MASARYKOVA UNIVERZITA V BRNĚ
PEDAGOGICKÁ FAKULTA

KATEDRA MATEMATIKY

Lineární programování
Diplomová práce

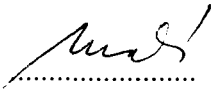
Brno 2005

Vedoucí diplomové práce:
PhDr. Jiřina Novotná, Ph.D.

Vypracovala:
Hana Malá

Prohlašuji, že jsem diplomovou práci zpracoval/a samostatně a použil/a jen prameny uvedené v seznamu literatury.

Souhlasím, aby práce byla uložena na Masarykově univerzitě v Brně v knihovně Pedagogické fakulty a zpřístupněna ke studijním účelům.



podpis

Poděkování

Děkuji touto cestou PhDr. Jiřině Novotné, Ph.D. za zpřístupnění literatury a v neposlední řadě za cenné rady a konzultace při psaní diplomové práce.

Obsah

Úvod

Lineární programování je aplikovaná matematická disciplína, kterou lze využít v oblastech vědy, techniky,školství, financí i ekonomie. Cílem diplomové práce je vysvětlit a ukázat na příkladech úlohy lineárního programování v teoretické rovině i v praxi. Teorie lineárního programování obsahuje vícero metod a algoritmů pro nalezení optimálního řešení. Uvedené způsoby optimalizace řešení jsou výhodné hlavně pro jeho jednoduchost při řešení. Při praktických úlohách se setkáváme s velkým množstvím omezení a parametrů, a proto základním úkolem je matematická formulace úlohy. Dané problémy řeší matematická disciplina nazývaná matematické programování. Jeho cílem je určit a vytvořit výpočetní postupy, které řeší zformulované úlohy. Částí matematického programování je lineární programování. V lineárním programování jsou vztahy mezi proměnnými lineární. Ucelený a jasný náhled dávají čtenáři i elementární poznatky z lineární algebry a matematické analýzy, následuje problematika lineárního programování, ke které jsou vyřešeny příklady jak graficky tak pomocí metod lineárního programování. V závěru jsou popsány některé z počítačových systémů, používaných k výpočtům úloh lineárního programování.

V první kapitole je velmi stručně popsána hlavní myšlenka problému lineárního programování. Druhá kapitola je věnovaná základním matematickým větám a definicím, které jsou nevyhnutelné pro pochopení problematiky lineárního programování. Její podkapitoly tvoří základní věty a definice vektorových prostorů, matic, konvexních množin, lineárních rovnic, lineárních nerovnic a extrémů funkcí. Jsou zde uvedeny i odkazy na důkazy v seznamu literatury.

Třetí kapitola obeznamuje s teorií lineárního programování, s jejími počátky v první podkapitole s názvem Historie lineárního programování. Dále je popsaná formulace obecné úlohy lineárního programování, s následným postupem jejího řešení. Jako nejznámější a nejpoužívanější metoda je vybraná simplexová metoda pro výpočet úloh lineárního programování a její algoritmus.

Grafickému řešení úloh lineárního programování v prostoru dimenze dva je věnovaná čtvrtá kapitola. Na konkrétních příkladech je znázorněna jednoduchost řešení úloh, avšak pouze v případě dvou parametrů, z důvodu použitelnosti na základní škole.

Pro lepší představu a pochopení jsou v kapitole číslo pět stručně charakterizovány speciální úlohy lineárního programování. Na příkladě dopravní úlohy jsou uvedeny

postupy a metody pro řešení klasické úlohy lineárního programování, jenž se v praxi vyskytuje velmi často.

Při rozvoji počítačů vznikaly i systémy pro řešení různých úloh lineárního i nelineárního programování. Úlohy o více proměnnými se řeší graficky obtížněji, proto byl vyvinut software k řešení úloh lineárního programování o velkém počtu proměnných. Tato problematika je naznačena v šesté kapitole, kde je vyřešen příklad pomocí počítačového systému Matlab i ukázka řešení úlohy programem LPS. Uvedený software je v obou případech veřejně dostupný na internetu, rovněž některé ukázky řešených úloh lineárního programování. V příloze jsou připojeny obrázky a grafy vyřešeného příkladu systémem Matlab.

1. ÚVOD K LINEÁRNÍMU PROGRAMOVÁNÍ

Lineární programování umožňuje vybudovat teorii a vypracovat výpočetní postupy, které nám slouží k určení úloh s následující formulací:

Najít absolutní extrém lineární funkce (minimum nebo maximum)

$$z = f_0(x_1,x_2,\ldots,x_n)$$

za omezujících podmínek:

$f_i(x_1,x_2,\ldots,x_n) = b_i \quad (i = 1,2,\ldots,l)$

$f_j(x_1,x_2,\ldots,x_n) \leq b_j \quad (j = 1,2,\ldots,p)$

$f_k(x_1,x_2,\ldots,x_n) \geq b_k \quad (k = p+1,\ldots,m)$

$x_j \geq 0 \qquad (j = 1,2,\ldots,n)$

kde f_i $i \in \{0,1,...,k\}$ jsou lineární funkce.

Formulace úlohy je složena ze soustavy rovnic a nerovnic, které nám vyjadřují podmínky úlohy, a z lineární funkce, udávající hledaný cíl. Funkce, jejíž extrém hledáme, se nazývá účelová funkce. Později ukážeme, že nerovnice lze vždy převést na rovnice o větším počtu proměnných.

Poznámka: Nebudeme rozlišovat mezi bodem X prostoru V a jeho polohovým vektorem **x**.

Úloha lineárního programování, ve které se v omezujících podmínkách vyskytují pouze rovnice, se nazývá úloha v kanonickém tvaru, který můžeme zapsat:

Najít absolutní extrém lineární účelové funkce

$$f(\mathbf{x}) = c_1x_1 + \ldots + c_nx_n$$

Za podmínek:

$a_{11}x_1 + a_{12}x_2 + \ldots + a_{1j}x_j + \ldots + a_{1n}x_n = b_1$

$a_{21}x_2 + a_{22}x_2 + \ldots + a_{2j}x_j + \ldots + a_{2n}x_n = b_2$

.....

$a_{i1}x_1 + a_{i2}x_2 + \ldots + a_{ij}x_j + \ldots + a_{in}x_n = b_i$

.....

$a_{m1}x_1 + a_{m2}x_2 + \ldots + a_{mj}x_j + \ldots + a_{mn}x_n = b_m$

$xi \geq 0$

$j = 1,2,\ldots,n \quad i = 1,2,\ldots,m$

$m < n$, a_{ij}, b_i jsou konstanty.

Veličiny c_j se nazývají koeficienty lineární účelové funkce.

Při úlohách lineárního programování jde v podstatě o řešení soustavy lineárních rovnic a nerovnic.

K matematické formulaci vede velké množství především ekonomických problémů. Uvedeme zde zjednodušenou podobu jednoho z ekonomických problémů.

Příklad 1.1 :

Dopravní problém patří k nejjednodušším problémům lineárního programování. Cílem je minimalizace dopravních nákladů při přepravě zboží ze skladů ke spotřebitelům. Přepravce má dopravit určitý počet zboží z různých skladů do různých prodejen. Každý sklad má omezené množství zboží a každá prodejna požaduje jen určité zboží.

Označení:

a_1 – celkové množství zboží v prvním skladu

a_2 - celkové množství zboží v druhém skladu

b_1 - celkové množství zboží požadované první prodejnou

b_2 - celkové množství zboží požadované druhou prodejnou

b_3 - celkové množství zboží požadované třetí prodejnou

c_{ij} – přepravní náklady na rozvoz i - tého skladu do j - té prodejny

x_{ij} – množství přepravovaného zboží z i - tého skladu do j - té prodejny

Údaje dopravního problému uspořádáme do tabulky:

Sklad \ prodejna	1	2	3	
1	x_{11}	x_{12}	x_{13}	a_1
2	x_{21}	x_{22}	x_{23}	a_2
	b_1	b_2	b_3	

Jde nám o minimalizaci lineární účelové funkce:

$f(\mathbf{x}) = c_{11}x_{11} + c_{12}x_{12} + c_{13}x_{13} + c_{21}x_{21} + c_{22}x_{22} + c_{23}x_{23}$

Z prvního skladu se přepraví $x_{11} + x_{12} + x_{13} = a_1$ zboží, z druhého skladu se přepraví $x_{21} + x_{22} + x_{23} = a_2$ zboží. První prodejna požaduje $x_{11} + x_{21} = b_1$ zboží, druhá $x_{12} + x_{22} = b_2$ zboží, třetí $x_{13} + x_{23} = b_3$ zboží. Platí zde podmínky nezápornosti $x_{ij} \geq 0$, pro všechna i, j . Předpokládáme, že zásoby ve skladech se rovnají množství, které požadují prodejny, tedy : $\Sigma_i a_i = \Sigma_j b_j$

Dopravní úlohu můžeme formulovat takto:

Najděte minimum lineární účelové funkce, která vyjadřuje náklady

$f(\mathbf{x}) = c_{11}x_{11} + c_{12}x_{12} + c_{13}x_{13} + c_{21}x_{21} + c_{22}x_{22} + c_{23}x_{23}$

za podmínek:

$x_{ij} \geq 0$ pro $i \in \{1,2\}, j = \{1,2,3\}$

$$\begin{aligned} x_{11} + x_{12} + x_{13} &= a_1 \\ x_{21} + x_{22} + x_{23} &= a_2 \\ x_{11} + x_{21} &= b_1 \\ x_{13} + x_{23} &= b_3 \end{aligned}$$

Další úlohy, které lze řešit pomocí lineárního programování jsou např. úloha rozřezání polotovaru na menší díly, aby odpad byl minimální, nebo úloha plánování výroby za účelem dosažení co největší ceny odbytu. Jejich řešení se budeme věnovat v kapitole pět, kde jsou řešeny pomocí počítačových systémů.

2. MATEMATICKÝ ZÁKLAD

Základní množiny, se kterýma pracujeme, značíme takto: N ... přirozená čísla, R ... reální čísla, C... komplexní čísla, M ...množina matic různých typů, E_n ....n – rozměrný Euklidovský prostor, S ...množina bodů.

2.1 Vektorové prostory

Buď G množina, $\circ$ operace na G, splňující podmínky:

1) Pro a, b patřící do G je $a \circ b$ také v G (tzv. Operace je uzavřená.)
2) Pro a, b, c patřící do G je $(a \circ b) \circ c = a \circ (b \circ c)$ (Asociativní zákon)
3) Existuje e patřící do G, že pro každé $a \in G$ platí $e \circ a = a \circ e = a$
   (e je tzv. neutrální prvek.)
4) Pro každé $a \in G$, existuje $b \in G$ tak, že $a \circ b = b \circ a = e$
   (a,b jsou navzájem inverzní.)

Pak se $(G, \circ)$ nazývá Grupa a G je její nosná množina, $\circ$ (grupná) operace.

Grupa $(G, \circ)$ se nazývá komutativní neboli abelovská, jestliže navíc platí:

5) Pro a, b patřící do G je $a \circ b = b \circ a$.

(Příklady grup: $(R, +)$, $(R - \{0\}, \cdot)$, $(Z, +)$, $(M_n, +)$, $(RegM, \cdot)$,$(\{e\}, \circ)$ ... tzv. triviální grupa.)

Definice 2.1

Buď $(V, +)$ abelovská grupa, jejíž prvky nazýváme vektory a nechť ke každému α patřící do množiny R a $\boldsymbol{u}$ patřící do množiny V existuje prvek $a \cdot \boldsymbol{u} \in V$ tak, že pro všechna $\alpha, \beta \in R$ a $\boldsymbol{u}, \boldsymbol{v} \in V$ platí:

$\alpha \cdot (\boldsymbol{u} + \boldsymbol{v}) = (\alpha \cdot \boldsymbol{u}) + (\alpha \cdot \boldsymbol{v})$ (analogie distributivního zákona)

$(\alpha + \beta) \cdot \boldsymbol{u} = (\alpha \cdot \boldsymbol{u}) + (\beta \cdot \boldsymbol{u})$

(Pokud uvažujeme, že $\cdot$ má přednost před + lze vynechat závorky)

$\alpha \cdot (\beta \cdot \boldsymbol{u}) = (\alpha \cdot \beta) \cdot \boldsymbol{u}$

$1 \cdot \boldsymbol{u} = \boldsymbol{u}$ (Vlastně: α,ß...skaláry; ***u,v***...vektory)

Pak $(V, +, \cdot)$ se nazývá vektorový prostor nad tělesem R. Neutrální prvek grupy $(V, +)$ se značí $\boldsymbol{o}$ nebo $\boldsymbol{0}$ a nazývá se nulovým vektorem.

Důkazy uvedených a vět je možné nalézt v literatuře [1], [6].

Věta 2.1

Je-li $(V, +, \cdot)$ vektorový prostor, pak:

a) $0 \cdot \boldsymbol{u} = o$ pro každý vektor *u patřící do V* (o....nulový vektor)

b) $\alpha \cdot o = o$ pro každý skalár α *patřící do R*

c) Jestliže $\alpha \cdot \boldsymbol{u} = o$, pak buď $\alpha = 0$ nebo $\boldsymbol{u} = o$.

d) $(-1) \cdot \boldsymbol{u} = -\boldsymbol{u}$ pro každé vektor *u patřící do V*

Definice 2.2

Nechť V je vektorový prostor s operacemi +, · a W je podmnožina V. Jestliže $(W, +, \cdot)$ je vektorový prostor, nazývá se vektorovým podprostorem prostoru V.

Definice 2.3

Nechť V je vektorový prostor, vektory $\boldsymbol{v_1}, \boldsymbol{v_2}, \ldots \boldsymbol{v_k}$ patřící do V se nazývají lineárně nezávislé, jestliže pro každé $\alpha_1, \alpha_2, \ldots \alpha_k$ platí:

$$\alpha_1 \cdot \boldsymbol{v_1} + \alpha_2 \cdot \boldsymbol{v_2} + \ldots + \alpha_k \cdot \boldsymbol{v_k} = o \Rightarrow \alpha_1 = \alpha_2 = \ldots = \alpha_k = 0.$$

Je-li $\boldsymbol{u_1}, \ldots, \boldsymbol{u_k}$ skupina lineárně nezávislých prvků vektorového prostoru V, pak množina všech lineárních kombinací těchto prvků je podprostorem V. Říkáme, že prvky $\boldsymbol{u_1}, \ldots, \boldsymbol{u_k}$ tento podprostor generují.

Maximální lineárně nezávislý systém vektorů ve vektorovém prostoru V se nazývá báze. Počet prvků v báze se nazývá dimenze prostoru V a značí se dim V.

Je-li $x = x_1 \cdot \boldsymbol{e_1} + x_2 \cdot \boldsymbol{e_2} + \ldots + x_n \cdot \boldsymbol{e_n}$. Čísla $x_1, \ldots x_n$ se nazývají souřadnice vektoru $\boldsymbol{x}$ v bázi $\boldsymbol{e_1}, \boldsymbol{e_2}, \ldots \boldsymbol{e_n}$.

Věta 2.2

Buď V vektorový prostor nad R. Pak skalárním součinem na V

nazýváme funkci $(\cdot, \cdot) : V \times V \rightarrow R$

a) $(\boldsymbol{u}, \boldsymbol{u}) > 0$ pro u patřící do V, $\boldsymbol{u} \neq \boldsymbol{o}$

b) $(\boldsymbol{u}, \boldsymbol{v}) = (\boldsymbol{v}, \boldsymbol{u})$, resp. $(\boldsymbol{u}, \boldsymbol{v}) = (\boldsymbol{v}, \boldsymbol{u})^*$ nad C pro libovolné $\boldsymbol{u},\boldsymbol{v}$ patřící do V

c) $(\boldsymbol{u}+\boldsymbol{v}, \boldsymbol{w}) = (\boldsymbol{u}, \boldsymbol{w}) + (\boldsymbol{v}, \boldsymbol{w})$ pro každé $\boldsymbol{u},\boldsymbol{v},\boldsymbol{w}$ patřící do V

d) $(\alpha \cdot \boldsymbol{u}, \boldsymbol{v}) = \alpha \cdot (\boldsymbol{u}, \boldsymbol{v})$ pro libovolné $\boldsymbol{u},\boldsymbol{v}$ patřící do V a skalár α patřící do R(resp. C)

2.2 Matice

Nyní budeme definovat základní pojmy z teorie matic.

