	2	1	$\frac{19}{7}$	$\frac{17}{7}$	$\frac{13}{7}$	$\frac{9}{7}$	$-\frac{6}{7}$	$\frac{8}{7}$
			0	0	0	$-\frac{9}{7}$	$-\frac{4}{7}$	$-\frac{22}{7}$	$\frac{9}{7}$	$-\frac{6}{7}$	$\frac{8}{7}$

Din tabel se observă că există $\Delta_j < 0$, aceștia fiind $\Delta_4, \Delta_5, \Delta_6, \Delta_8$ iar minimum lor este Δ_6 .

Observație Dacă vom căuta acel Δ_j care dă cea mai bună îmbunătățire vom avea de găsit

acel Δ_j dintre cei negativi pentru care se obține $\max_{\substack{4 \leq j \leq 9 \\ \Delta_j < 0}} \left(-\Delta_j \cdot \min_{\substack{1 \leq i \leq 3 \\ a_{ij} > 0}} \frac{x_i}{a_{ij}} \right)$ și deci de calculat:

$$-\Delta_4 \cdot \min_{\substack{1 \leq i \leq 3 \\ a_{i4} > 0}} \frac{x_i}{a_{i4}} = \frac{9}{7} \cdot \min \left(\frac{1}{\frac{6}{7}}, \frac{2}{\frac{2}{7}} \right) = \frac{3}{2}$$

$$-\Delta_5 \cdot \min_{\substack{1 \leq i \leq 3 \\ a_{i5} > 0}} \frac{x_i}{a_{i5}} = \frac{4}{7} \cdot \min \left(\frac{2}{\frac{11}{7}}, \frac{3}{\frac{1}{7}} \right) = \frac{8}{11}$$

$$-\Delta_6 \cdot \min_{\substack{1 \leq i \leq 3 \\ a_{i6} > 0}} \frac{x_i}{a_{i6}} = \frac{22}{7} \cdot \min \left(\frac{1}{\frac{3}{7}}, \frac{2}{\frac{1}{7}}, \frac{3}{\frac{2}{7}} \right) = \frac{22}{3}$$

$$-\Delta_8 \cdot \min_{\substack{1 \leq i \leq 3 \\ a_{i8} > 0}} \frac{x_i}{a_{i8}} = \frac{6}{7} \cdot \min \left(\frac{3}{\frac{5}{7}} \right) = \frac{18}{5}$$

și în final $\max \left(\frac{3}{2}, \frac{8}{11}, \frac{22}{3}, \frac{18}{5} \right) = \frac{22}{3}$ și corespunde tot lui Δ_6 .

În concluzie, soluția actuală nu este optimă și soluția cea mai bună dintre cele posibile ca succesoare va avea variabila x_6 printre cele principale.

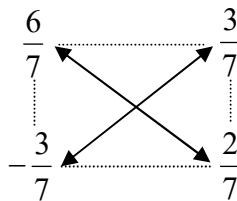
Analizând coloana a_6 observăm că există componente strict pozitive (de fapt, în acest caz sunt chiar toate) și calculăm pentru acestea rapoartele θ_i obținând:

$$\theta_1 = \frac{1}{\frac{3}{7}} = \frac{7}{3}, \quad \theta_2 = \frac{2}{\frac{1}{7}} = 14, \quad \theta_3 = \frac{3}{\frac{2}{7}} = \frac{21}{2}$$

deci minimumul corespunde variabilei x_1 și aceasta este cea care va ieși din bază. În acest moment cunoaștem noua bază și vom construi tabelul simplex pe baza regulilor de calcul de la pasul 4:

1. pivotul este $a_{16} = \frac{3}{7}$

4. de exemplu, pentru elementul de pe poziția a_{34} avem dreptunghiul:



și noua valoare de pe această poziție va fi: $\frac{3}{7} \cdot \left(-\frac{3}{7} \right) - \frac{6}{7} \cdot \frac{2}{7} = -1$

În final, tabelul corespunzător noii baze va fi:

			3	2	1	4	3	5	0	0	0
C _B	X _B	x _B	x ₁	x ₂	x ₃	x ₄	x ₅	x ₆	x ₇	x ₈	x ₉
5	x ₆	$\frac{7}{3}$	$\frac{7}{3}$	0	0	2	$-\frac{2}{3}$	1	$\frac{1}{3}$	-1	$\frac{4}{3}$
		$\frac{3}{3}$	$\frac{3}{3}$				$\frac{5}{3}$	0	$\frac{2}{3}$	0	$\frac{1}{3}$
2	x ₂	$\frac{5}{3}$	$-\frac{1}{3}$	1	0	0	$\frac{5}{3}$	0	$\frac{2}{3}$	0	$-\frac{1}{3}$
		$\frac{3}{3}$	$-\frac{3}{3}$				$\frac{3}{3}$	0	$\frac{3}{3}$		$-\frac{3}{3}$
1	x ₃	$\frac{7}{3}$	$-\frac{2}{3}$	0	1	-1	$\frac{1}{3}$	0	$-\frac{2}{3}$	1	$-\frac{2}{3}$
		$\frac{3}{3}$	$-\frac{3}{3}$				$\frac{3}{3}$	0	$-\frac{2}{3}$		$-\frac{3}{3}$
		$\frac{52}{3}$	$\frac{22}{3}$	0	0	5	$-\frac{8}{3}$	0	$\frac{7}{3}$	-4	$\frac{16}{3}$