Maticí typu m/n rozumíme skupinu m × n komplexních čísel uspořádaných do m řádků a n sloupců (m, n ∈ N). Tato čísla nazýváme prvky matice. Označíme-li a_{ij} prvek v i-tém řádku a j-tém sloupci, pak matici typu m/n můžeme zapsat zkráceně $\mathbf{A} = [a_{ij}]$. Buď **A** matice typu m/n. Posloupnost $a_{11}, a_{22}, ..., a_{mm}$ se nazývá *hlavní diagonála.* Prvky této posloupnosti se nazývají *diagonální.*
Nechť **A,B** jsou matice téhož typu m/n. Potom klademe:
$\mathbf{A} = \mathbf{B}$, právě tehdy když $a_{ik} = b_{ik}$, pro všechna i = 1, 2, ...,m, k = 1, 2, ..., n.

Typy matic :

1. Matice typu m/1 se nazývá *sloupcová.*
2. Matice typu 1/n se nazývá *řádková.*
3. Je-li m = n, nazýváme matici *čtvercovou* maticí řádu n.
4. Čtvercová matice **A** se nazývá jednotková, když $a_{ik} = 0$ pro všechna $i \neq k$ a $a_{ii} = 1$, i = 1, 2, ..., n, tj. na hlavní diagonále má jedničky, jinde nuly. Označujeme ji **E.**
5. Čtvercová matice **A** řádu n se nazývá *diagonální*, jestliže má na hlavní diagonále alespoň jeden prvek nenulový a ostatní prvky jsou rovny nule.
6. *Transponovaná* matice $\mathbf{A}^T$ k matici **A** vznikne z matice **A** tak, že vyměníme řádky matice za sloupce a opačně. Jestliže matice **A** je typu m/n, potom matice transponovaná je typu n/m. Platí $(\mathbf{A}^T)^T = \mathbf{A}$.
7. Matice **A** se nazývá *dolní* (resp. horní) *trojúhelníková* matice, jestliže $a_{ik} = 0$ pro $i < k$ (resp. $i > k$).
8. Matici **D**, kde $a_{ik} = 0$ pro všechna i = 1, 2, ...,m, k = 1, 2, ..., n, nazýváme *nulovou* maticí.

9. Matice **A** se nazývá *symetrická*, jestliže $\mathbf{A} = \mathbf{A}^T$ (tj. $a_{ik} = a_{ki}$ pro i, k = 1, 2, ..., n).

10. Matice **A** se nazývá *antisymetrická*, jestliže $\mathbf{A}^T = -\mathbf{A}$ (tj. $a_{ik} = -a_{ki}$ pro i, k = 1,2,..., n).

Matice **A**, **B** matice stejného typu m/n. Součet **A** + **B** je taková matice $\mathbf{C} = [c_{ik}]$, typu m/n kde platí: $c_{ik} = a_{ik} + b_{ik}$.

Matice $\mathbf{A} = [a_{ik}]$ typu m/n, c je konstanta. Pak platí: c.**A** je taková matice $\mathbf{B} = [b_{ik}]$ kde $b_{ik} = c.\ a_{ik}$.

Vlastnosti součtu a násobku matice:

$\mathbf{A},\mathbf{B} \in M \Rightarrow (\mathbf{A} + \mathbf{B}) \in M$

$\mathbf{A} + \mathbf{B} = \mathbf{B} + \mathbf{A}$

$(\mathbf{A} + \mathbf{B}) + \mathbf{C} = \mathbf{A} + (\mathbf{B} + \mathbf{C})$

$\mathbf{A} + \mathbf{0} = \mathbf{A}$

$\mathbf{A} + (-\mathbf{A}) = \mathbf{0}$

$k \in R, \mathbf{A} \in M \Rightarrow k.\mathbf{A} \in M$

$k = 1 \Rightarrow 1.\mathbf{A} = \mathbf{A}$

$k_1(k_2\mathbf{A}) = (k_1.\ k_2)\mathbf{A}$

$(k_1 + k_2)\mathbf{A} = k_1\mathbf{A} + k_2\mathbf{A}$

Důkazy uvedených vět je možné nalézt v literatuře [1].

Definice 2.4

Násobení matic:

Pro matici A typu m/n a matici **B** typu n/p klademe:

$\mathbf{A}\cdot\mathbf{B} = \mathbf{C}$ typu m/n, přičemž

$$c_{ik} = \sum_{j=1}^{n} a_{ij}.b_{jk}$$

Tedy prvek c_{ij} dostaneme jako skalární součin i-tého řádku matice **A** a k-tého sloupce matice **B**.

Součinu **A.B** říkáme, že matice **A** je násobena maticí **B** zprava nebo že matice **B** je násobena maticí **A** zleva. Obecně tedy neplatí $\mathbf{A.B} \neq \mathbf{B.A}$. Násobení matic je nekomutativní operací.

Vlastnosti součinu:

$\mathbf{A.E} = \mathbf{E.A} = \mathbf{A}$

$\mathbf{A.(B + C)} = \mathbf{A.B + A.C}$ $\quad$ $(\mathbf{(A + B).C} = \mathbf{A.C + B.C})$

$(\mathbf{A.B^T}) = \mathbf{B^T.A^T}$

Maximální počet lineárně nezávislých řádků matice **A** typu m/n nazveme hodností této matice. Označujeme $r(\mathbf{A})$, $h(\mathbf{A})$.
Hodnost nulové matice je 0.
Řádkovými elementárními transformacemi (úpravami) matice, kterými se nemění hodnost matice A, nazýváme tyto úpravy:

- Výměna dvou řádků matice.
- Vynásobení nějakého řádku matice číslem různým od nuly.
- Přičtením c-násobku jednoho řádku matice k jinému řádku.

Podobně definujeme sloupcové elementární transformace.
Řekneme, že matice **A,B** jsou ekvivalentní, lze-li jednu z nich převést v druhou konečným počtem elementárních transformací. Označujeme $\mathbf{A} \sim \mathbf{B}$.

Definice 2.5
Čtvercovou matici **A** nazveme *regulární*, je – li $|\mathbf{A}| \neq 0$, (nebo-li hodnost je rovna řádu matice). Pokud $|\mathbf{A}| = 0$, potom matici **A** nazveme *singulární*.

Definice 2.6
Čtvercovou matici **X** n-tého řádu nazveme maticí *inverzní* ke čtvercové matici **A** n-tého řádu, jestliže platí: $\mathbf{A} \cdot \mathbf{A^{-1}} = \mathbf{A^{-1}} \cdot \mathbf{A} = \mathbf{E}$. (Kde **E** je jednotková matice)

2.3 Konvexní množiny

Množina bodů v rovině či prostoru se nazývá *konvexní* množina, obsahuje-li s každými svými dvěma body i body úsečky, která je spojuje.

Definice 2.7

Nechť $n \in N$. Konvexní kombinací bodů $A_1, A_2, \ldots, A_n \in E_n$ nazveme bod $A_n \in E_n$, pro který platí $A = a_1A_1 + a_2A_2 + \ldots + a_nA_n$, kde $\sum_{i=1}^{n} a_i = 1$, $a_i \geq 0$.

Definice 2.8

Podmnožina $C \in E_n$ se nazývá konvexní, jestliže každá konvexní kombinace libovolné dvojice bodů $X, Y \in C$ je opět prvkem C.

Věta 2.3

Průnik konvexních množin je konvexní množina (Obr. 2.3)

Definice 2.9

Bod A konvexní množiny se nazývá krajním (vrcholem množiny), jestliže jej nelze vyjádřit jako konvexní kombinaci nějakých dvou bodů patřících do konvexní množiny, tj. například bod K je vrcholem konvexní množiny C, nelze-li jej vyjádřit ve tvaru

$$K = tV + (1 - t)W, \quad \text{kde } V, W \in C,\ V \neq W,\ 0 < t < 1.$$

Definice 2.10

Jestliže množina C prostoru E_n je libovolná, pak průnik všech konvexních množin obsahujících množinu C nazýváme *konvexní obal* množiny C, označujeme K(C).

Například konvexním obalem dvou bodů je jejich spojnice a konvexním obalem tří bodů, které neleží v jedné přímce, je trojúhelník.

Definice 2.11

Neprázdná ohraničená konvexní množina bodů prostoru E_n, která má konečný počet krajních bodů (vrcholů), se nazývá *konvexní polyedr* prostoru E_n.

Platí, že všechny konvexní kombinace konečného počtu bodů tvoří konvexní polyedr.

Definice 2.12

Simplexem v E_n nazveme každý konvexní mnohostěn (polyedr), který má právě n+1 vrcholů.

Poznámky:

Konvexní polyedr je konvexní obal konečné množiny.

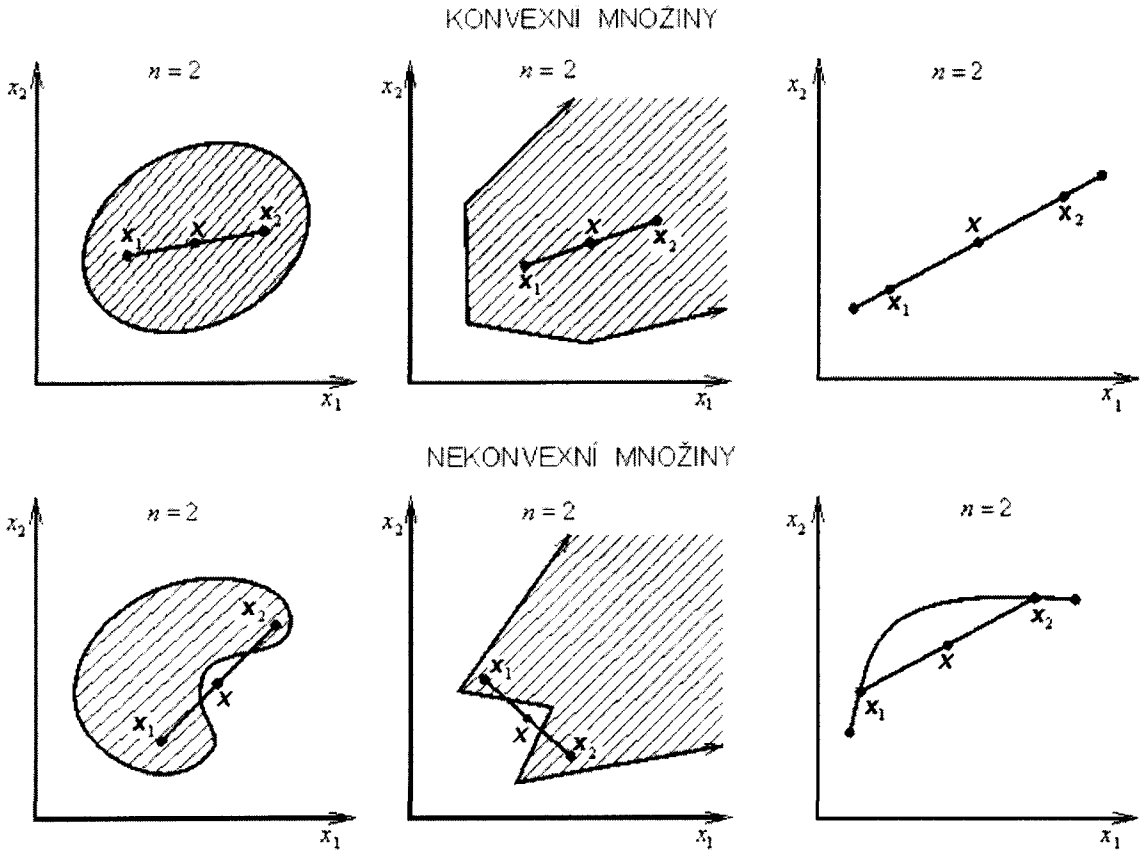
Konvexní polyedrická množina se definuje jako neprázdný průnik konečně mnoha uzavřených poloprostorů v E_n.

Průnik konvexní polyedrické množiny M s některou určujících nadrovin se nazývá *stěna množiny* M.

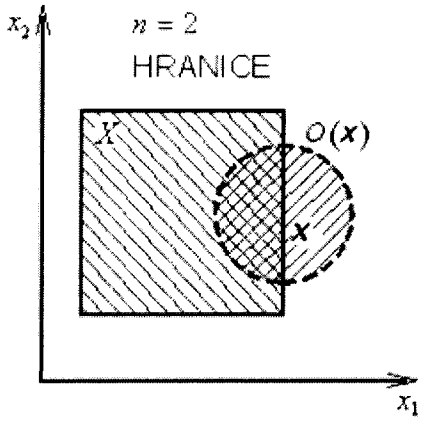
Stěna konvexní polyedrické množiny, která je přímkou se nazývá *hrana.*

Stěna konvexní polyedrické množiny, která je bodem se nazývá *vrchol.*

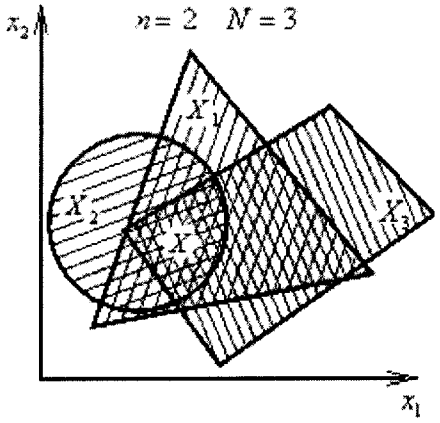
Obrázek 2.1 :



Obrázek 2.2 :



Obrázek 2.3 : Průnik konvexních množin.



2.4 Extrémy funkcí

Definice 2.13

Extrémy funkce:

Maximum: Pro všechna $x \in D(f)$ platí že $f(x) < f(c)$ pro $x \neq c$.

V bodě c nabývá funkce svého *maxima.*

Minimum: Pro všechna $x \in D(f)$ platí že $f(x) > f(c)$ pro $x \neq c$.

V bodě c nabývá funkce svého *minima.*

2.5 Lineární rovnice

$$
\begin{aligned}
&a_{11}x_1 + a_{12}x_2 + \ldots + a_{1j}x_j + \ldots + a_{1n}x_n = b_1 \\
&a_{21}x_2 + a_{22}x_2 + \ldots + a_{2j}x_j + \ldots + a_{2n}x_n = b_2 \\
&\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \qquad (1) \\
&a_{i1}x_1 + a_{i2}x_2 + \ldots + a_{ij}x_j + \ldots + a_{in}x_n = b_i \\
&\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \\
&a_{m1}x_1 + a_{m2}x_2 + \ldots + a_{mj}x_j + \ldots + a_{mn}x_n = b_m
\end{aligned}
$$

(1) se nazývá soustavou *m* lineárních rovnic o *n* neznámých. Čísla a_{ij} nazýváme *koeficienty* soustavy, čísla $x_1, x_2, \ldots, x_n$ nazýváme *neznámé* a čísla b_i jsou pravou stranou dané soustavy.

Soustavu nazýváme *homogenní*, jestliže pro všechny b_i platí $b_i = 0$. Řešením soustavy *m* lineárních rovnic o *n* neznámých je každá soustava n čísel $x_1, x_2, \ldots, x_n$, která splňuje současně všech *m* rovnic.

Tuto soustavu můžeme také napsat v maticovém tvaru: $\mathbf{Ax} = \mathbf{b}$,

kde $\mathbf{A} = (\mathbf{a}_1, \mathbf{a}_2, \ldots, \mathbf{a}_n)$, $\mathbf{x} = (\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n)^T$, $\mathbf{b} = (\mathbf{b}_1, \mathbf{b}_2, \ldots, \mathbf{b}_m)^T$.

Řešit soustavu znamená určit všechna řešení této soustavy, přitom soustava buď nemá žádné řešení nebo má právě jedno řešení nebo má nekonečně mnoho řešení.

Důkazy uvedených vět nalezneme v literatuře [1].

Věta 2.4 (Frobeniova věta)

Soustava lineárních rovnic je řešitelná právě když hodnost matice této soustavy je rovna hodnosti rozšířené matice této soustavy.

Při simplexové metodě se využívá Jordanovy metody řešení soustavy lineárních rovnic, proto tuto metodu připomeneme podrobněji.

Jordanova metoda

1) Soustavě rovnic (1) přiřadíme rozšířenou matici soustavy

$$
\left(\begin{array}{l}
a_{11}a_{12} \ \ldots\ldots \ a_{1n} \ / \ b_1 \\
\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \\
a_{m1}a_{m2} \ \ldots\ldots \ a_{mn} \ / \ b_m
\end{array}\right)
$$

kde $A = \begin{pmatrix} a_{11}a_{12} \ \ldots \ldots \ a_{1n} \\ \ldots\ldots\ldots\ldots\ldots\ldots \\ a_{m1}a_{m2} \ \ldots \ldots \ a_{mn} \end{pmatrix}$

se nazývá matice soustavy.

2) Užitím elementárních úprav upravujeme matici soustavy na jednotkovou matici respektive na jednotkovou submatici. Úpravy provádíme s celými řádky rozšířené matice.
3) a) na jednotkovou matici – řešení jediné
   b) na jednotkovou submatici – nekonečně mnoho řešení.

V ekonomických aplikacích se vyjadřuje tzv. základní nebo-li bázické řešení, které obdržíme tak, že volitelné neznámé zvolíme nuly. Ostatní proměnné se nazývají bázické proměnné. Jordanova metoda umožňuje přechod od jednoho základního řešení k jinému změnou jedné bázické proměnné.

Příklad 2.1 :

Příklad řešení soustavy lineárních rovnic, která má jediné řešení.

$x_1 + x_2 - 2x_3 = 5$

$2x_1 - 2x_2 + x_3 = 7$

$x_1 - 3x_2 + 2x_3 = 1$

Matice má tvar:

$$\begin{pmatrix} 1 & 1 & -2 & / & 5 \\ 1 & -2 & 1 & / & 7 \\ 1 & -3 & 2 & / & 1 \end{pmatrix} \approx$$

$$\begin{pmatrix} 1 & 1 & -2 & / & 5 \\ 0 & -4 & 5 & / & -3 \\ 0 & -4 & 4 & / & -4 \end{pmatrix} \approx$$

$$\left(\begin{array}{cccc} 1 & 1 & -2 & /5 \\ 0 & -4 & 5 & / -3 \\ 0 & 0 & -1 & / -1 \end{array}\right) \approx$$

$$\left(\begin{array}{cccc} 1 & 1 & -2 & / 5 \\ 0 & -4 & 5 & / -3 \\ 0 & 0 & 1 & / 1 \end{array}\right) \approx$$

$$\left(\begin{array}{cccc} 1 & 1 & -2 & / 5 \\ 0 & -4 & 0 & / -8 \\ 0 & 0 & 1 & / 1 \end{array}\right) \approx$$

$$\left(\begin{array}{cccc} 1 & 1 & 0 & / 7 \\ 0 & 1 & 0 & / 2 \\ 0 & 0 & 1 & / 1 \end{array}\right) \approx$$

$$\left(\begin{array}{cccc} 1 & 0 & 0 & / 5 \\ 0 & 1 & 0 & / 2 \\ 0 & 0 & 1 & / 1 \end{array}\right) \approx$$

Řešení soustavy je: $x_1 = 5, x_2 = 2, x_3 = 1$.

Příklad 2.2 :

Příklad řešení soustavy lineárních rovnic, která má nekonečně mnoho řešení.

V příkladě se základní řešení určuje užitím Jordanovy metody.

$x_1 + x_2 - 2x_3 + 3x_4 = 2$

$-2x_1 - x_2 - 3x_3 + x_4 = -1$

Soustavě lineárních rovnic přiřadíme rozšířenou matici soustavy a tu upravíme na schodovitý tvar.

$$\left(\begin{array}{cccc} 1 & 1 & -2 & 3 & / 2 \\ 1 & 1 & -2 & 3 & / 2 \end{array}\right) \approx$$

$$\left(\begin{array}{cccc|c} -2 & -1 & -3 & 1 & -1 \\ 0 & 1 & -7 & 7 & 3 \end{array}\right)$$

Hodnost matice soustavy je h = 2, hodnost rozšířené matice soustavy je h´= 2, počet neznámých je n = 4. Soustava má nekonečně mnoho řešení, přičemž dvě neznámé jsou volitelné. Dále provedeme takovou elementární úpravu, aby prvek v 1. řádku a 2.sloupci byl roven 0, tj. :

$$\left(\begin{array}{cccc|c} 1 & 1 & -2 & 3 & 2 \\ 0 & 1 & -7 & 7 & 3 \end{array}\right) \approx$$

$$\left(\begin{array}{cccc|c} 1 & 0 & 5 & -4 & -1 \\ 0 & 1 & -7 & 7 & 3 \end{array}\right)$$

Při volbě $x_3 = 0$, $x_4 = 0$ získáme přímo základní (bázické) proměnné $x_1 = 1$, $x_2 = 3$, jedno ze základních řešení $x_I = (-1,3,0,0)$.

Budeme-li chtít, aby základní (bazickými) proměnnými byly místo x_1,x_2 proměnné x_1,x_3, upravíme matici tak, aby jednotkový vektor (0,1) byl ve třetím sloupci, tj. místo prvku $a_{23} = -7$ byla jednička a nad ní nula.

$$\left(\begin{array}{cccc|c} 1 & 0 & 5 & -4 & -1 \\ 0 & 1 & -7 & 7 & 3 \end{array}\right) \approx$$

$$\left(\begin{array}{cccc|c} 1 & 0 & 5 & -4 & -1 \\ 0 & -1/7 & 1 & -1 & -3/7 \end{array}\right) \approx$$

$$\left(\begin{array}{cccc|c} 1 & 5/7 & 0 & 1 & 8/7 \\ 0 & -1/7 & 1 & -1 & -3/7 \end{array}\right)$$

Při volbě $x_2 = 0$, $x_4 = 0$ získáme přímo základní (bázické) proměnné $x_1 = 8/7$, $x_3 = -3/7$, tj. další základní řešení $x_{II} = (8/7,0,-3/7,0)$. Určíme-li, že základními (bazickými) proměnnými budou x_1,x_4, upravíme matici tak, aby jednotkový vektor byl ve čtvrtém sloupci, tj. místo klíčového prvku (definovaný v kapitole simplexové metody) $a_{24} = -1$ byla jednička a nad ní nula.

$$\begin{pmatrix} 1 & 5/7 & 0 & 1 & / 8/7 \\ 0 & -1/7 & 1 & -1 & / -3/7 \end{pmatrix} \approx$$

$$\begin{pmatrix} 1 & 4/7 & 1 & 0 & / 5/7 \\ 0 & -1/7 & 1 & -1 & / -3/7 \end{pmatrix} \approx$$

$$\begin{pmatrix} 1 & 4/7 & 1 & 0 & / 5/7 \\ 0 & 1/7 & -1 & 1 & / 3/7 \end{pmatrix}$$

Při volbě $x_2 = 0, x_3 = 0$ získáme přímo základní (bázické) proměnné $x_1 = 5/7$, $x_4 = 3/7$, tj. další základní řešení $x_{III} = (5/7,0,0,3/7)$.
Zbývající základní řešení určíme analogicky. Sloupce příslušné k základním (bazickým) proměnným tvoří vždy v upravené matici submatici 2. řádu.

Řešení $x_{IV} = (0,0,5/7,8/7)$ odpovídá matici $\begin{pmatrix} 1 & 4/7 & 1 & 0 & / & 5/7 \\ 1 & 5/7 & 0 & 1 & / & 8/7 \end{pmatrix}$

Řešení $x_V = (0,5/4,0,1/4)$ odpovídá matici $\begin{pmatrix} 7/4 & 1 & 7/4 & 0 & / & 5/4 \\ -1/4 & 0 & -5/4 & 1 & / & 1/4 \end{pmatrix}$

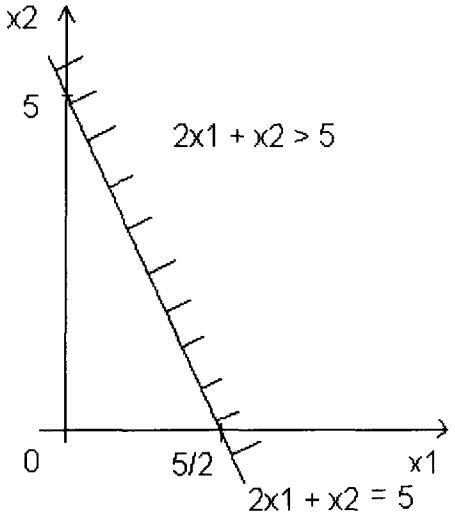
Řešení $x_{IV} = (0,8/5,-1/5,0)$ odpovídá matici $\begin{pmatrix} 7/5 & 1 & 0 & 7/5 & / & 8/5 \\ 1/5 & 0 & 1 & -4/5 & / & -1/5 \end{pmatrix}$

2.6 Lineární nerovnice – řešené graficky

Úlohy lineárního programování vedou k řešení soustavy lineárních nerovnic. Pohybujeme-li se v rovině víme, že lineární rovnice o dvou neznámých $a_{i1} + a_{i2} = b_i$, lze znázornit přímkou, která rozděluje rovinu na dvě poloroviny.

Příklad 2.3 :

Vyřešit graficky nerovnici o dvou neznámých $2x_1 + x_2 > 5$.



Nerovnici o dvou neznámých $2x_1 + x_2 > 5$ jsme znázornili pomocí poloroviny. Body ležící v této polorovině, vyhovují řešení dané nerovnice. Hraniční body přímky $2x_1 + x_2 = 5$ do řešení nepatří.

Máme-li určit řešení soustavy nerovnic o dvou neznámých hledáme bod (body) společný všem polorovinám. Mohou nastat tyto možnosti:

a) poloroviny nemají žádný společný bod
b) průnik tvoří aspoň jedna polopřímka
c) průnik polorovin tvoří mnohoúhelník

Nerovnost o třech neznámých $a_{i1}x_1 + a_{i2}x_2 + a_{i3}x_3 = b_i$ (odpovídá rovina), lze analogicky znázornit poloprostorem. Poloprostor stejně jako polorovina je konvexní množina. Při řešení tohoto případu mohou nastat opět tři možnosti. Je třeba upozornit, že množina řešení je konvexní, neboť je průnikem konvexních polorovin (poloprostorů).

Lineární rovnice o n neznámých $a_{i1}x_1 + a_{i2}x_2 + \ldots + a_{in}x_n = b_i$ určuje nadrovinu ve vícerozměrném prostoru Rozdělí n-rozměrný prostor na dva poloprostory:

$a_{i1}x_1 + a_{i2}x_2 + \ldots + a_{in}x_n \leq b_i$ a $a_{i1}x_1 + a_{i2}x_2 + \ldots + a_{in}x_n > b_i$.

Množinu řešení soustavy nerovností o n neznámých tvoří společná část poloprostorů odpovídajících nerovnostem soustavy. Je to tedy množina konvexní.

Soustava

$$a_{11}x_1 + a_{12}x_2 + \ldots + a_{1j}x_j + \ldots + a_{1n}x_n = b_1$$
$$a_{21}x_2 + a_{22}x_2 + \ldots + a_{2j}x_j + \ldots + a_{2n}x_n = b_2$$
$$\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots$$
$$a_{i1}x_1 + a_{i2}x_2 + \ldots + a_{ij}x_j + \ldots + a_{in}x_n = b_i$$
$$\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots$$
$$a_{m1}x_1 + a_{m2}x_2 + \ldots + a_{mj}x_j + \ldots + a_{mn}x_n = b_m$$

je konvexní množinou v n- rozměrném prostoru. V takovéto formě lze zapsat libovolnou soustavu nerovnic. (Například, znak nerovnosti lze změnit na opačný vynásobením obou stran dané nerovnosti číslem (-1)).

Řešíme-li úlohu lineárního programování, máme zadanou soustavu lineárních rovnic a nerovnic, které nám určují konvexní množinu. Hledáme body této konvexní množiny, ve kterých nastane minimum nebo maximum pro danou účelovou funkci, která vyjadřuje cíl úlohy.

3. LINEÁRNÍ PROGRAMOVÁNÍ

3.1 Historie lineárního programování

Prvopočátky lineárního programování sahají do období konce 19. století, zmínku o lineárním programování můžeme nalézt v pracích Fouriera z roku 1888. Dále se těmto myšlenkám věnoval maďarský matematik Farkas začátkem 20. století, který jako první vytvořil teorii soustav lineárních nerovností považovanou za základ teorie lineárního programování. Směr vývoje se ubíral k řešení speciálních úloh lineárního programování, k přiřazovacímu a dopravnímu problému. V roce 1941 americký matematik Hitchcock vypracoval postup pro řešení dopravního problému.
Největší vývin teorie a aplikace lineárního programování je datován od období po druhé světové válce, kde hlavním důvodem byly ekonomické problémy poválečných států.
V roce 1947 Dantzig po mnohaletých diskusích podal obecnou formulaci lineárního programování včetně simplexové metody. Důvodem bylo řešení úloh lineárního programování zformulovaných pro potřeby letectva USA. Její název „simplexová" pochází z prvních speciálních úloh, kde množina přípustných řešení byla simplexem, což je konvexní obal množiny nezávislých bodů (jednorozměrný simplex je úsečka, dvojrozměrný trojúhelník, trojrozměrný čtyřstěn, ...).
Vývin dalších oblastí aplikací lineárního programování začal v letech 1950 – 1960.
S nástupem výpočetní techniky vznikali nová řešení, elipsoidová metoda, Karmarkarova metoda.

3.2 Obecná úloha lineárního programování

Úlohy lineárního programování jsou sestaveny pomocí otázek efektivního využívání omezených prostředků, potřebných k dosažení určitých cílů. Metody lineárního programování mají v praxi široké uplatnění. Např. při určování optimální ceny výroby, pro optimalizaci vysokopecní vsázky, pro optimální rozřezávání materiálů , pro optimální plány přepravy produkce od dodavatelů ke spotřebitelům, atd.

Protože základním podmínkám vyhovuje velké množství řešení, nejlepší řešení závisí na zaměření dané úlohy. Pak řešení, které daným podmínkám vyhovuje nazýváme *optimálním řešením.*

Důkazy uvedených vět nalezneme v literatuře [13].

Formulujeme obecnou úlohu lineárního programování takto:

Definice 3.1

Nalézt extrém maximum nebo minimum (lineární) účelové funkce

$$f(X) = \sum_{j=1}^{n} c_j x_j , \quad \text{kde } X = (x_{1,\ldots,} x_n) \qquad (1)$$

za podmínek: $\sum_{j=1}^{n} a_{ij}x_j = b_i$, i = 1,2, …, k,

$\sum_{j=1}^{n} a_{ij}x_j \leq b_i$, i = k + 1,…,m,

$x_j \geq 0$, j = 1,2,...,n (2)

kde $a_{ij}, b_i, c_j \in R$, veličiny c_j se nazývají koeficienty účelové funkce,

j = 1,2…n ; i = 1,2…m ; m < n.

Úloha je ve tvaru:

a) obecném (základním), jestliže v podmínkách úlohy řešíme soustavu nerovnic $\mathbf{Ax} \geq \mathbf{b}$, pro vektor **x** omezení nejsou
b) standardním, jestliže v podmínkách úlohy řešíme soustavu nerovnic $\mathbf{Ax} \geq \mathbf{b}$ a pro vektor platí $\mathbf{x} \geq 0$
c) kanonickém, jestliže v podmínkách úlohy řešíme soustavu nerovnic $\mathbf{Ax} = \mathbf{b}$ a pro vektor platí $\mathbf{x} \geq 0$.

Kanonická a standardní úloha jsou speciálním případem obecné úlohy. Nyní ukážeme převedení obecné úlohy na kanonickou úlohu: Máme úlohu lineárního programování v obecném tvaru:

Minimalizujte účelovou funkci $f(x) = c_1x_1 + c_2x_2 + \ldots + c_jx_j + \ldots + c_nx_n$ (I)

za podmínek: $x_j \geq 0$, j = 1,2,…,n (II)

$$
\begin{aligned}
&a_{11}x_1 + a_{12}x_2 + \ldots + a_{1j}x_j + \ldots + a_{1n}x_n \geq b_1 \\
&a_{21}x_2 + a_{22}x_2 + \ldots + a_{2j}x_j + \ldots + a_{2n}x_n \geq b_2 \\
&\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \\
&a_{i1}x_1 + a_{i2}x_2 + \ldots + a_{ij}x_j + \ldots + a_{in}x_n \geq b_i \\
&\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \\
&a_{m1}x_1 + a_{m2}x_2 + \ldots + a_{mj}x_j + \ldots + a_{mn}x_n \geq b_m
\end{aligned}
\qquad \text{(III)}
$$

Zavedeme nové omezující *doplňkové proměnné* v soustavě (III). Přeměnili jsme soustavu nerovnic (III) na soustavu rovnic:

$$
\begin{aligned}
&a_{11}x_1 + a_{12}x_2 + \ldots + a_{1j}x_j + \ldots + a_{1n}x_n - x_{n+i} = b_1 \\
&a_{21}x_2 + a_{22}x_2 + \ldots + a_{2j}x_j + \ldots + a_{2n}x_n - x_{n+2} = b_2 \\
&a_{i1}x_1 + a_{i2}x_2 + \ldots + a_{ij}x_j + \ldots + a_{in}x_n - x_{n+i} = b_i \\
&a_{m1}x_1 + a_{m2}x_2 + \ldots + a_{mj}x_j + \ldots + a_{mn}x_n - x_{n+m} = b_m
\end{aligned}
$$

Odečetli jsme z levé strany každé nerovnice nějaké neznámé nezáporné číslo. Při tomto převedení se nezmění řešení původní úlohy v obecném tvaru.