Continuând algoritmul vom găsi tablele:

			3	2	1	4	3	5	0	0	0
C _B	X _B	x _B	x ₁	x ₂	x ₃	x ₄	x ₅	x ₆	x ₇	x ₈	x ₉
5	x ₆	$\frac{14}{3}$	$\frac{5}{3}$	0	1	1	$-\frac{1}{3}$	1	$-\frac{1}{3}$	0	$\frac{2}{3}$
2	x ₂	$\frac{5}{3}$	$-\frac{1}{3}$	1	0	0	$\frac{5}{3}$	0	$\frac{2}{3}$	0	$-\frac{1}{3}$
0	x ₈	$\frac{7}{3}$	$-\frac{2}{3}$	0	1	-1	$\frac{1}{3}$	0	$-\frac{2}{3}$	1	$-\frac{2}{3}$
			$\frac{80}{3}$	$\frac{14}{3}$	0	4	1	$-\frac{4}{3}$	0	$-\frac{1}{3}$	$\frac{8}{3}$

			3	2	1	4	3	5	0	0	0
C _B	X _B	x _B	x ₁	x ₂	x ₃	x ₄	x ₅	x ₆	x ₇	x ₈	x ₉
5	x ₆	5	$\frac{8}{5}$	$\frac{1}{5}$	1	1	0	1	$-\frac{1}{5}$	0	$\frac{3}{5}$
3	x ₅	1	$-\frac{1}{5}$	$\frac{3}{5}$	0	0	1	0	$\frac{2}{5}$	0	$-\frac{1}{5}$
0	x ₈	2	$-\frac{3}{5}$	$-\frac{1}{5}$	1	-1	0	0	$-\frac{4}{5}$	1	$-\frac{3}{5}$
			28	$\frac{22}{5}$	$\frac{4}{5}$	4	1	0	$\frac{1}{5}$	0	$\frac{2}{5}$

Ultimul tabel conține soluția optimă, deoarece toți $\Delta_j \geq 0$. Deoarece nu mai există nici un $\Delta_j = 0$ în afara celor din bază rezultă că soluția optimă este unică.

Determinarea unei soluții de bază admisibile de start

Presupunem încă odată că problema este la forma standard.

Algoritmul simplex necesită, pentru pornire, o soluție admisibilă de bază. Găsirea acesteia pur și simplu prin încercări nu este deloc o sarcină ușoară, gândindu-ne că aceasta presupune găsirea unui minor principal, inversarea acestuia și calcularea soluției, abia în acest moment putând vedea dacă aceasta are toate componentele pozitive, această căutare putând dura foarte mult.

Rezolvarea problemei pleacă de la observația că singura bază pentru care calculul de mai sus se poate face imediat este matricea unitate, caz în care soluția de bază corespunzătoare este chiar vectorul termenilor liberi. Aceasta presupune ca problema să aibă toți termenii liberi mai mari sau egali cu 0 și în matricea A să existe toate coloanele matricii unitate.

Dacă toți termenii liberi pot fi făcuți mai mari sau egali cu 0 foarte simplu, prin înmulțirea eventual cu -1 a restricției respective, existența tuturor coloanelor matricii unitate este evident că este foarte puțin probabilă și mai greu de obținut.

În acest sens plecăm de la observația că existența unui vector din coloana unitate printre coloanele matricii A este echivalentă cu existența unei variabile care apare doar în ecuația corespunzătoare lui 1 din acel vector, cu coeficientul 1. Acest lucru poate fi obținut în două moduri:

- a) Începând de la prima ecuație, căutăm o variabilă care are coeficientul de același semn cu termenul liber, o scoatem din această ecuație în funcție de celelalte variabile, o înlocuim în celelalte și repetăm procedeul pornind de la a doua ecuație. Pot apărea trei cazuri:
- 1) la un moment dat obținem o ecuație în care toți coeficienții variabilelor au semn contrar termenului liber, măcar unul dintre toți fiind diferit de 0. În acest caz ecuația nu are evident soluție admisibilă (pozitivă) și deci problema nu are soluție;
 - 2) toți coeficienții variabilelor și termenul liber sunt 0. În acest caz rezultă că această ecuație rezultă din cele anterioare și ea va fi eliminată, trecându-se la următoarea;
 - 3) se epuizează toate ecuațiile. În acest moment, variabilele care au fost scoase din fiecare ecuație formează baza dorită.

Procedeul de mai sus poate părea atractiv, dar presupune introducerea unui algoritmul diferit de simplex, cu efect asupra omogenității calculelor și a vitezei de lucru. De asemenea, prin efectuarea calculelor de mai sus, datele problemei nu mai au semnificația economică inițială, ne mai putându-se face interpretări economice.