Každou úlohu lineárního programování v obecném tvaru lze ekvivalentně převést na tvar standardní, neboť každou rovnici $\sum_{i=1}^{n} a_{ij}x_j = b_i$ můžeme nahradit dvěma nerovnicemi

$$\sum_{i=1}^{n} a_{ij}x_j \leq b_i, \quad \sum_{i=1}^{n} a_{ij}x_j \geq b_i .$$

Kanonickou úlohu, kde není splněna podmínka nezápornosti proměnných x_j, lze ekvivalentně převést na úlohu, kde je tato podmínka splněna. Zavedeme substituci :

$x_i = x'_i - x''_i$

$x'_i = \max(x_i, 0)$

$x''_i = \max(-x_i, 0)$, $i = 1,2,\ldots,n$, $x'_i \geq 0$, $x''_i \geq 0$.

Poznámka: Pokud máme určit maximum funkce f(x) dané v úvodní úloze, musíme si uvědomit, že platí: $\sum_{i=1}^{n} c_j x_j = \max \mathbf{c}^T\mathbf{x} = -\min(-\mathbf{c})^T\mathbf{x} = -\min \sum_{i=1}^{n} (-c_j)x_j$

Definice 3.2

Přípustným řešením úlohy lineárního programování je každý vektor $\mathbf{x} = (x_1, x_2, \ldots, x_n)$, který vyhovuje podmínkám (II) a (III).

Věta 3.1

Množina přípustných řešení úlohy lineárního programování (I), (II), (III) je konvexní.

Konvexní množina řešení má označení K. Tato množina je určena jednoznačně soustavou lineárních rovnic a nerovnic (II) a (III). Mohou nastat tyto případy:

a) množina K konvexním polyedrem (omezená)
b) množina K je neohraničenou konvexní oblastí
c) množina K je prázdná

Definice 3.3

Přípustné řešení úlohy lineárního programování, které minimalizuje (maximalizuje) účelovou funkci $f(x) = \sum_{j=1}^{n} c_j x_j$, se nazývá *optimální řešení.*

Řešení soustavy rovnic a úloh lineárního programování má mnoho přípustných řešení, z nichž máme vybrat řešení optimální. Někdy je nemožné určit všechna řešení a vyzkoušet, která z nich optimalizuje účelovou funkci. Proto nám může pomoci omezení na tzv. řešení základní (bázická).

Základní řešení úlohy lineárního programování má nejvýše tolik kladných souřadnic, kolik je v dané soustavě rovnic. Zbývající souřadnice se rovnají nule. Je-li v soustavě m rovnic a n neznámých, bázické řešení dostaneme, jestliže položíme n-m proměnných rovno nule a soustavu řešíme pouze pro m proměnných. Tyto m proměnné nazýváme bázickými.

Definice 3.4

Nedegenerované bázické řešení je takové, jehož všechny bázické proměnné jsou kladné.

Věta 3.2

Účelová funkce f(x) definovaná na K (K – konvexní polyedr nebo konvexní oblast) nabývá minima (maxima) v krajním bodě K.

Jestliže tato funkce nabývá minima (maxima) ve více krajních bodech (vrcholech) K, pak nabývá této hodnoty i v libovolném bodě, který je konvexní kombinací těchto krajních bodů.

Označme $\mathbf{a}_1,\mathbf{a}_2,\ldots,\mathbf{a}_n$ sloupcové vektory matice **A** typu mxn v (III), $\mathbf{A}_0 = \mathbf{b}$.

Věta 3.3

Je-li soustava vektorů $\mathbf{a}_1,\mathbf{a}_2,\ldots,\mathbf{a}_n$ ($k \leq n$) matice A lineárně nezávislá a je-li $x_1\mathbf{a}_1 + x_2\mathbf{a}_2 + \ldots + x_k\mathbf{a}_k = \mathbf{a}_0$ ($x_1 \geq 0$), pak $\mathbf{x} = (x_1,x_2,\ldots x_k,0,\ldots,0)$ je *krajním bodem množiny přípustných řešení* K této úlohy.

Věta 3.4

Je-li $X = (x_1,x_2,\ldots x_n)$ krajním bodem množiny všech přípustných řešení K úlohy lineárního programování, pak tvoří sloupcový vektor $\mathbf{a}_j$ matice **A** odpovídající kladným x_j ve vztahu $\sum_{j=1}^{n} x_j\mathbf{a}_j = \mathbf{a}_0$ lineárně nezávislou soustavu.

Důsledek: Každému krajnímu bodu z K odpovídá m lineárně nezávislých vektorů z dané soustavy $\mathbf{a}_1,\mathbf{a}_2,\ldots,\mathbf{a}_n$.

Z uvedených vět vyplývá, že při řešení úloh lineárního programování se můžeme soustředit pouze na krajní body K (řešení bázické). Vyšetřením těchto bodů lze nejefektivněji provést pomocí simplexové metody. Tato metoda nám umožní najít libovolný krajní bod a určit, zda je optimální. Pokud tomu tak není, vyšetřuje se sousední bod. (Dva vrcholy konvexní množiny, které jsou incidentní s touže hranou, se nazývá sousední). Takto postupujeme dále a po konečném počtu kroků dojdeme k výsledku, tj. minimum dané účelové lineární funkce.
Simplexovou metodou zjistíme, že úloha nemá přípustné řešení nebo zjistíme neohraničenost účelové funkce na množině přípustných řešení.

3.3 Simplexová metoda

Důkazy následujících vět jsou uvedeny v [5], [12] nebo [13].

Budeme se věnovat simplexové metodě, jejím základním prvkům a výpočetnímu algoritmu, pomocí kterého je možné najít krajní bod a určit jestli je optimální. Metoda je založena na tvrzení, že existuje-li minimální řešení úlohy lineárního programování, je řešení dosaženo v alespoň jednom vrcholu konvexního polyedru.
Metoda hledání minima účelové funkce:
Vyjdeme z libovolného vrcholu X_1 konvexního polyedru K a ověříme, neodpovídá-li již minimálnímu řešení úlohy. Pokud vyhovuje, výpočet skončil. V opačném případě ověříme, zda nevychází z tohoto vrcholu X_1 hrana konvexního polyedru K, podél níž účelová funkce f(x) neomezeně klesá. Existuje-li taková hrana, úloha nemá řešení a výpočet skončil. Jestliže je X_1 řešením , ale není optimálním řešením, přejdeme do vrcholu X_2 konvexního polyedru K (vrchol X_2 je sousední k X_1), kde je hodnota účelové funkce menší. V tomto vrcholu X_2 postup opakujeme. Konvexní polyedr má konečný počet vrcholů, po konečném počtu kroků najdeme vrchol, ve kterém nastává minimum nebo dokážeme, že minimální řešení úlohy neexistuje.
Postup je uveden v následujících krocích:

1. výchozí přípustné bázické řešení
2. běžný krok simplexového algoritmu
3. test optimality

Budeme řešit kanonickou úlohu lineárního programování tj. minimum účelové funkce

$f(x) = \sum_{j=1}^{n} c_j x_j$ za podmínek:

$\sum_{j=1}^{n} a_{ij} x_j = b_i, \qquad i = 1,2, \ldots,m$

$x_j \geq 0, \qquad j = 1,2,\ldots,n$

a předpokládáme, že

- hodnost matice $\mathbf{A} = (a_{ij})$ je rovna m
- $1 \leq m \leq n$
- $b_i \geq 0$ $(i = 1,2, \ldots,m)$
- $K \neq \emptyset$ (K je množina, na níž řešíme kanonickou úlohu),

$$K = \{ X \in E_n : \sum_{j=1}^{n} a_{ij}x_j = b_i,\ i = 1,2, \ldots,m,\ x_i \geq 0,\ j = 1,2,...,n \}.$$

Tyto předpoklady neomezují obecnou úlohu lineárního programování.

Důkazy následujících vět jsou uvedeny v literatuře [5], [13].

3.4 Výchozí přípustné bázické řešení

Rozlišujeme dva případy, které mohou nastat pro úlohu lineárního programování:

I. Ze zadání je výchozí přípustné bázické řešení okamžitě zřejmé.
Jestliže matice **A** obsahuje jednotkovou submatici **I** řádu m, pak výchozí přípustné bázické řešení dostaneme tak, že položíme $x_i = b_i$, když i je index sloupce, který je mezi sloupci submatice **I**, a $x_i = 0$, v opačném případě.

Obecný postup:

1. Upravíme všechny rovnice (nerovnice) tak, aby pro všechny b_i platilo
$$b_i \geq 0 \qquad (i = 1,2, \ldots,m).$$
2. Obsahuje-li matice **A** jednotkovou submatici řádu m/n, pak proměnné x_i $(i = 1,2, \ldots,m)$ odpovídající jednotkovým sloupcům této submatice položíme rovny číslům b_i $(i = 1,2, \ldots,m)$ a ostatní proměnné x_j $(j = m+1,\ldots,n)$ položíme rovny nule. Bod X_1 o souřadnicích $x_i = b_i \geq 0$ $(i = 1,2, \ldots,m)$, $x_j = 0$ $(j = m+1,\ldots,n)$ je výchozím přípustným bazickým řešením úlohy lineárního programování.

II. Výchozí přípustné bázické řešení úlohy není okamžitě zřejmé ze zadání. Matice **A** neobsahuje jednotkovou submatici. Tuto situaci budeme řešit *metodou umělé báze*.

Sestavíme novou úlohu, kterou budeme řešit místo původní kanonické úlohy. Máme řešit úlohu lineárního programování:

Nalézt minimum funkce $f_1(\mathbf{x},\mathbf{u}) = \sum_{j=1}^{n} c_j x_j + w \sum_{i=1}^{m} u_i$, kde w je libovolné velké číslo, na množině $K1 = \{(\mathbf{x},\mathbf{u}) \in E_{n+m} : \sum_{j=1}^{n} a_{ij}x_j + u_i = b_i,\ i = 1,2,\ldots,m,\ x_i \geq 0,\ u_i \geq 0,\ j = 1,2,\ldots,n\}$. Koeficienty proměnných u_i $(i = 1,2,\ldots,m)$ vytváří jednotkovou matici řádu m/n.

Systém rovnic $\sum_{j=1}^{n} a_{ij}x_j + u_i = b_i$ řešíme vzhledem k proměnným u_i $(i = 1,2,\ldots,m)$. Bod (X_0,U_0) je výchozím přípustným bazickým řešením úlohy, jehož n souřadnic je nulových a ostatní souřadnice jsou řešením systému rovnic $u_i = b_i$ $(i = 1,2,\ldots,m)$.

Definice 3.5

Proměnné u_i $(i = 1,2,\ldots,m)$ nazýváme *umělými proměnnými.*

Věta 3.5

Jestliže existuje optimální (minimální) řešení $(X_0,U_0) = (x_{01},\ldots,x_{0n},u_{01},\ldots,u_{0m})$ nové úlohy, pro než platí $\sum_{i=1}^{m} u_{0i} > 0$, kde $U_0 = (u_{01}, u_{02},\ldots,u_{0m})$, potom množina K je prázdná a tedy původní úloha nemá řešení.

Věta 3.6

Existuje-li optimální (minimální) řešení (X_0,U_0) nové úlohy, pro než platí $\sum_{i=1}^{m} u_{0i} = 0$, potom bod X_0 je optimálním (minimálním) řešením původní úlohy.

Věta 3.7

Neexistuje-li optimální (minimální) řešení nové úlohy, pak také neexistuje optimální (minimální) řešení původní úlohy.

Definice 3.6

Jestliže v nějakém kroku simplexové metody získáme vrchol konvexního polyedru K, který má více než (n – m) nulových souřadnic (přípustné bázické řešení úlohy je degenerované), potom říkáme, že v tomto kroku nastává degenerace.

Výchozí přípustné bázické řešení lze určit i jinými způsoby. Uvedeme nejčastěji používaný.

Je-li úloha dána omezeními ve tvaru $a_{i1}x_1 + a_{i2}x_2 + \ldots + a_{in}x_n \geq b_i$, (i = 1,2,…,m), pak pomocí doplňkových proměnných upravíme rovnici $a_{i1}x_1 + a_{i2}x_2 + \ldots + a_{in}x_n - x_{n+i} = b_i$, (i = 1,2,…,m). Měli bychom zavést m umělých proměnných. Ovšem vylučování umělých proměnných z řešení vyžaduje nejméně tolik kroků, kolik je umělých proměnných. Je vhodné zavést umělé proměnné jen v té míře, jak je to nezbytně nutné. V našem případě lze matici koeficientů získat z jednotkové matice nepatrnou úpravou původních podmínek. Vybereme rovnici s největším absolutním členem b_i a odečteme od ní postupně ostatní rovnice. Dostaneme v matici m – 1 jednotkových sloupcových vektorů a stačí tedy doplnit jedinou umělou proměnnou (v rovnici s největším absolutním členem).

Poznámka. Pomocí doplňkových proměnných přecházíme ze soustavy nerovnic na ekvivalentní soustavu rovnic. Doplňkové proměnné nemají vliv na výsledek. Přidáním umělých proměnných jsme přešli ze soustavy rovnic na rozšířenou soustavu rovnic, která s ní není ekvivalentní. Řešení rozšířené soustavy nemusí odpovídat žádnému řešení původní soustavy. Úkolem je vyřešit rozšířenou soustavu tak, aby se v ní umělé proměnné rovnaly nule. (Jen v tomto případě bude řešení původní soustavy ekvivalentní s řešením rozšířené soustavy).

Test optimality

Nové přípustné bázické řešení, které dostaneme v běžném kroku simplexové metody, je nutno prověřit a rozhodnout jestli již dává optimální řešení a nebo je nutno pokračovat. Tento test provedeme dle následujících vět, jejichž důkazy jsou uvedeny v literatuře [3], [5].

Věta 3.8

Nechť pro některé $j \in \{ m+1,\ldots,n\}$ jest $z_j - c_j > 0$ a pro všechna i = 1,2,…,m jest $x_{ij} \leq 0$. Pak účelová funkce f(**X**) je zdola neohraničená.

Věta 3.9

Nechť pro některé $j \in \{ m+1,\ldots,n\}$ jest $z_j - c_j > 0$ a pro některé $i \in \{1,\ldots,m\}$ jest

$x_{ij} > 0$. Pak $f(\mathbf{X_2}) \leq f(\mathbf{X_1})$. V případě, že X_1 je nedegenerované řešení platí

$$f(\mathbf{X_2}) < f(\mathbf{X_1}).$$

Věta 3.10

Nechť pro všechna $j \in \{ m+1,\ldots,n\}$ jest $z_j - c_j \leq 0$, pak $f(\mathbf{X_1})$ je minimum účelové funkce $f(\mathbf{X})$ na množině dané systémem omezení. V případě, že $\mathbf{X_1}$ je nedegenerované řešení, platí i tvrzení opačné. Pak simplexový algoritmus dospěl k optimálnímu řešení a končí.

3.5 Algoritmus simplexové metody

V následujících krocích je shrnutý postup algoritmu:

1. Zadanou úlohu lineárního programování převedeme na kanonický tvar a ověříme, že jsou splněny obecné předpoklady.
2. Určíme výchozí přípustné bázické řešení a zapíšeme do následující *simplexové tabulky*:

Tabulka 3.1 :

c_j		c_1 ..... c_m c_{m+1} ..... c_n	
c_i	báze	x_1 ..... x_m x_{m+1} ..... x_n	b_i
c_1	x_1	1 0 ..... 0 $x_{1\,m+1}$ ..... x_{1n}	b_1
.	.	0	.
.	.	. . . . .	.
.	.	. . . . .	.
.	.	. . . . .	.
.	.	. . . . .	.
.	.	. . . . .	.
.	.	. . . . .	.
c_m	x_m	0 ..... 1 $x_{m\,m+1}$ ..... x_{mn}	b_m
	$z_j - c_j$	$z_1 - c_1$ ..... $z_n - c_n$	$f(\mathbf{X})$

Kde $\mathbf{a}_1, \ldots, \mathbf{a}_m$ jsou bázové vektory.