- b) Pentru toți vectorii coloană introducem în ecuațiile corespunzătoare câte o variabilă cu coeficientul 1. Vom obține evident un nou sistem de restricții și rămâne de văzut ce legătură este între soluțiile acestuia și cel inițial. Cele două sisteme au forma:

$$A \cdot x = b \quad \text{și} \quad A \cdot x + y = b$$

Este evident că o soluție a primului sistem este soluție și a celui de-al doilea (luăm $y = 0$) iar o soluție a celui de-al doilea este soluție și pentru primul doar dacă $y = 0$. Scopul nostru va fi deci, ca pornind de la soluția inițială a celei de-a doua probleme să ajungem la o soluție a acesteia în care $y = 0$. Ținând cont că în soluțiile de bază, variabilele secundare sunt toate egale cu 0, vom încerca să scoatem din bază variabilele y . Scopul algoritmului simplex este însă maximizarea funcției obiectiv, nu scoaterea a uneia sau alteia din variabile din bază. Pentru a echivala cele două scopuri putem proceda în două feluri:

- 1) Alegem o nouă funcție obiectiv care să-și atingă extremul printre soluțiile pozitive chiar pentru $y = 0$ și în momentul când am obținut soluția respectivă pornim cu aceasta ca soluție inițială algoritmul simplex pentru fosta problemă.
- 2) Adăugăm la fosta funcție obiectiv noile variabile cu niște coeficienți de asemenea natură aleși încât aportul variabilelor y la valoarea funcției să fie contrar scopului dorit (foarte mari pozitivi într-o problemă de minim și foarte mari negativi într-o problemă de maxim).

Vom detalia în continuare cele două metode:

Algoritmul simplex în două faze

Data problema de programare liniară la forma standard de maxim:

$$\begin{cases} (\max) f = c^T \cdot x \\ A \cdot x = b \\ x \geq 0 \end{cases}$$

în care am aranjat deja ca toți termenii liberi să fie pozitivi ($b \geq 0$).

Faza 1 Construim problema:

$$\begin{cases} (\min) g = y \\ A \cdot x + y = b \\ x, y \geq 0 \end{cases}$$

pe care o rezolvăm cu algoritmul simplex pornind rezolvarea de la baza matrice unitate, putând ajunge la două situații:

- 1) **minimul funcției g este strict pozitiv.** Aceasta este echivalent cu faptul că egalitatea $Ax + y = b$ se poate obține doar pentru $y > 0$ sau altfel spus $Ax > b$ pentru orice $x \geq 0$, deci sistemul $Ax = b$ nu are soluții admisibile și în concluzie **problema inițială nu are soluție.**
- 2) **minimul funcției g este 0.** În acest caz, soluția optimă obținută are $y = 0$, deci verifică $Ax = b$ fiind în concluzie o soluție admisibilă de bază a primei probleme.

Faza 2 Începând de la soluția găsită la faza 1 se rezolvă problema inițială cu algoritmul simplex.

Dezavantajul metodei constă în faptul că tabelul simplex final de la faza 1 trebuie modificat pentru a obține tabelul simplex inițial de la faza 2 (vectorii x , c , c_B , z , Δ , $f(x_B)$, se elimină coloanele corespunzătoare lui y) și în plus nu vom mai avea în tabelele simplex ale problemei inițiale inversa bazei (care se găsea în dreptul coloanelor matricii unitate din prima fază) necesară în anumite variante ale algoritmului simplex.

Metoda bazei artificiale (metoda penalizării)

Data problema de programare liniară la forma standard de maxim:

$$\begin{cases} (\max) f = c^T \cdot x \\ A \cdot x = b \\ x \geq 0 \end{cases}$$

în care am aranjat deja ca toți termenii liberi să fie pozitivi ($b \geq 0$).

Construim problema:

$$\begin{cases} (\max) g = c^T x - My \\ A \cdot x + y = b \\ x, y \geq 0 \end{cases}$$

în care M este o constantă presupusă foarte mare (mai mare decât orice constantă care ar putea apare în rezolvarea problemei). Rezolvăm problema cu algoritmul simplex pornind rezolvarea de la baza matrice unitate, putând ajunge la trei situații:

- 1) problema are optim infinit. În acest caz, problema inițială are optim infinit.
- 2) problema are optim finit și în soluția de bază avem cel puțin o variabilă din vectorul y . În acest caz problema inițială nu are soluții admisibile.
- 3) problema are optim finit și în soluția de bază nu avem nici o variabilă din vectorul y . În acest caz problema inițială are optim finit, soluția optimă și maximul funcției fiind aceleași cu cele ale problemei modificate.