3. Provedeme test optimality dle výše uvedených vět.Tabulku doplníme o vypočítané hodnoty $z_j - c_j$ (pro bázové vektory $z_j - c_j = 0$) a určíme, zda pro některé j platí, že $z_j - c_j > 0$. Jestliže takové j najdeme, pokračujeme dalším krokem, pokud takové j neexistuje platí:

- není-li v bázi žádná umělá proměnná nebo nejsou v bázi umělé proměnné s nulovou hodnotou, pak nalezené řešení je *optimální*

- jestliže v bázi zůstala aspoň jedna proměnná, která má kladnou hodnotu, pak úloha nemá řešení.

4. Jsou-li splněny předpoklady z věty 3.9, algoritmus pokračuje tak, že běžným krokem najdeme nové přípustné bázické řešení. Při přechodu k novému řešení, tedy k nové bázi, musíme určit vektor, který do báze vstupuje. Tento vektor určíme ze vztahu $f(\mathbf{X_2}) = f(\mathbf{X_1}) - \theta.(z_j - c_j)$. Aby výpočet byl efektivní, budeme se k optimu,tj. k minimu, blížit co nejrychleji a proto zvolíme j_o tak, aby

$$z_{jo} - c_{jo} = \max\{z_h - c_h : \text{h není bázový index}\}.$$

Pak vektor $\mathbf{a}_{jo}$ vstupuje do nové báze a index j_o určuje v simplexové tabulce tzv. *klíčový sloupec*. Není-li určen jednoznačně, zvolíme libovolný index popsané vlastnosti.

5. rozhodneme, která z proměnných z báze vystupuje. Pro j, které je stanoveno v kroku 4. mohou nastat tyto případy:

a) existuje-li aspoň jedno i tak, že $x_{ij} > 0$, vypočteme číslo $\delta = \min\{x'_i / x_{ij}\} = (x'_{i0}/x_{i0j})$. Číslo i0 určuje index vektoru, který z báze vystupuje, tzv. klíčového řádku. Je-li dosaženo minima pro více hodnot i0, bereme zpravidla nejmenší z nich. Přejdeme k dalšímu kroku.

b) Je-li $x_{ij} < 0$ pro všechna i, pak úloha má přípustné řešení s libovolně malou hodnotou účelové funkce. Je-li navíc umělá proměnná v bázi, nemá úloha řešení, výpočet ukončíme.

Prvek v průsečíku klíčového řádku a klíčového sloupce se nazývá *klíčový prvek.*

6. Změna báze popsaná v bodech 4. a 5. nám určí nové přípustné bázické řešení. Pak také musíme změnit simplexovou tabulku dle vztahů uvedených v části 2. algoritmus se vrací k bodu 2., kde místo výchozího se bere nové přípustné bázické řešení.

Uvedené věty jsou dokázány v literatuře [3].

Věta 3.11

Nechť $(x_{1j},\ldots\ldots,x_{nj})$, případně $(x'_{1j},\ldots\ldots,x'_{nj})$ jsou souřadnice vektoru $\mathbf{a}_j$ $(j = 1,\ldots,n)$ v bázi tvořené vektory $\mathbf{a}_1,\ldots, \mathbf{a}_m$, případně $\mathbf{a}_1,\ldots, \mathbf{a}_{io-1}, \mathbf{a}_{jo}, \mathbf{a}_{io+1},\ldots,\mathbf{a}_m$, kde (i_o, j_o) jsou indexy klíčového prvku. Pak platí:

I. $x'_{ij} = x_{ij} - x_{ijo} \cdot x_{ioj} \cdot x_{iojo}^{-1}$, pro $i = 1,\ldots,n,\ i \neq i_o$

II. $x'_{ioj} = x_{ioj} \cdot x_{iojo}^{-1}$, pro $j = 1,\ldots,n$

III. $(z_j - c_j)' = (z_j - c_j) - (z_{jo} - c_{jo}) \cdot x_{ioj} \cdot x_{iojo}^{-1}$, pro $j = 1,\ldots,n$

Nenastane-li degenerace během simplexového algoritmu, tak po konečném počtu kroků tento algoritmus končí.

Věta 3.12

Pro simplexový algoritmus, při kterém nenastane degenerace v žádném kroku, platí právě jeden z následujících případů:

Po konečném počtu běžných kroků jsou splněny předpoklady věty 3.8, optimální řešení neexistuje.

Po konečném počtu běžných kroků jsou splněny předpoklady věty 3.10, optimální řešení je nalezeno.

Příklad 3.1:

Určit minimum funkce $f(\mathbf{x}) = -x_1 + x_2$ při omezeních:

$-2x_1 + x_2 + x_3 = 2$

$x_1 - 2x_2 + x_4 = 2$

$x_1 + x_2 + x_5 = 5$, $x_i \geq 0$ pro i = 1,2,3,4,5.

Řešení:

Označme $\mathbf{a}_1 = (-2,1,1)T$, $\mathbf{a}_2 = (1,-2,1)^T$, $\mathbf{a}_3 = (1,0,0)^T$, $\mathbf{a}_4 = (0,1,0)^T$, $\mathbf{a}_5 = (0,0,1)^T$, $\mathbf{b} = (2,2,5)^T$. Pak $\{ \mathbf{a}_3, \mathbf{a}_4, \mathbf{a}_5 \}$ je báze a platí $\mathbf{a}_1 = -2\,\mathbf{a}_3 + \mathbf{a}_4 + \mathbf{a}_5$, $\mathbf{a}_2 = \mathbf{a}_3 - 2\,\mathbf{a}_4 + 5\,\mathbf{a}_5$. Výchozí přípustné bázické řešení je $X_1 = (0,0,2,2,5)$. Dále $f(X_1) = 0$ a sestavíme výchozí simplexovou tabulku:

Tabulka 3.2 :

$\mathbf{c}_j$		-1	1	0	0	0	
	báze	x_1	x_2	x_3	x_4	x_5	$\mathbf{b}_i$
0	x_3	-2	1	1	0	0	2
0	x_4	1	-2	0	1	0	2
0	x_5	1	1	0	0	1	5
	$z_j - c_j$	1	-1	0	0	0	0

$\Theta = \min \{2/1, 5/1\} = 2$

Klíčový prvek je označen v rámečku, a provedeme další krok.

Najdeme nové přípustné bázické řešení X_2:

$B - 2\,\mathbf{a}_1 = (2\,\mathbf{a}_3 + 2\,\mathbf{a}_4 + 5\mathbf{a}_5) - 2(-2\,\mathbf{a}_3 + \mathbf{a}_4 + \mathbf{a}_5) = 6\,\mathbf{a}_3 + 3\,\mathbf{a}_5 \Rightarrow$

$B = 2\,\mathbf{a}_1 + 6\,\mathbf{a}_3 + 3\,\mathbf{a}_5 \Rightarrow X_2 = (2,0,6,0,3)$. Dále počítáme $\mathbf{a}_2 = x_{12}\,\mathbf{a}_1 + x_{32}\,\mathbf{a}_3 + x_{52}\,\mathbf{a}_5$ $\Rightarrow -2x_{12} + x_{32} = 1$, $x_{12} = -2$, $x_{12} + x_{52} = 1$. Odtud $x_{12} = -2$, $x_{32} = -3$, $x_{52} = 3$. Podobně zjistíme $\mathbf{a}_4 = \mathbf{a}_1 + 2\mathbf{a}_3 - \mathbf{a}_5$ a sestavíme novou simplexovou tabulku:

Tabulka 3.3 :

c_j		-1	1	0	0	0	
	báze	x_1	x_2	x_3	x_4	x_5	$\mathbf{b}_i$
-1	x_1	1	-2	0	1	0	2
0	x_3	0	-3	1	2	0	6
0	x_5	0	3	0	-1	1	3
	$z_j - c_j$	0	1	0	-1	0	-2

$\Theta = \min \{3/3\} = 1$

Je nutno provést další krok.

Najdeme nové přípustné bázické řešení X_3:

$\mathbf{b} - \Theta\, \mathbf{a}_2 = 2\, \mathbf{a}_1 + 6\, \mathbf{a}_3 + 3\, \mathbf{a}_5 - (-2\, \mathbf{a}_1 - 3\, \mathbf{a}_3 + 3\, \mathbf{a}_5) \Rightarrow \mathbf{b} = 4\, \mathbf{a}_1 + \mathbf{a}_2 + 9\, \mathbf{a}_3 \Rightarrow$

$X_3 = (4,1,9,0,0)$. Dále počítáme, že platí

$\mathbf{a}_4 = 1/3\, \mathbf{a}_1 - 1/3\, \mathbf{a}_2 + \mathbf{a}_3$, $\mathbf{a}_5 = 2/3\, \mathbf{a}_1 + 1/3\, \mathbf{a}_2 + \mathbf{a}_3$.

Sestavíme novou simplexovou tabulku:

Tabulka 3.4 :

c_j		-1	1	0	0	0	
	báze	x_1	x_2	x_3	x_4	x_5	b_i
-1	x_1	1	0	0	1/3	2/3	4
1	x_2	0	1	0	-1/3	1/3	1
0	x_3	0	0	1	1	1	9
	$z_j - c_j$	0	0	0	-2/3	-1/3	-3

Našli jsme optimální řešení.

Výsledek: Minimum $f(x) = -3$ v bodě $X_3 = (4,1,9,0,0)$.

Užití umělých proměnných si ukážeme na následujícím příkladě.

Příklad 3.2 :

Nalezněte minimum funkce $f(x) = -x_1 - 2x_2 - 3x_3 + x_4$ na množině M dané rovnicemi:

$x_1 + 2x_2 + 3x_3 = 15$

$2x_1 + x_2 + 5x_3 = 20$

$x_1 + x_2 + x_3 + x_4 = 10$, $x_1,x_2,x_3,x_4 \geq 0$

Řešení:

Matice systému omezení obsahuje jednotkový vektor A^4, pak stačí zavést dvě umělé proměnné u_1, u_2 pro snížení počtu iterací.

$f(X) = -x_1 - 2x_2 - 3x_3 + x_4 + wu_1 + wu_2$,

M: $x_1 + 2x_2 + 3x_3 + u_1 = 15$

$2x_1 + x_2 + 5x_3 + u_2 = 20$

$x_1 + x_2 + x_3 + x_4 = 10$

Řešíme opět pomocí simplexové tabulky:

Tabulka 3.5 :

c_j		-1	-2	-3	1	w	w	
	báze	x_1	x_2	x_3	x_4	u_1	u_2	b_i
w	u_1	1	2	3	0	1	0	15
w	u_2	2	1	5	0	0	1	20
1	x_4	1	2	1	1	0	0	10
	$z_j - c_j$	2	4	4	0	0	0	10
		3w	3w	8w	0	-w	-w	35w
w	u_1	-1/5	7/5	0	0	1	-3/5	3
-3	x_3	2/5	1/5	1	0	0	1/5	4
1	x_4	3/5	9/5	0	1	0	-1/5	6
		2/5	16/5	0	0	0	-4/5	-6
	$z_j - c_j$	-1/5w	7/5w	0	0	-w	-8/5w	3w
-2	x_2	-1/7	1	0	0	5/7	-3/7	15/7
-3	x_3	3/7	0	1	0	-1/7	2/7	25/7
1	x_4	6/7	0	0	1	-9/7	4/7	15/7

		6/7	0	0	0	-16/7	4/7	-90/7
	$z_j - c_j$	0	0	0	0	-w	-w	0
-2	x_2	0	1	0	1/6	1/2	-1/3	5/2
-3	x_3	0	0	1	-1/2	1/2	0	5/2
-1	x_1	1	0	0	7/6	-3/2	2/3	5/2
		0	0	0	-1	-1	0	-15
	$z_j - c_j$	0	0	0	0	-w	-w	0

Našli jsme optimální řešení.

Výsledek. Minimum f(x) = -15 v bodě X = (5/2, 5/2, 5/2, 0).

3.6 Degenerace

Pokud v určitém kroku dostaneme přípustné bázické řešení, které má méně než m nenulových souřadnic, nastává případ *degenerace*. V degenerovaném případě může běžný krok simplexové metody přejít od řešení $\mathbf{X_1}$ k $\mathbf{X_2}$ a přitom $f(\mathbf{X_2}) = f(\mathbf{X_1})$.
Je možnost, že nastane zacyklení simplexového algoritmu, tedy získáme posloupnosti řešení $\mathbf{X_1}, \ldots, \mathbf{X_p}$ tak, že $f(\mathbf{X_1}) = \ldots = f(\mathbf{X_p})$ a z toho plynoucí selhání, protože nemůžeme postoupit k optimálnímu řešení.

Nyní uvedeme příčiny degenerace řešení:

Při získání výchozího přípustného bazického řešení, je-li některý absolutní člen b_i v systému omezení roven 0.
Při běžném kroku, je-li klíčový řádek určen nejednoznačně.

Způsob odstranění degenerace provádíme ε - *modifikací úlohy,* kterou požíváme při výchozím přípustném bazickém řešení i při běžném kroku. Postupujeme ve smyslu rušení (perturbace) konstantních prvků v úloze lineárního programování. Tak odstraníme degeneraci a nenarušíme přesnost výpočtu.

Uvedeme geometrickou úvahu odstranění degenerace u simplexové metody.
Vektory báze úlohy tvoří konvexní kužel. Všechny kladné lineární kombinace vektorů báze nám vytvoří vnitřek tohoto kužele. Označme $\mathbf{a}_1,\mathbf{a}_2,\ldots,\mathbf{a}_m$ vektory báze, pro vektor **b** platí $\mathbf{b} = x_1\mathbf{a}_1 + x_2\mathbf{a}_2 + \ldots + x_m\mathbf{a}_m$. Degenerace geometricky znamená, že vektor **b** leží na hranici kužele tvořeného vektory $\mathbf{a}_1,\mathbf{a}_2,\ldots,\mathbf{a}_m$. Pokud chceme degeneraci odstranit, musíme vektor **b** posunout, tj. *perturbovat* (odchýlit) tak, aby ležel ve vnitřku kužele. Toho dosáhneme tak, že k vektoru b připočteme libovolně malou kladnou kombinaci vektorů báze $\mathbf{a}_1,\mathbf{a}_2,\ldots,\mathbf{a}_m$.

Důkazy následujících vět je možno najít v literatuře [3], [13].

Věta 3.13
Ke každé kanonické úloze lineárního programování existuje $\varepsilon_0 > 0$, tak že pro

všechna ε , $0 \leq \varepsilon \leq \varepsilon_0$ nenastane degenerace u přiřazené ε – modifikované úlohy v žádném kroku simplexového algoritmu.

Věta 3.14
Je-li $\mathbf{X}_0(\varepsilon)$ optimální řešení ε – modifikované úlohy, pak $X_0 = \lim_{\varepsilon \to 0} \mathbf{X}_0(\varepsilon)$ je optimální řešení původní úlohy. Neexistuje-li pro žádné $\varepsilon > 0$ optimální řešení ε – modifikované úlohy, pak rovněž neexistuje optimální řešení původní úlohy.

3.7 Duální simplexová metoda

Duální simplexová metoda je jednou z variant simplexové metody, které využívají princip duality. Byla vypracována v roce 1954 C. E. Lemkem. Pomocí této metody lze efektivně řešit úlohy v kanonickém tvaru bez ohledu na podmínky nezápornosti jednotlivých složek vektoru pravých stran. Při výpočtu vycházíme z duálně přípustného řešení a kontrolujeme, zda řešení je přípustným řešením úlohy primární.

Příslušné pojmy a postupy jsou uvedeny v literatuře [3], [5], [13].

3.8 Dualita úloh lineárního programování

Ke každé úloze lineárního programování lze určitým způsobem přiřadit jinou úlohu lineárního programování, úlohu sdruženou neboli duální, která je sestavena z týchž konstant.
Dvojice sdružených úloh spolu úzce souvisí a má řadu zajímavých společných vlastností (jak z hlediska výpočetního, tak z hlediska ekonomické interpretace). Dualita je vztahem vzájemným, kterákoliv úloha z dvojice duálních úloh může být vzata jako primární. Každému vlastnímu omezení primární úlohy odpovídá jedna duální proměnná a každé podmínce nezápornosti primární úlohy odpovídá jedno duální omezení.

Nejprve uvedeme tzv. *souměrné* (symetrické) duální úlohy.

Nechť je dána primární úloha ve tvaru:

Nalézt maximum funkce

$$z = (\mathbf{c})^T\mathbf{x} \qquad (3.1)$$

za podmínek

$$A\mathbf{x} \leq \mathbf{b} \quad \mathbf{x} \geq \boldsymbol{0}$$

Úlohu duální k úloze (3.1) můžeme formulovat takto:

Nalézt minimum funkce

$$f = (\mathbf{b})^T\mathbf{u} \qquad (3.2)$$

za podmínek

$$A^T\mathbf{u} \geq \mathbf{c} \quad \mathbf{u} \geq \boldsymbol{0}$$

Úloha (3.2) je duální k úloze (3.1), ale platí také opak.

Poznámka: Pokud v primární maximalizační úloze máme vlastní omezení vyjádřena nerovnostmi, lze úlohu převést na typ (3.1) tak, že nerovnosti vynásobíme číslem -1.

Pokud jsou v primární úloze vlastní omezení vyjádřena nerovnostmi, ale úloha je minimalizační, lze úlohu převést na typ (3.1) tak, že maximalizujeme funkci $-(\mathbf{c})^T\mathbf{x}$.

Příklad 3.3 :

(Souměrné duální úlohy)

Primární úloha	Duální úloha
Nalézt maximum funkce	Nalézt minimum funkce
$z = 20x_1 + 25x_2 + 30x_3$	$f = 600u_1 + 1200u_2 + 800u_3$
za podmínek	za podmínek
$2x_1 + x_2 + 3x_3 \leq 600$	$u_1 \geq 0$
$x_1 + x_2 + 2x_3 \leq 1200$	$u_2 \geq 0$
$2x_1 + 2x_2 + x_3 \leq 800$	$u_3 \geq 0$
$x_1 \geq 0$	$2u_1 + u_2 + 2u_3 \geq 20$
$x_2 \geq 0$	$u_1 + u_2 + 2u_3 \geq 25$
$x_3 \geq 0$	$3u_1 + 2u_2 + u_3 \geq 30$

Optimální řešení obou úloh:

$z = 11\ 400$	$f = 11\ 400$
$x'_1 = 0$ $x'_2 = 680$ $x'_3 = 0$	$u_1 = 7$ $u_2 = 0$ $u_3 = 9$
$x_1 = 0$ $x_2 = 360$ $x_3 = 80$	$u'_1 = 12$ $u'_2 = 0$ $u'_3 = 0$

Uveďme ještě tzv. nesouměrné (nesymetrické) duální úlohy, které lze odvodit ze souměrných duálních úloh.

Nechť je dána primární úloha ve tvaru:

Nalézt minimum funkce

$$z = (\mathbf{c})^T\mathbf{x} \qquad (3.3)$$

za podmínek

$$A\mathbf{x} = \mathbf{b} \qquad x \geq 0$$

Úlohu duální k úloze (3.3) můžeme formulovat takto:

Nalézt maximum funkce

$$f = (\mathbf{b})^T\mathbf{u} \qquad (3.4)$$

za podmínek

$$A^T\mathbf{u} \leq \mathbf{c} \qquad \mathbf{u} \geq 0 \qquad \mathbf{u} \leq 0$$

(Jsou-li v primární úloze vlastní omezení vyjádřena rovnicemi, jsou odpovídající proměnné v duální úloze neomezené co do znaménka).

Úloha (3.4) je duální k úloze (3.3) a obráceně.

Příklad 3.4 :

(Nesouměrné duální úlohy)

Primární úloha	Duální úloha
Nalézt minimum funkce	Nalézt maximum funkce
$z = 30\,x_1 + 39x_2 + 33\,x_3 + 26x_4$	$f = 186u_1 + 114u_2$
za podmínek	za podmínek
$0{,}5x_1 + 0{,}3x_2 + 0{,}4x_3 + 0{,}7x_4 = 186$	$0 \geq u_1 \geq 0$
$0{,}5x_1 + 0{,}7x_2 + 0{,}6x_3 + 0{,}3x_4 = 114$	$0 \geq u_2 \geq 0$
$x_1 \geq 0$	$0{,}5\,u_1 + 0{,}5u_2 \leq 30$
$x_2 \geq 0$	$0{,}3u_1 + 0{,}7u_2 \leq 39$
$x_3 \geq 0$	$0{,}4u_1 + 0{,}6u_2 \leq 33$
$x_4 \geq 0$	$0{,}7u_1 + 0{,}3u_2 \leq 26$

Optimální řešení obou úloh:

$z = 8280$	$f = 8280$
$x'_1 = 0$ $x'_2 = 0$	$u_1 = 20$ $u_2 = 40$
$x_1 = 120$ $x_2 = 0$ $x_3 = 0$ $x_4 = 180$	$u'_1 = 0$ $u'_2 = 5$ $u'_3 = 1$ $u'_4 = 0$

Příklad 3.5 :

(Nesouměrné duální úlohy – podmínky smíšeného typu)

Primární úloha	Duální úloha
Nalézt maximum funkce	Nalézt minimum funkce
$z = 3x_1 + 2x_2 - x_3 + 3x_4$	$f = 5u_1 + 3u_2 + u_3$
za podmínek	za podmínek
$x_1 - 3x_2 + x_3 = 5$	$0 \geq u_1 \geq 0$
$2x_1 + 2x_2 - x_3 + 2x_4 \leq 3$	$u_2 \geq 0$
$8x_1 - 4x_2 + x_4 \geq 1$	$u_3 \leq 0$
$0 \geq x_1 \geq 0$	$u_1 + 2u_2 + 8u_3 = -3$
$x_2 \geq 0$	$-3u_1 + 2u_2 - 4u_3 \geq 2$
$0 \geq x_3 \geq 0$	$u_1 - u_2 = -1$
$x_4 \geq 0$	$2u_2 + u_3 \geq 3$

Optimální řešení obou úloh:

$z = 10$	$f = 10$
$x_1 = -0,46$	
$x_2 = 0$	$u'_2 = 3$
$x_3 = 5,46$	
$x_4 = 4,69$	$u'_4 = 0$
	$u_1 = 1$
$x'_2 = 0$	$u_2 = 2$
$x'_3 = 0$	$u_3 = -1$

Pro sdružené úlohy platí následující tzv. základní věta o dualitě, a důkazy uvedených vět je možno nalézt v literatuře [3],[5] ,[13].

Věta 3.15

Má-li jedna z dvojice sdružených úloh optimální řešení, má optimální řešení i úloha druhá a optimální hodnoty obou účelových funkcí jsou stejné. Nemá-li jedna z dvojice sdružených úloh přípustné řešení, nemá druhá úloha optimální řešení, tj. buď také nemá přípustné řešení nebo má, ale účelová funkce může neomezeně růst (klesat).

Později při řešení tzv. dopravní úlohy využijeme vlastnosti duálních úloh, kterou vyjadřuje další věta, tz. *podmínky rovnováhy*.

Věta 3.16

Je-li v optimálním řešení jedné z dvojice sdružených úloh příslušné omezení splněno jako ostrá nerovnost, je odpovídající duální omezení splněno jako rovnice.
Je třeba dodat, že není nutno obě duálně sdružené úlohy řešit samostatně. Řešíme-li totiž jednu z úloh simplexovou metodou, získáme tím automaticky i řešení úlohy k ní duální, které je dáno koeficienty účelové funkce v posledním kroku simplexové tabulky. V posledním řádku tabulky nalezneme:

- optimální hodnoty duálních proměnných ve sloupcích odpovídají přidaným a pomocným proměnným
- složky vektoru $\mathbf{u}' = A^T\mathbf{u} - \mathbf{c}$ (vlastně hodnoty duálních přidaných proměnných) ve sloupcích odpovídajících ostatním proměnným

Interpretace duální úlohy.
Optimální hodnoty duálních proměnných (někdy se jim říká duální nebo stínové ceny) udávají, o kolik se zvýší hodnota primární účelové funkce při maximalizaci (o kolik klesne při minimalizaci), vzrostou-li složky vektoru **b** o jednotku (změny vektoru **b** nesmí být velké, nesmí způsobit překročení meze stability řešení).

Interpretace duálních cen je jednoduchá například u úlohy výrobního programování, tj. stanovení výrobního programu, jsou-li dané ceny výrobků a omezená množství surovin tak, aby bylo dosaženo maximální rentability. Pak duální ceny jsou relativním oceněním surovin.

Poznámka: Při řešení úloh lineárního programování v praxi simplexovou metodou na počítačích, se může stát, že primární úloha přesahuje rozsah použitelnosti příslušného

programu. Příslušná duální úloha může mít rozměry přijatelné. Pak stačí vyřešit tuto úlohu.

3.9 Speciální úlohy lineárního programování - dopravní úloha

V případě speciálních úloh lineárního programování, se místo simplexové metody užívá distribuční metoda. Její reprezentací je dopravní (problém) úloha, označována za nejstarší úlohu lineárního programování. V dopravní úloze jde o přepravu produktu od skupiny dodavatelů ke skupině odběratelů za nejnižší cenu (nebo po nejkratší cestě), viz příklad 1.1 v první kapitole (str. 4).

Přípustné řešení úlohy nazýváme *plán distribuce*. Hledáme tedy *optimální* (minimální) *plán distribuce*.

Dopravní úlohu lze jako každou úlohu lineárního programování řešit simplexovým algoritmem, který je však pro tento typ úloh zbytečně složitý. K určení výchozích řešení se část užívá tzv. metoda severozápadního rohu. Podrobněji o této problematice pojednává literatura [3], [5], [13].

4. ÚLOHY LINEÁRNÍHO PROGRAMOVÁNÍ NA ZÁKLADNÍ ŠKOLE

4.1 Úlohy v prostoru dimenze 2 řešené graficky

Tento způsob řešení nemá v praxi význam, ale je velmi názorný. Grafickou metodu lze užít při řešení nejjednodušších úloh, v nichž se vyskytují dvě proměnné a vlastní omezení jsou vyjádřena nerovnostmi

Z analytické geometrie je známo, že grafickým znázorněním lineární nerovnosti s dvěma proměnnými je polorovina. Nejprve tedy sestrojíme oblast přípustných řešení. Na množině přípustných řešení hledáme maximum (nebo minimum) účelové funkce z. Dosadíme-li za z různé hodnoty, dostáváme soustavu rovnoběžných přímek. Z této soustavy vybereme přímku, která má s množinou přípustných řešení alespoň jeden společný bod a je přitom nejdále od počátku (při hledání maxima) nebo nejblíže počátku (při hledání minima). Pokud existuje alespoň jeden takový bod, odpovídá tento bod optimálnímu řešení.

Uvedeme dále 6 řešených úloh grafickou metodou.
První příklad 4.1 je řešením soustavy nerovnic, a v grafu je vyznačena množina přípustných řešení.
V příkladě 4.2 je znázorněno grafické řešení příkladu 3.1 v prostoru dimenze 2. Jediné optimální řešení je v obrázku vyznačeno bodem X_{opt}.
V příkladě 4.3 existuje jediné optimální řešení, vyznačeno bodem Xopt.
Příklad 4.4 má nekonečné mnoho optimálních řešení, které tvoří body úsečky AB.
Příklad 4.5 nemá konečné optimální řešení, což znázorňuje vyznačená polorovina na obrázku.
Příklad 4.6 má jediné optimální řešení, v obrázku je to bod X_{opt}.

Příklad 4.1 :

1. Znázorněte konvexní množinu C, která je řešením soustavy nerovnic

$x_1 \geq 0$

$-x_1 + x_2 \leq 4$

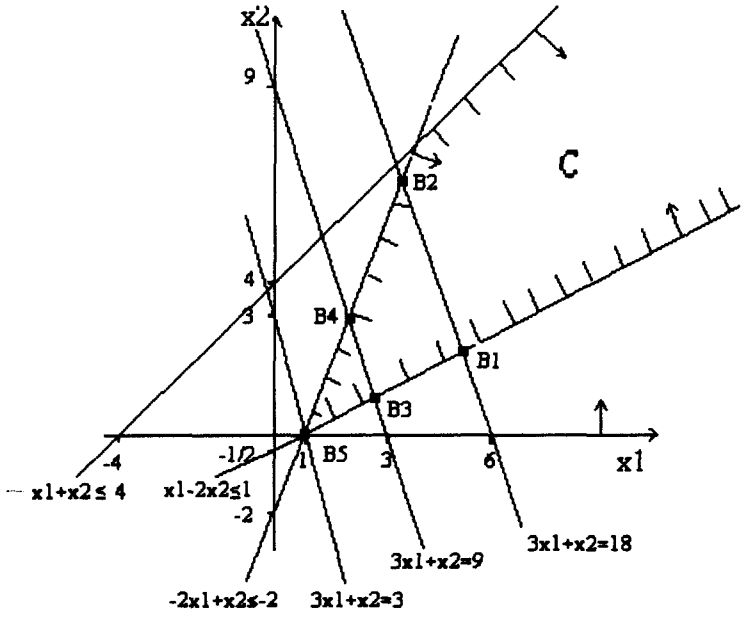
$-2x_1 + x_2 \geq -2$

$x_1 - 2x_2 \leq 1$

2.Najděte množinu bodů, ve kterých má účelová funkce $3x_1 + x_2$ definovaná na C, minimum.

Řešení:

1. V grafu je znázorněna konvexní množina C, která je neomezeným konvexní polyedrem dimenze 2.
2. Jsou tu znázorněny přímky $3x_1 + x_2 = b_i$. Hledáme b_i, při kterém bude průsečík množiny C a účelová funkce $3x_1 + x_2$ minimální. Pro $b_1 = 18$, $b_2 = 9$ jsou průsečíkem množiny a účelové funkce (přímky) úsečky B_1B_2 a B_3B_4. našemu případu určení minima vyhovuje $b_i = 3$, průsečík množiny a účelové funkce je jediný bod $B_5 = (1,0)$, který je krajním bodem množiny C.



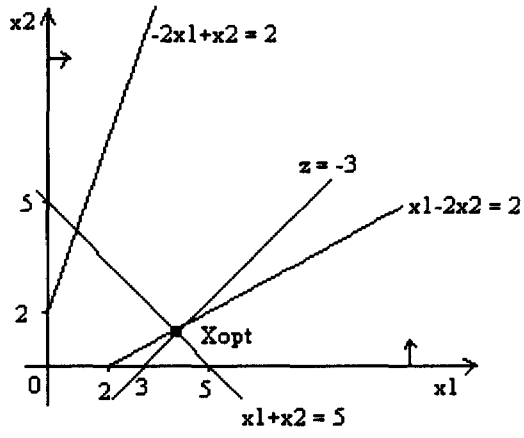
Množina přípustných řešení je oblast C, tj. řešení, která vyhovují omezujícím podmínkám.

Grafické řešení příkladu 3.1 v prostoru dimenze 2:

Příklad 4.2:

Nalézt minimum funkce $z = -x_1 + x_2$, **při omezeních**

$$-2x_1 + x_2 = 2$$
$$x_1 - 2x_2 = 2$$
$$x_1 + x_2 = 5$$
$$x_1 \geq 0$$
$$x_2 \geq 0$$



Množina přípustných řešení je pouze jediný bod.

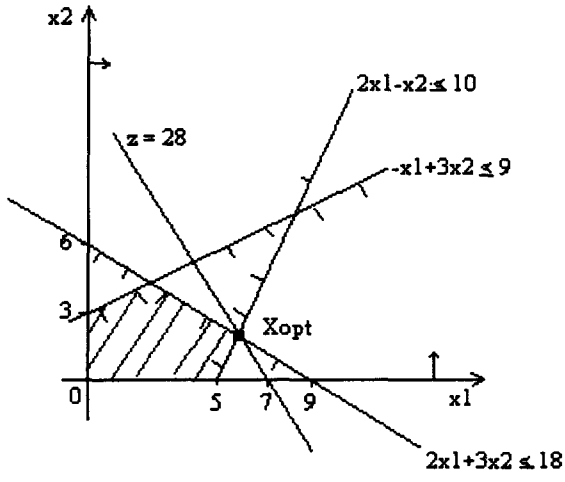
Optimální řešení X_{opt}: $x_1 = 4$, $x_2 = 1$, $z = -3$

Příklad 4.3 :

Nalézt maximum funkce $z = 4\ x_1 + 2\ x_2$ za podmínek:

$$
\begin{aligned}
-\ x_1 + 3\ x_2 &\leq 9 \\
2x_1 + 3\ x_2 &\leq 18 \\
2x_1 - \ x_2 &\leq 10 \\
x_1 &\geq 0 \\
x_2 &\geq 0
\end{aligned}
$$

Grafické řešení úlohy:



Množina přípustných řešení je konvexní pětiúhelník.