În final vom remarca faptul că variabilele y nu corespund unor mărimi economice ca celelalte, ele fiind introduse doar ca un artificiu de calcul pentru a putea porni algoritmul simplex. Din acest motiv ele se numesc **variabile artificiale.**

Exemplu Fie problema de programare liniară:

$$(\max)f = 2x_1 + 3x_2$$

$$\begin{cases} 3x_1 + x_2 \leq 10 \\ x_1 + 4x_2 \geq 2 \\ x_1, x_2 \geq 0 \end{cases}$$

Forma standard a problemei va fi:

$$(\max)f = 2x_1 + 3x_2$$

$$\begin{cases} 3x_1 + x_2 + x_3 = 10 \\ x_1 + 4x_2 - x_4 = 2 \\ x_1, x_2, x_3, x_4 \geq 0 \end{cases}$$

Avem deja termenii liberi și o coloană a matricii unitate $\begin{pmatrix} 1 \\ 0 \end{pmatrix}$ corespunzătoare variabilei x_3 .

Pentru a obține și a doua coloană $\begin{pmatrix} 0 \\ 1 \end{pmatrix}$ vom introduce variabila x_5 cu coeficientul 1 în a doua ecuație și în final vom avea de rezolvat problema:

Algoritm simplex în două faze	Metoda bazei artificiale
$(\min)g = x_5$ $\begin{cases} 3x_1 + x_2 + x_3 = 10 \\ x_1 + 4x_2 - x_4 + x_5 = 2 \\ x_1, x_2, x_3, x_4, x_5 \geq 0 \end{cases}$	$(\max)f = 2x_1 + 3x_2 - Mx_5$ $\begin{cases} 3x_1 + x_2 + x_3 = 10 \\ x_1 + 4x_2 - x_4 + x_5 = 2 \\ x_1, x_2, x_3, x_4, x_5 \geq 0 \end{cases}$

Aplicând algoritmul simplex în două faze vom obține în prima fază succesiunea de tabele:

C_B	X_B	x_B	0	0	0	0	1
			x_1	x_2	x_3	x_4	x_5
0	x_3	10	3	1	1	0	0
1	x_5	2	1	4	0	-1	1
		2	1	4	0	-1	1
			1	4	0	-1	0

C_B	X_B	x_B	0	0	0	0	1
			x_1	x_2	x_3	x_4	x_5
0	x_3	$\frac{19}{2}$	$\frac{11}{4}$	0	1	$\frac{1}{4}$	$-\frac{1}{4}$
0	x_2	$\frac{1}{2}$	$\frac{1}{4}$	1	0	$-\frac{1}{4}$	$\frac{1}{4}$
		0	0	0	0	0	-1

Am obținut optimul egal cu 0 în soluția de bază (x_3, x_2) care va fi soluția inițială pentru algoritmul simplex aplicat problemei inițiale în a doua fază. Eliminăm din tabel coloana lui x_5 , înlocuim valorile coeficienților funcției obiectiv și deci și valoarea acesteia, valorile Δ și obținem tabelul:

c_B	x_B	x_B	2 x_1	3 x_2	0 x_3	0 x_4
0	x_3	$\frac{19}{2}$	$\frac{11}{4}$	0	1	$\frac{1}{4}$
3	x_2	$\frac{1}{2}$	$\frac{1}{4}$	1	0	$-\frac{1}{4}$
		$\frac{3}{2}$	$-\frac{5}{4}$	0	0	$-\frac{3}{4}$

c_B	x_B	x_B	2 x_1	3 x_2	0 x_3	0 x_4
0	x_3	4	0	-11	1	3
2	x_1	2	1	4	0	-1
		4	0	5	0	-2

c_B	x_B	x_B	2 x_1	3 x_2	0 x_3	0 x_4
0	x_4	$\frac{4}{3}$	0	$-\frac{11}{3}$	$\frac{1}{3}$	1
2	x_1	$\frac{10}{3}$	1	$\frac{1}{3}$	$\frac{1}{3}$	0
		$\frac{20}{3}$	0	$-\frac{7}{3}$	$\frac{2}{3}$	0

c_B	x_B	x_B	2 x_1	3 x_2	0 x_3	0 x_4
0	x_4	38	11	0	4	1
3	x_2	10	3	1	1	0
		30	7	0	3	0

Soluția optimă a primei probleme este deci $x_1 = 0$ și $x_2 = 10$ care dă un maxim al funcției egal cu 30. Dacă aplicăm **a doua metodă** vom obține succesiv tablele:

c_B	x_B	x_B	2 x_1	3 x_2	0 x_3	0 x_4	-M x_5
0	x_3	10	3	1	1	0	0
-M	x_5	2	1	4	0	-1	1
		-2M	-M	-4M	0	M	-M
			-M-2	-4M-3	0	M	0

c_B	x_B	x_B	2 x_1	3 x_2	0 x_3	0 x_4	-M x_5
0	x_3	$\frac{19}{2}$	$\frac{11}{4}$	0	1	$\frac{1}{4}$	$-\frac{1}{4}$
3	x_2	$\frac{1}{2}$	$\frac{1}{4}$	1	0	$-\frac{1}{4}$	$\frac{1}{4}$
		$\frac{3}{2}$	$-\frac{5}{4}$	0	0	$-\frac{3}{4}$	$M + \frac{3}{4}$

c _B	x _B	x _B	2	3	0	0	-M
			x ₁	x ₂	x ₃	x ₄	x ₅
0	x ₃	4	0	-11	1	3	-3
2	x ₁	2	1	4	0	-1	1
		4	0	5	0	-2	2+M

c _B	x _B	x _B	2	3	0	0	-M
			x ₁	x ₂	x ₃	x ₄	x ₅
0	x ₄	$\frac{4}{3}$	0	$-\frac{11}{3}$	$\frac{1}{3}$	1	-1
2	x ₁	$\frac{10}{3}$	1	$\frac{1}{3}$	$\frac{1}{3}$	0	0
		$\frac{20}{3}$	0	$-\frac{7}{3}$	$\frac{2}{3}$	0	M

c _B	x _B	x _B	2	3	0	0	-M
			x ₁	x ₂	x ₃	x ₄	x ₅
0	x ₄	38	11	0	4	1	-1
3	x ₂	10	3	1	1	0	0
		30	7	0	3	0	M

Rezultatul final este evident același.

VARIANTE ALE ALGORITMULUI SIMPLEX

1. Algoritmul simplex dual

Pentru expunerea acestui algoritm trebuie introduse noțiunile de bază dual admisibilă și soluție dual admisibilă de bază. Prin definiție numim:

soluție de bază dual admisibilă = soluție de bază pentru care toți $\Delta_j \geq 0$
bază dual admisibilă = bază corespunzătoare unei soluții de bază dual admisibile

Ca și în cazul algoritmului simplex, pentru a putea rezolva problema cu algoritmul simplex dual trebuie să dispunem deja de o bază inițială dual admisibilă. Această soluție poate exista ca urmare a unor calcule anterioare (vezi capitolul cu reoptimizarea) sau, ca și în cazul algoritmului simplex, poate fi aplicată o procedură prin care să găsim această bază. Prezentarea acesteia este mai complicată decât cea de la algoritmul simplex astfel încât:

- dacă dispunem de o bază dual admisibilă vom rezolva problema cu algoritmul simplex dual
- dacă nu dispunem de o bază dual admisibilă vom rezolva problema cu algoritmul simplex.

Pentru a evita orice confuzie, vom numi în continuare algoritmul simplex obișnuit **algoritm simplex primal** și o bază admisibilă **bază primal admisibilă**.

Se observă că o soluție dual admisibilă nu este în general primal admisibilă și reciproc, iar o soluție care este simultan primal și dual admisibilă este o soluție optimă și reciproc.

Presupunem că am adus problema la forma standard de maxim și dispunem de o bază dual

admisibilă și dispunem deja de o bază dual admisibilă. Algoritmul simplex dual constă în următorii pași:

Pasul 1. Se construiește tabelul simplex asociat acestei baze, la fel ca și la algoritmul simplex primal;

Pasul 2. Se analizează componentele soluției:

– Dacă **toate componentele sunt mai mari sau egale cu 0** atunci soluția este și primal admisibilă, deci optimă.

– Dacă **există componente strict negative**, variabila corespunzătoare celei mai negative ($x_i = \min_{1 \leq k \leq m} x_k$) este cea care iese din bază. Dacă minimul este multiplu se ia una la întâmplare.

Pasul 3. Se analizează linia l_i corespunzătoare variabilei x_i aleasă la pasul 2:

– Dacă **toate componentele a_{ij} $j = 1, \dots, n$ sunt mai mari sau egale cu 0**, atunci am avea o ecuație cu toți coeficienții necunoscutelor pozitivi și termenul liber strict negativ, deci problema nu are soluții primal admisibile.

– Dacă **există componente strict negative**, atunci pentru acestea se găsește acel indice

pentru care se obține $\min_{a_{ij} < 0} \left| \frac{\Delta_j}{a_{ij}} \right|$ (prin $|\alpha|$ am notat modulul numărului α). Dacă

minimul este multiplu alegem unul dintre aceștia la întâmplare. Variabila corespunzătoare acestuia este cea care intră în bază.

Pasul 4. Se construiește tabelul corespunzător noii baze aplicând aceleași reguli ca la algoritmul simplex primal;

Pasul 5. Se reia algoritmul de la pasul 2

2. Forma secundară

Este evident că modul de organizare al datelor în tabelul simplex nu este unicul mod de a organiza datele într-un tabel. Forma secundară este tocmai o astfel de alternativă. Presupunem și în acest caz că problema este la forma standard de maxim, că problema este la forma standard și că dispunem de o soluție inițială de bază care este primal sau dual admisibilă.

Pasul 1. Datele problemei vor fi organizate într-un tabel de forma:

			c_S
c	x	f	x_S
		$f(x_B)$	Δ_S
c_B	x_B	$x_B = B^{-1} \cdot b$	$B^{-1} \cdot S$
c_S	x_S	0	$-I_{n-m}$

Se observă că singura diferență este că la coloana variabilelor bazei din stânga se adaugă și cele secundare, pe orizontală se lasă doar variabilele secundare iar în loc de matricea unitate în dreptul variabilelor principale vom avea matricea unitate în dreptul celor secundare (dar cu minus), Δ fiind calculat la fel, neglijându-se cei din dreptul variabilelor principale, care oricum erau 0.