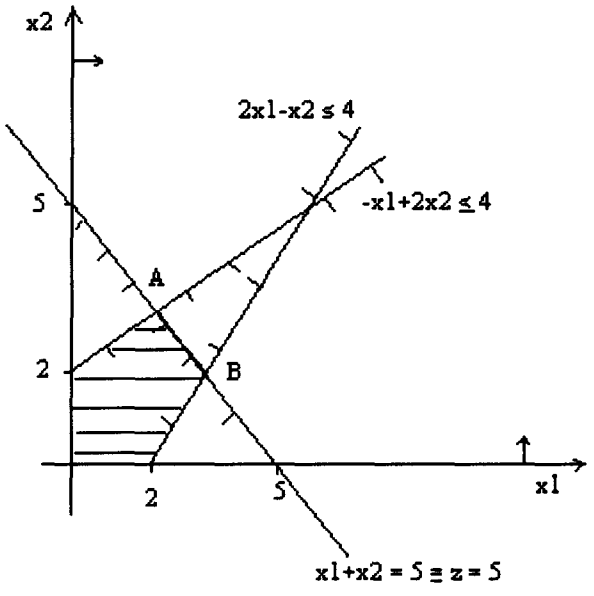
Optimální řešení X_{opt}: $x_1 = 6$, $x_2 = 2$, $z = 28$

Příklad 4.4 :

Nalézt maximum funkce $z = x_1 + x_2$ za podmínek:

$$-x_1 + 2x_2 \leq 4$$
$$2x_1 - x_2 \leq 4$$
$$x_1 + x_2 \leq 5$$
$$x_1 \geq 0$$
$$x_2 \geq 0$$

Grafické řešení:



Množina přípustných řešení je konvexní pětiúhelník.

Optimální řešení :

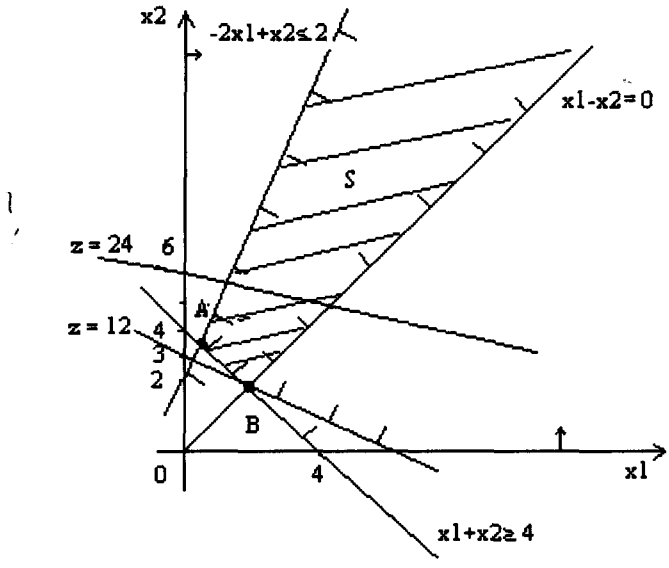
Úloha má nekonečně mnoho optimálních řešení - všechny body úsečky AB, pro které účelová funkce $z = 5$.

Příklad 4.5 :

Nalézt maximum funkce $z = 2x_1 + 4x_2$ za podmínek:

$$x_1 + x_2 \geq 4$$
$$-2x_1 + x_2 \leq 2$$
$$x_1 - x_2 \leq 0$$
$$x_1 \geq 0$$
$$x_2 \geq 0$$

Grafické řešení:



Množina přípustných řešení je oblast S.

Optimální řešení:

Úloha nemá konečné optimální řešení, účelová funkce může na neomezené množině přípustných řešení nabývat libovolně velkých hodnot. V případě, že bychom řešili minimalizační problém, tj. hledali minimum funkce $z = 2x_1 + 4x_2$ při stejných omezujících podmínkách, optimální řešení by odpovídalo bodu B.

Příklad 4.6 :

Určit maximum z funkce $z = x1 + x2$ za podmínek $3x_1 + 5x_2 \leq 15$

$$-x_1 + x_2 \leq 2$$

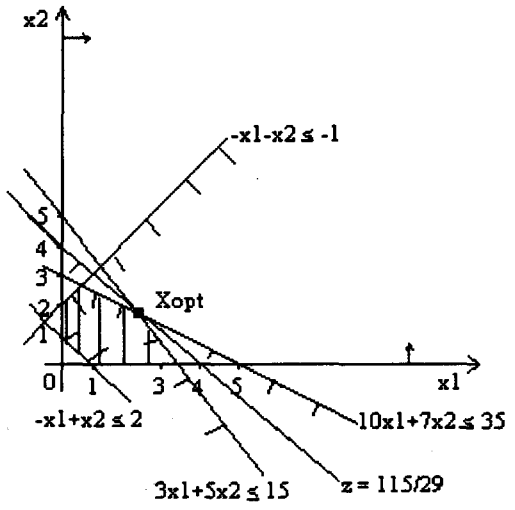
$$10x_1 + 7x_2 \leq 35$$

$$-x_1 - x_2 \leq -1$$

$$x_1 \geq 0$$

$$x_2 \geq 0$$

Grafické řešení:



Množinou přípustných řešení je konvexní šestiúhelník.

Optimální řešení X_{opt}: $x_1 = 70/29$, $x_2 = 45/29$, $z = 115/29$.

5. VYUŽITÍ PROGRAMŮ A POČÍTAČŮ V ÚLOHÁCH LINEÁRNÍHO PROGRAMOVÁNÍ

5.1 Programový systém Matlab

Úlohy s více proměnnými se graficky řeší obtížněji, proto se při rozvoji počítačů vyvinul software k řešení nejenom úloh lineárního programování, o velkém počtu proměnných. Kdysi dávno se numerická matematika počítala velmi neprakticky. K dispozici sice byly algoritmy pro řešení nejrůznějších úloh, ale nebyly moc šikovné na použití, byly ve Fortraně (Fortran je programovací jazyk) a dalším důvodem bylo, aby si člověk nemusel při každé změně počátečních podmínek kompilovat program.

Výpočetní a vizualizační prostředí Matlabu je produktem firmy MATHWORKS, Inc.. Je to moderní nástroj na analýzu a simulaci různých systémů, který v sobě integruje veškeré nástroje potřebné na modelování složitých systémů. Matlab poskytuje svým uživatelům nejenom mocné grafické a výpočetní nástroje ale i rozsáhlé knihovny funkcí spolu s programovacím jazykem čtvrté generace. Matlab umožňuje:

- podporu vícerozměrných polí a uživatelsky definovaných datových struktur
- působivá 2D a 3D grafika, otevřený a rozšiřitelný systém
- objektové programování
- rozšířená podpora řídkých matic
- integrované prostředí pro ladění programů (debugger)

Jádrem Matlabu jsou algoritmy pro operaci s maticemi komplexních čísel. Vektor reálních čísel může reprezentovat i polynomy, může reprezentovat časové řady nebo signály, přičemž Matlab obsahuje funkce pro jejich analýzu, jako je hledání extrémů, střední hodnoty, odchylky, rychlou Fourierovu transformaci a jiné.
Otevřená architektura Matlabu vedla k vzniku knihoven nazývaných toolbox, orientovaných na příslušné vědné a technické obory. Toolboxy jsou nadstavby nad Matlabem, pozůstávají z tzv. M-files, což jsou funkce v Matlabě. Takovéto toolboxy umožňují potom pracovat např. s neuronovými sítěmi, zpracovávat signály, pracovat s

fuzzy logikou, symbolickou matematikou, parciálními diferenciálními rovnicemi, statistikou a mnohými dalšími věcmi.
Knihovny jsou svým rozsahem využitelné prakticky ve všech oblastech lidské činnosti.Za nejsilnější stránku Matlabu je možno považovat mimořádně rychlé výpočtové jádro s optimálními algoritmy. Celý balík obsahuje základní systém (viď níž) a tzv. toolboxy..

Základní systém má 5 částí:

1. jazyk Matlabu - speciální jazyk pro práci s maticemi (chcete používat Matlab = myslet v maticích)
2. operátory, datové typy, funkce pro vstup/výstup
3. working environment - starostlivost o proměnné a pod.
4. práce s grafikou - 2D, 3D, grafy, obrázky, tvorba uživatelského rozhraní a pod.
5. matematické funkce -lineární algebra, diferenciální rovnice, polynomy, matice, goniometrické funkce a pod

Další počítačový systém, který obsahuje algoritmy z Matlabu, je rovněž dostupný na internetu. SIMULINK je program pro simulaci a modelování dynamických systémů, které využívají algoritmy Matlabu pro numerické řešení nelineárních diferenciálních rovnic. Poskytuje uživatelovi možnost rychle a lehce vytvářet modely dynamických soustav ve formě blokových schémat a rovnic. Pomocí Simulinku a jeho grafického editoru je možné vytvářet modely lineárních , nelineárních, v čase diskrétní nebo spojité systémy, jen přesouváním a spájením funkčních bloků pomocí myši.

Použití programového systému Matlab ve výuce, a možnosti použití programového souboru „ Optimalization Toolbox“ programového systému Matlab. Je zde ukázáno použití funkce *lp* k řešení úlohy lineárního programování, dále možnosti vlivu změn jednotlivých parametrů na řešení úlohy.

Příklad 5.1

Čokoládovna vyrábí pět druhů výrobků. Spotřebovává tři základní suroviny: tuk, kakao a cukr, jež jsou k dispozici v omezených množstvích 1500 kg, 300 kg a 450 kg na den. Kapacita strojového zařízení je dostatečná, stejně tak energie, pracovní síly; další zdroje jsou k dispozici v dostatečném množství. Spotřeba surovin na výrobky V_1, V_2, V_3, V_4, V_5 je uvedena v následující tabulce:

Tabulka 5.1 :

	V_1	V_2	V_3	V_4	V_5
Tuk		0,4	0,3	0,6	0,6
Kakao	0,05	0,2	0,1	0,1	
Cukr	0,1	0,2	0,2	0,1	0,2

Odbytové ceny v Kč za 1 kg jsou:

V_1..... 20,-

V_2..... 120,-

V_3..... 100,-

V_4..... 140,-

V_5..... 40,-

Stanovte denní výrobní program takový, aby hodnota výroby v Kč byla maximální.

Označíme-li x_i množství výrobku V_i v kg (i = 1,2,3,4,5), jež bude vyráběno za den, potom úlohu lze převést na nalezení maxima účelové funkce:

$$z = 20x_1 + 120x_2 + 100x_3 + 140x_4 + 40x_5$$

vyhovující nerovnostem:

$0,4x_2 + 0,3x_3 + 06,x_4 + 0,6x_5 \leq 1500$

$0,05x_1 + 0,2x_2 + 0,1x_3 + 0,1x_4 \leq 300$

$0,1x_1 + 0,2x_2 + 0,2x_3 + 0,1x_4 + 0,2x_5 \leq 400$

tuto úlohu řešíme užitím funkce lp z Optimalization Toolboxu programového systému Matlab takto:

```
A = [ 0 0.4 0.3 0.6 0.6
      0.05 0.2 0.1 0.1 0
```

```
0.2 0.2 0.1 0.2]
b = [1500 300 450]'
f = -[20 120 100 140 40]'
lb = [0 0 0 0 0]'
x0 = lp(f,A,b,lb)
```

Řešením dostáváme

```
x0 =
1.0e+003 *
   0.0000
   0.0000
   1.0000
   2.0000
        0
```

Optimální výrobní program je tedy vyrábět 1000 kg výrobku V_3 a 2000 kg výrobku V_4.

Nyní si ukážeme vliv změny jednotlivých parametrů na řešení úlohy. Měníme-li omezení pro množství některé ze surovin, např. pro tuk, získáme užitím následujících příkazů hodnoty účelové funkce pro měnící se omezení pro množství tuku při nezměněných hodnotách pro ostatní suroviny včetně grafického znázornění.

```
x0=lp(f,A,b,lb);
for i=1:61
   j=(i-1)*50;
   b(1)=j;
   x0=lp(f,A,b,lb);
   y(i)=-f'*x0;
   x(i)=j;
end
plot(x,y);
```

Poznámka : Všechny jmenované obrázky z příkladu se nacházejí v přílohách.

Následující tabulka 5.2 a obrázek 5.1 udává hodnoty účelové funkce (tj. celkovou hodnotu denní produkce) pro jednotlivé hodnoty omezení množství tuku při nezměněném omezení pro ostatní suroviny. Obrázek 5.2 udává odpovídající řešení, tj. množství jednotlivých výrobků.

Tabulka 5.2 :

Maximální množství tuku	Hodnota denní produkce	Maximální množství tuku	Hodnota denní produkce	Maximální množství tuku	Hodnota denní produkce
0	90 000	1050	300 000	2050	436 666
50	100 000	1100	310 000	2100	440 000
100	110 000	1150	320 000	2150	443 333
150	120 000	1200	330 000	2200	446 666
200	130 000	1250	340 000	2250	450 000
250	140 000	1300	350 000	2300	450 000
300	150 000	1350	360 000	2350	450 000
350	160 000	1400	366 666	2400	450 000
400	170 000	1450	373 333	2450	450 000
450	180 000	1500	380 000	2500	450 000
500	190 000	1550	386 666	2550	450 000
550	200 000	1600	393 333	2600	450 000
600	210 000	1650	400 000	2650	450 000
650	220 000	1700	406 666	2700	450 000
700	230 000	1750	413 333	2750	450 000
750	240 000	1800	420 000	2800	450 000
800	250 000	1850	423 333	2850	450 000
850	260 000	1900	426 666	2900	450 000
900	270 000	1950	430 000	2950	450 000
950	280 000	2000	433 333	3000	450 000
1000	290 000				

Podobně lze provést výpočet pro změny omezení pro zbývající suroviny. Následující tabulka 5.3 a obrázek 5.2 udává hodnoty účelové funkce (celkovou hodnotu denní produkce) pro jednotlivé hodnoty omezení množství kakaa při nezměněném omezení pro ostatní suroviny. Obrázek 5.4 znázorňuje odpovídající řešení, množství jednotlivých výrobků.

Tabulka 5.3:

Maximální množství kakaa	Hodnota denní produkce
0	90 000
50	150 000
100	200 000
150	250 000
200	300 000
250	350 000
300	380 000
350	390 000
400	390 000
450	390 000
500	390 000

Následující tabulka 5.4 a obrázek 5.5 zobrazují hodnoty účelové funkce (celkovou hodnotu denní produkce) pro jednotlivé hodnoty omezení množství cukru při nezměněném omezení pro ostatní suroviny. Obrázek 5.6 udává odpovídající řešení, množství jednotlivých výrobků.

Tabulka 5.4 :

Maximální množství cukru	Hodnota denní produkce
0	0
50	70 000
100	140 000
150	210 000
200	280 000
250	350 000
300	360 000
350	370 000
400	380 000
450	380 000
500	380 000

Obrázek 5.7 znázorňuje , jak lze pomocí grafiky v Matlabu demonstrovat, jak se mění hodnota účelové funkce, měníme-li současně omezení pro dvě suroviny (tuk a kakao), při nezměněném omezení pro surovinu třetí. K jeho vytvoření byla použita následující posloupnost příkazů:

```
x=0:100:3000
y=x
for i=1:31,
for j=1:31,
   b(1)=x(i);
   b(2)=y(j);
   x0=lp(f,A,b,lb);
   z(i,j)=-f'*x0;
end
end
mesh(x,y,z)
```

Z předchozích úvah je vidět, že zvýšení množství cukru již neumožní zvýšení celkové hodnoty produkce, pokud není možno zvýšit množství ostatních používaných surovin, naproti tomu zvyšování množství tuku umožňuje zvyšovat hodnotu produkce. Obrázky 5.2, 5.4, 5.6 umožňují demonstrovat, jak se mění množství jednotlivých výrobků při zvyšování množství jedné ze surovin vstupujících do výroby, jestliže je maximalizována hodnota produkce.

5.2 Programový systém LPS

LPS - program pro řešení úloh lineárního programování, byl vyvinut s využitím simplexové metody, matematického programování, řešení soustav lineárních nerovností a rovnic, vícekriteriální (vektorové) optimalizace a diskrétní proměnné.
Tento software pro počítače typu PC je určen odborníkům, kteří řeší úlohy lineární optimalizace. Dobře slouží i při výuce studentů vysokých a středních škol, proto ho zde uvádíme jako ukázku.