Pasul 2. + **Pasul 3.** Se găsește variabila care iese din bază și cea care intră în bază cu aceleași reguli ca la algoritmul simplex primal sau, după caz, ca la cel dual.

Pasul 4. Se construiește tabelul asociat noii baze aplicând regulile:

- 1) Pivotul este același ca la tabelul simplex;
- 2) Linia pivotului are -1 în dreptul pivotului și 0 în rest;
- 3) Coloana pivotului se împarte la pivotul cu semn schimbat
- 4) Celelalte elemente se calculează cu regula dreptunghiului

Pasul 5. Se reia algoritmul de la pasul 2.

Exemplu Fie problema de programare liniară rezolvată mai sus:

$$(\max)f = 2x_1 + 3x_2$$

$$\begin{cases} 3x_1 + x_2 \leq 10 \\ x_1 + 4x_2 \geq 2 \\ x_1, x_2 \geq 0 \end{cases}$$

Forma standard a problemei va fi:

$$(\max)f = 2x_1 + 3x_2$$

$$\begin{cases} 3x_1 + x_2 + x_3 = 10 \\ x_1 + 4x_2 - x_4 = 2 \\ x_1, x_2, x_3, x_4 \geq 0 \end{cases}$$

iar după introducerea variabilelor artificiale obținem:

$$(\max)f = 2x_1 + 3x_2 - Mx_5$$

$$\begin{cases} 3x_1 + x_2 + x_3 = 10 \\ x_1 + 4x_2 - x_4 + x_5 = 2 \\ x_1, x_2, x_3, x_4, x_5 \geq 0 \end{cases}$$

Tabelul corespunzător va fi:

c _B	x _B	x _B	2	3	0
			x ₁	x ₂	x ₄
		-2M	-M-2	-4M-3	M
2	x ₁	0	-1	0	0
3	x ₂	0	0	-1	0
0	x ₃	10	3	1	0
0	x ₄	0	0	0	-1
-M	x ₅	2	1	4	-1

Aplicăm simplex primal și obținem cel mai negativ Δ_j corespunzător lui x_2 și θ_i minim corespunzător lui x_5 . În acest moment avem o soluție de bază a problemei inițiale și putem elimina variabila x_5 . Obținem:

c _B	x _B	x _B	2	-M	0
			x ₁	x ₅	x ₄
		$\frac{3}{2}$	$-\frac{5}{4}$	$M+\frac{3}{4}$	$-\frac{3}{4}$
2	x ₁	0	-1	0	0
3	x ₂	$\frac{1}{2}$	$\frac{1}{4}$	$\frac{1}{4}$	$-\frac{1}{4}$
0	x ₃	$\frac{19}{2}$	$\frac{11}{4}$	$-\frac{1}{4}$	$\frac{1}{4}$
0	x ₄	0	0	0	-1
-M	x ₅	0	0	-1	0

 \Leftrightarrow

c _B	x _B	x _B	2	0
			x ₁	x ₄
		$\frac{3}{2}$	$-\frac{5}{4}$	$-\frac{3}{4}$
2	x ₁	0	-1	0
3	x ₂	$\frac{1}{2}$	$\frac{1}{4}$	$-\frac{1}{4}$
0	x ₃	$\frac{19}{2}$	$\frac{11}{4}$	$\frac{1}{4}$
0	x ₄	0	0	-1

și în continuare:

c _B	x _B	x _B	3	0
			x ₂	x ₄
		4	5	-2
2	x ₁	2	4	-1
3	x ₂	0	-1	0
0	x ₃	4	-11	3
0	x ₄	0	0	-1

→

c _B	x _B	x _B	3	0
			x ₂	x ₃
		$\frac{20}{3}$	$-\frac{7}{3}$	$\frac{2}{3}$
2	x ₁	$\frac{10}{3}$	$\frac{1}{3}$	$\frac{1}{3}$
3	x ₂	0	-1	0
0	x ₃	0	0	-1
0	x ₄	$\frac{4}{3}$	$-\frac{11}{3}$	$\frac{1}{3}$

c _B	x _B	x _B	2	0
			x ₁	x ₃
		30	7	3
→ 2	x ₁	0	-1	0
3	x ₂	10	3	1
0	x ₃	0	0	-1
0	x ₄	38	11	4

Din rezolvarea de mai sus se observă că această metodă este mai eficientă doar dacă numărul variabilelor secundare este mai mic decât numărul variabilelor principale. Totuși, motivul principal pentru care a fost introdusă această variantă, este existența unor probleme în care, pe parcursul rezolvării, se adaugă foarte multe restricții (de exemplu rezolvarea problemelor în numere întregi), pentru o restricție în plus tabelul simplex mărindu-se cu o linie și o coloană (cum se va vedea la capitolul de reoptimizare) iar tabelul corespunzător formei secundare doar cu o linie.

3. Forma revizuită a algoritmului simplex

O problemă mare (de exemplu cu 1000 variabile și 100 restricții) necesită în algoritmul simplex introducerea, memorarea și lucrul cu un tabel cu 100.000 componente, adică un număr imens de date. Din acest motiv, și remarcând faptul că în algoritmul simplex doar o parte din acestea contribuie efectiv la luarea deciziilor, s-a construit un algoritm care să țină cont de observațiile de mai sus, numit "forma revizuită a algoritmului simplex", în care se lucrează cu minimumul necesar din datele tabelului simplex. Astfel, pentru testarea soluției actuale și trecerea eventuală la o nouă bază avem nevoie doar de următoarele elemente:

- (1) inversa bazei curente B^{-1}
- (2) componentele vectorului Δ pentru a face testul de optim și a găsi variabila care intră în bază (dacă soluția nu e optimă)
- (3) coloana a_k din $B^{-1}A$ corespunzătoare celui mai negativ Δ_j (dacă există strict negativi) pentru a face testul de optim infinit
- (4) soluția curentă pentru a găsi variabila care iese din bază (dacă mai e cazul)

Aceste elemente se așează într-un tabel de forma:

c _B	x _B	$x_B = B^{-1}b$	B^{-1}
		$f(x_B)$	$\pi = c_B^T \cdot B^{-1}$

numit tabelul simplex redus.

Operațiile ce se efectuează în cadrul algoritmului simplex revizuit sunt (presupunem că problema este la forma standard de maxim și deținem o soluție admisibilă de bază și inversa bazei corespunzătoare):

Pasul 1. Se calculează Δ_j corespunzători variabilelor secundare cu formula cunoscută:

$$\Delta_j = c_B^T \cdot B^{-1} \cdot P_j - c_j = \pi \cdot P_j - c_j$$

unde P_j este coloana corespunzătoare variabilei x_j din matricea inițială.

Pasul 2. Se analizează Δ_j calculați:

- dacă toți sunt mai mari sau egali cu 0 soluția curentă este optimă. STOP
- dacă există Δ_j strict negativi se alege minimul dintre aceștia Δ_k , variabila x_k va intra în bază și se trece la pasul 3.

Pasul 3. Se calculează $a_k = B^{-1}P_k$ care se atașează ca nouă coloană la tabel:

c_B	x_B	$x_B = B^{-1}b$	B^{-1}	$a_k = B^{-1}P_k$
		$f(x_B)$	$\pi = c_B^T \cdot B^{-1}$	Δ_k

Pasul 4. Se analizează componentele coloanei a_k :

- dacă toate sunt mai mici sau egale cu 0 problema are optim infinit. STOP
- dacă există a_{ik} strict pozitivi se alege cel pentru care se obține:

$$a_{rk} = \min_{\substack{1 \leq i \leq m \\ a_{ik} > 0}} \frac{x_i}{a_{ik}}$$

și variabila x_r va ieși din bază apoi se trece la pasul 5.

Pasul 5. Se pivotază întregul tabel simplex și se elimină ultima coloană.

Pasul 6. Se reia algoritmul de la pasul 2.

Exemplu Fie problema de programare liniară rezolvată mai sus:

$$(\max) f = 2x_1 + 3x_2$$

$$\begin{cases} 3x_1 + x_2 \leq 10 \\ x_1 + 4x_2 \geq 2 \\ x_1, x_2 \geq 0 \end{cases}$$

Forma standard a problemei va fi:

$$(\max) f = 2x_1 + 3x_2$$

$$\begin{cases} 3x_1 + x_2 + x_3 = 10 \\ x_1 + 4x_2 - x_4 = 2 \\ x_1, x_2, x_3, x_4 \geq 0 \end{cases}$$

(max) $f = 2x_1 + 3x_2 - Mx_5$

iar după introducerea variabilelor artificiale obținem:

$$\begin{cases} 3x_1 + x_2 + x_3 = 10 \\ x_1 + 4x_2 - x_4 + x_5 = 2 \\ x_1, x_2, x_3, x_4, x_5 \geq 0 \end{cases}$$

Iterația 1. Prima bază este $B = I_2 = B^{-1}$ corespunzătoare variabilelor x_3 și x_5 de unde rezultă $x_B = \begin{pmatrix} 10 \\ 2 \end{pmatrix}$ și $\pi = (0, -M) \cdot I_2 = (0, -M)$. Primul tabel redus va fi:

0	x_3	10	1	0
-M	x_5	2	0	1
		-2M	0	-M

Se calculează $\Delta_S = \pi \cdot S - c_S = (0, -M) \cdot \begin{pmatrix} 3 & 1 & 0 \\ 1 & 4 & -1 \end{pmatrix} - (2 \quad 3 \quad 0) = (-M-2 \quad -4M-3 \quad M)$

de unde rezultă că cel mai negativ este Δ_2 și vom calcula coloana $a_2 = I_2 \cdot \begin{pmatrix} 1 \\ 4 \end{pmatrix} = \begin{pmatrix} 1 \\ 4 \end{pmatrix}$ care se adaugă la tabelul anterior rezultând:

0	x_3	10	1	0	1
-M	x_5	2	0	1	4
		-2M	0	-M	-4M-3

θ_i minim este cel corespunzător lui x_5 și deci variabila x_2 va intra în locul variabilei x_5 . După pivotare și eliminarea ultimei coloane obținem:

0	x_3	$\frac{19}{2}$	1	$-\frac{1}{4}$
3	x_2	$\frac{1}{2}$	0	$\frac{1}{4}$
		$\frac{3}{2}$	0	$\frac{3}{4}$

Deoarece variabila artificială x_5 a ieșit din bază ea va fi eliminată din problemă.

Iterația 2. $\Delta_S = \pi \cdot S - c_S = (0, \frac{3}{4}) \cdot \begin{pmatrix} 3 & 0 \\ 1 & -1 \end{pmatrix} - (2 \quad 0) = (-\frac{5}{4} \quad -\frac{3}{4})$

cel mai negativ este Δ_1 și vom calcula coloana $a_1 = \begin{pmatrix} 1 & -\frac{1}{4} \\ 0 & \frac{1}{4} \end{pmatrix} \cdot \begin{pmatrix} 3 \\ 1 \end{pmatrix} = \begin{pmatrix} \frac{11}{4} \\ \frac{1}{4} \end{pmatrix}$ care se

adaugă la tabelul anterior rezultând:

0	x_3	$\frac{19}{2}$	1	$-\frac{1}{4}$	$\frac{11}{4}$
3	x_2	$\frac{1}{2}$	0	$\frac{1}{4}$	$\frac{1}{4}$
		$\frac{3}{2}$	0	$\frac{3}{4}$	$-\frac{5}{4}$

θ_i minim este cel corespunzător lui x_2 și deci variabila x_1 va intra în locul variabilei x_2 . După pivotare și eliminarea ultimei coloane obținem:

0	x_3	4	1	-3
2	x_1	2	0	1
		4	0	2

Iterația 3. $\Delta_S = \pi \cdot S - c_S = (0,2) \cdot \begin{pmatrix} 1 & 0 \\ 4 & -1 \end{pmatrix} - (3 \ 0) = (8 \ -2)$

cel mai negativ este Δ_4 și vom calcula coloana $a_4 = \begin{pmatrix} 1 & -3 \\ 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} 0 \\ -1 \end{pmatrix} = \begin{pmatrix} 3 \\ -1 \end{pmatrix}$ care se adaugă la tabelul anterior rezultând:

0	x_3	4	1	-3	3
2	x_1	2	0	1	-1
		4	0	2	-2

θ_i minim este cel corespunzător lui x_3 și deci variabila x_4 va intra în locul variabilei x_3 . După pivotare și eliminarea ultimei coloane obținem:

0	x_4	$\frac{4}{3}$	$\frac{1}{3}$	-1
2	x_1	$\frac{10}{3}$	$\frac{1}{3}$	0
		$\frac{20}{3}$	$\frac{2}{3}$	0

Iterația 4. $\Delta_S = \pi \cdot S - c_S = (\frac{2}{3}, 0) \cdot \begin{pmatrix} 1 & 1 \\ 4 & 0 \end{pmatrix} - (3 \ 0) = (-\frac{7}{3} \ 0)$

cel mai negativ este Δ_2 și vom calcula coloana $a_2 = \begin{pmatrix} \frac{1}{3} & -1 \\ \frac{1}{3} & 0 \end{pmatrix} \cdot \begin{pmatrix} 1 \\ 4 \end{pmatrix} = \begin{pmatrix} -\frac{11}{3} \\ \frac{1}{3} \end{pmatrix}$ care se adaugă la tabelul anterior rezultând:

0	x_4	$\frac{4}{3}$	$\frac{1}{3}$	-1	$-\frac{11}{3}$
2	x_1	$\frac{10}{3}$	$\frac{1}{3}$	0	$\frac{1}{3}$
		$\frac{20}{3}$	$\frac{2}{3}$	0	$-\frac{7}{3}$

θ_i minim este cel corespunzător lui x_1 și deci variabila x_2 va intra în locul variabilei x_1 . După pivotare și eliminarea ultimei coloane obținem:

0	x_4	38	4	-1
3	x_2	10	1	0
		30	3	0

Iterația 5. $\Delta_S = \pi \cdot S - c_S = (3,0) \cdot \begin{pmatrix} 3 & 1 \\ 1 & 0 \end{pmatrix} - (2 \ 0) = (7 \ 3) > 0$ deci soluția este optimă (și unică)

STOP

Chiar dacă la prima vedere calculele par mai împrăștiate și deci mai lente, pentru probleme mari, pe calculator se economisește foarte multă memorie și se mărește foarte mult viteza de calcul.

Totuși, marele avantaj al acestei metode este acela că, la fiecare iterație, se folosesc datele problemei inițiale, ceea ce face ca erorile inerente de rotunjire să nu se propage, rămânând în limite acceptabile.