Vlastnosti systému:

- pro vyhledání optima účelové funkce je použita modifikovaná simplexová metoda
- řeší úlohy do 6000 sloupců a 4000 řádků
- nenulové prvky matice jsou uloženy na vnějším médiu (disku)
- počet nenulových prvků matice je omezen pouze velikostí volné paměti na disku
- proměnné mohou být i bivalentní nebo diskrétní
- obsahuje čtyři metody pro vícekriteriální rozhodování
- výsledné řešení je uvedeno v přehledné tabulce
- umožňuje analýzu řešení z hlediska účelové funkce
- přesnost nalezeného řešení lze ověřit zpětným dosazením do matice
- ovládá se jednoduchým výběrem příkazu z nabídky
- pracuje interaktivně - zavedenou úlohu lze modifikovat přímo z monitoru
- umožňuje sestavit velkou úlohu z dílčích úloh

Příkazy a data mohou být vkládány z klávesnice nebo čteny z předem připraveného ASCII souboru. Výsledky ukáže obrazovka nebo vytiskne tiskárna, případně mohou být

uloženy do textového souboru. Aktualizace modelu (úpravy sloupců, omezení, účelové funkce) lze provádět interaktivně a úpravy nenaruší současné bázické řešení.

Programový soubor LPS je určen odborníkům, kteří v různých institucích řeší úlohy lineární optimalizace. Dále může posloužit i při výuce studentů středních a vysokých škol. Příručku je nutné chápat jako manuál popisující jednotlivé příkazy systému - předpokládá se jistý stupeň znalosti výpočetní techniky a odpovídající znalosti o úlohách lineárního programování. Úlohy z oblasti lineární optimalizace (matematické programování, lineární programování) byly jedny z prvých, které byly na číslicových počítačích řešeny. Existovaly obsáhlé programové systémy pro sálové počítače, které řešily tyto úlohy. V oblasti počítačů typu PC existuje také široká škála programů. Tento programový soubor je jedním z nich.

Systém je vyřešen tak, že nemá pevná omezení velikosti úlohy, kterou může řešit. Omezení je dáno pouze velikostí operační paměti, kapacitou vnější diskové paměti a možnostmi adresace spojitého pole. Operační paměť je dynamicky přidělena podle velikosti úlohy a podle předdeklarovaných nebo zvolených délek vyrovnávacích pamětí pro spolupráci s vnější pamětí. LPS pracuje pod operačními systémy DOS 3.1 a vyššími. Systém LPS je řízen jednoduchými čtyřznakovými příkazy. Tyto příkazy je možné přímo vybírat z tabulky příkazů. První písmeno příkazu určuje širší činnost. Každá činnost je v jednom sloupci příkazů. Činností je celkem šest:

O ... otevírání souborů,
N ... nastavování režimu práce,
C ... čtení,
P ... provádění výpočtu a pomocných činností,
Z ... zápis úlohy nebo vybraných prvků,
A ... aktualizace stávajícího modelu.

Některé příkazy vyžadují parametr. Parametrem rozumíme jeden nebo dva údaje, které upřesňují požadovanou činnost. Příkazy pro čtení a aktualizaci vyžadují data. Příkazy a data mohou být vkládány z klávesnice, nebo čteny z předem připraveného textového souboru. Výsledky může ukázat obrazovka nebo vytisknout tiskárna, případně mohou být uloženy do textového souboru. Aktualizace modelu (úpravy sloupců, omezení, účelové funkce) lze provádět kdykoliv, a tyto úpravy nenaruší dosažené bázické řešení.

Nyní je uveden příklad zpracovaný LPS softwarem.

Příklad 5.2 :

Kamion - program optimalizace nakládky.

Program navrhne umístit hranolovité výrobky (deskové přířezy, krabice, bedny, jednotlivé kusy) na palety. Tyto palety pak rozmístí na korbu dopravního prostředku (kamionu, vagonu, kontejneru) tak, aby délka nákladu byla minimální (palety mohou být pouze fiktivní). Výrobky jsou definovány třemi rozměry: výška (tloušťka) výrobku a základna výrobku (dva rozměry). Výška palety je dána součtem výšek výrobků na paletě. Výrobky jsou na palety umisťovány tak, aby prostorové využití bylo optimální. Předpokládá se, že palety mohou být vyrobeny v požadovaných rozměrech, nebo se vybírají ze zadaného zásobníku rozměrů palet.

Jsou možné tyto kombinace:

- všechny výrobky se uloží na palety vyrobené v rozměrech, které jsou výsledkem optimalizace (na míru)
- je k dispozici zásobník rozměrů palet a palety, které vyhovují tolerančním podmínkám
- jsou použity všechny výrobky, uloženy jen na palety které jsou v zásobníku rozměrů.

Dále se předpokládá, že nakládka jednotlivých druhů výrobků na paletu se děje po vrstvách (stozích), dokud nejsou uloženy všechny výrobky, nebo dokud paleta není plná. V každé vrstvě jsou stejné výrobky. Jsou-li palety vyráběny v potřebných rozměrech, jsou výrobky na paletě uloženy jedním směrem. V případě použití palet ze zásobníku mohou být výrobky na paletě uložené různými směry a to vždy tak, aby procentní zaplnění palety bylo maximální (ze všech palet v zásobníku).

Kritéria optimálního řešení jsou minimální počet palet, rovnoměrnost výšky palet a minimální délka nákladu. Můžeme si zvolit jejich pořadí. Jako doplňující podmínku lze zadat délku korby. Program řeší tyto problémy:

a) minimální počet palet pro jednu zakázku
b) minimální délku nakládky na korbě
c) rozmístění jednotlivých palet na korbě
d) pořadí jízd ještěrky při nakládání palet

Vstupní data

Zásobník rozměrů palet je textový soubor. V prvním řádku má jmenovku, která se přepíše do výstupního souboru a slouží k identifikaci. Na dalších řádcích jsou parametry palet v pořadí: základna palety, šířka palety. Tento soubor nemusí existovat v případech, kdy jsou všechny palety vyráběné na míru podle výsledků optimalizace.
Údaje o požadované nakládce (zakázce, výrobě) musí být v jiném textovém souboru. První řádek souboru je jmenovka, která se přepíše do výstupního souboru a slouží k identifikaci. Na dalších řádcích jsou údaje o požadované nakládce. V každém řádku jsou data v tomto pořadí: tloušťka (výška), délka, šířka výrobku, požadovaný počet výrobků, typ výrobku, popis výrobku.

Popis výsledků.

Výsledky optimalizace jsou zapisovány do dvou souborů.
V prvním souboru jsou přepsány nastavené vstupní údaje. Řádky vstupních dat jsou očíslovány (Pořadí) a čísla řádků jsou uvedena v tabulce ULOŽENÍ VÝROBKU NA JEDNOTLIVÉ PALETY. U čísla palety je znak, který je použit v grafickém zobrazení pro tuto paletu. Jsou-li na jedné paletě i jiné druhy výrobků, jsou uvedeny na dalších řádcích. Ve sloupcích Počty je uveden celkový počet výrobků na jedné paletě a počet výrobků v jedné vrstvě. Ve sloupcích Toleran jsou skutečné tolerance ve směru základny a šířky. Objem palet je počítán jako součet objemů všech palet (musí být větší nebo roven objemu výrobků). Obal palety je počítán jako součet obvodové plochy a plochy palety. Délka nákladu (maximum) je dána jako součet maximálních délek palet v jednotlivých řadách (jízdách ještěrky). Tato délka nebude překročena ani když při ukládání nedodržíme vypočtené pořadí jízd a pořadí palet na ještěrce. Pořadí jízd je však nutné dodržet v případech, kdy některá paleta je extrémně dlouhá a následující jízdy doplňují volnou plochu. Využitá plocha je poměr Plocha všech palet / (Šířka korby *Délka nákladu)*100. Při grafické optimalizaci se hledá taková varianta uložení palet na ložnou plochu, při které se minimalizuje délka nákladu a nevyužitá plocha. Nebere se v potaz způsob naložení. Rozměry palet jsou zaokrouhleny.

NASTAVENÉ HODNOTY:

Maximální výška palety = 180 cm

Maximální základna palety= 150 Minimální základna= 80 cm

Maximální hloubka palety = 100 Minimální hloubka = 40 cm

Tolerance ve směru základny palety = 12 cm

Tolerance ve směru šířky palety = 10 cm

Šířka korby= 240 cm Délka korby= 99999 cm

Minimalizuji počet palet.

Kontejner1

PŘEPRAVOVANÉ VÝROBKY:

Poř.	Výška	A	B	Počet	Typ	Objem	Popis
1	170	50	60	20	1	10.200	ledničky
2	30	40	50	130	2	7.800	televizor 1
3	40	50	60	80	2	9.600	televizor 2
4	70	120	60	50	4	25.200	skříně 1
5	15	160	70	20	4	3.360	skříně 2

Součty : 300 56.160 m3

Varianta: 4

Kontejner1

ULOŽENÍ VÝROBKU NA JEDNOTLIVÉ PALETY

	Palety				*	Výrobky							
Číslo	Zákl	Šířka	Výška	Počet*	Poř	Výšk	A	B	Počty		Toleran		Popis
1 A	120	100	170	5 *	1	170	60	50	4	4	0	0	ledničky
2 B	100	40	180	10 *	2	30	50	40	12	2	0	0	televizor 1
3 C	120	100	160	5 *	3	40	60	50	16	4	0	0	televizor 2
4 D	120	120	140	12 *	4	70	60	120	4	2	0	0	skříně 1
5 E	140	160	150	1 *	5	15	70	160	20	2	0	0	skříně 2
6 F	120	120	70	1 *	4	70	60	120	2	2	0	0	skříně 1
7 G	100	40	150	1 *	2	30	50	40	10	2	0	0	televizor 1

Celkový počet palet = 35

Délka nákladu (maximum) = 16.800 m

Minimální výška palety = 0.700 m

Objem palet = 56.160 m3

Spotřeba obalu = 257.560 m2

Využitá plocha nákladu = 92.659 %

NAKLÁDÁNÍ JEŠTĚRKOU PALETY NA KORBĚ

Jízda Uložení na korbu Počet .....

1 Odzadu vyrovnat vlevo : 3 . B B E .

2 Zpředu vyrovnat vpravo: 3 . A A B .

3 Odzadu vyrovnat vlevo : 3 . B A A .

4 Zpředu vyrovnat vpravo: 3 . A C B .

5 Odzadu vyrovnat vlevo : 3 . B C C .

6 Zpředu vyrovnat vpravo: 3 . C C B .

7 Do středu : 2 . D D .

8 Do středu : 2 . D D .

9 Do středu : 2 . D D .

10 Do středu : 2 . D D .

11 Do středu : 2 . D D .

12 Do středu : 2 . D D .

13 Odzadu vyrovnat vlevo : 4 . B B B F .

14 Do středu : 1 . G .

.....

Ve druhém souboru jsou některé údaje z výsledné tabulky přepsány v jiném pořadí tak, aby vyhovovaly výrobě jednotlivých palet. Počty přířezů na paletě jsou navíc uvedeny ve stozích a ke každé paletě je k vůli identifikaci přidáno prvních šest znaků jmenovky.

Příklad 5.3 :

Optimální dělení materiálu.
Z existujícího skladu materiálu (tyčí, hutních profilů, stavebního dřeva, tabulí, svitků) máme vyrobit (nařezat, nastříhat) množinu pravoúhlých přířezů zadaných rozměrů.
K tomu účelu definujme:
výtěžnost výroby = přířezy (celková délka nebo plocha) / materiál vydaný ze skladu (délka nebo plocha). Často se stává, že po řezbě zůstávají zbytky. Je účelné, aby využitelnost těchto zbytků byla maximální.
Úkolem je nalézt takový řezný plán, který má maximální výtěžnost výroby a použitelnost zbytků. Při řešení je třeba respektovat technické podmínky výroby (tloušťku řezu, možnost otáčení materiálu při řezbě, rozměry zbytků pro znovu zařazení do skladu, rozměry řezné stolice apod.).

Programy pro řešení těchto problémů:
REZPROFW - pro řezání tyčí / profilů
REZPLANW - pro řezání tabulí / svitků

Vstupní data obsahují informace o skladu (rozměry, počty), následuje zadání požadované výroby (opět rozměry a počty).
Výstupem z programu je plán řezby. Výroba je rozdělena do jednotlivých kroků. V každém kroku se řežou přířezy stejných rozměrů. Zdrojem je buď materiál ze skladu, nebo zbytek z předešlých kroků.

Možnosti použití
Řezání skleněných tabulí, dřevotřískových desek, stříhání hutního materiálu (profilů), tabulí plechu (ze svitků) a podobné činnosti. Výpočet optimálního množství prvotního materiálu (tabulí) pro určitou zakázku.

Závěr

Cílem diplomové práce bylo ukázat aplikace lineárního programování v teoretické a praktické rovině. Práce je komponována tak, aby se v dané problematice orientoval i nezainteresovaný čtenář. Členění je postaveno na logické návaznosti , přičemž se vychází z matematického základu. Část věnovaná matematice obsahuje elementární poznatky ve formě definic a vět. Z důvodu jednoduchosti se v textu neuvádí všechny důkazy vět, předpokládá se jistá úroveň znalostí z lineární algebry a matematické analýzy.

Větší pozornost je kladena na lineární programování. Pomocí simplexové metody jsme schopni vyřešit daný problém lineárního programování. Užíváme algoritmu simplexové metody. Jelikož existuje mnoho variant simplexové metody, jsou v diplomové práci uvedeny také další varianty řešení různých typů úloh.

Za částí věnovanou teoretickým poznatkům, následuje grafické zpracování úloh. Protože jsou všechny tyto příklady řešeny v prostoru dimenze dva, můžou být aplikovány i na základní škole, se zřetelem na úroveň poznatků. Tato kapitola se nezabývá postupem a metodami řešení, slouží pouze jako vizuální prvek.

V případě speciálních úloh lineárního programování simplexovou metodu výpočtu nahrazuje distribuční metoda.

Při rozvoji počítačů vznikal také software pro řešení úloh lineárního programování, a to z důvodu usnadnění výpočtů úloh s více proměnnými. Jelikož se úlohy o velkém počtu proměnných řeší obtížně, je v dnešní době dostupných mnoho počítačových systémů, pro výpočet úloh lineárního programování. V systému Matlab je možno kromě řešení úloh lineárního programování možno provádět výpočty numerické matematiky, analyzovat, simulovat a modelovat různé systémy, používat objektové programování i grafické prostředí. V příkladě vyřešeném systémem Matlab jsou uvedeny ukázky užití příkazů algoritmu na výpočet jednotlivých měnících se podmínek úlohy lineárního programování. Výsledky řešených zadaných hodnot příkladu jsou zapsány v tabulkách, při každé změně podmínek. V příloze se pak může čtenář obeznámit s grafickým znázorněním jednotlivých výpočtů zpracovaných v tomto systému.

Další program pro řešení úloh lineárního programování dostupný na internetu je programový systém LPS. Využívá simplexovou metodu, avšak je vytvořen spíše pro odborníky.

Literatura

[1] Horák, P. : Algebra a teoretická aritmetika I. 2. vydání. Brno: Vydavatelství MU, 1994. 196 s.

[2] Korda, B. a kol. : Matematické metody v ekonomii. 1. vydání. Praha: SNTL, 1967. 604 s.

[3] Šmarda, B. : Lineární programování. Vydání 1/2. Praha: SPN, 1983. 151 s.

[4] Švrček, J. : Lineární programování v úlohách. 1. vydání. Olomouc: Vydavatelství univerzity Palackého, 1995. 103 s.

[5] Plesník, J. – Dupačová, J. – Vlach, M. : Lineárne programovanie. 1. vydání. Bratislava: Alfa, 1990. 320 s.

[6] Sekanina, M. – Boček, L. – Šedivý, J. : Geometrie I. 1. vydání. Praha: SPN, 1986. 200 s.

[7] Samek, J. : Lineární programování v příkladech. 3. vydání. Praha: SPN, 1978, 251 s.

[8] Rychetník, L. – Zelinka, J. – Pelbauerová, V. : Sbírka příkladů z lineárního programování. 1. vydání. Praha: SNTL, 1968. 316 s.

[9] Bláha, K. : Lineární programování v dopravě a plánování. 1. vydání. Praha: Nakladatelství dopravy a spojů, 1961. 176 s.

[10] Habr, J. : Lineární programování (Výklad pro ekonomy). 2. přepracované vydání. Praha: SNTL, 1960. 192 s.

[11] Grygarová, L. : Úvod do lineárního programování. 1. vydání. Praha: SPN, 1975. 189 s.

[12] Dantzig, G. B. : Lineárne programovanie a jeho rozvoj. 1. vydání. Bratislava: SVTL, 1996. 704 s.

[13] Gass, S. I. : Lineárne programovanie. 2. přepracované vydání. Bratislava: Nakladatelství ALFA, 1972. 400 s.

[14] Gál, T. : Úvod do lineárního programování. 1. vydání. Praha: SZN, 1968. 186 s.

[15] Bauer, L. Matlab. In XIV. Vědecké kolokvium o řízení osvojovacího procesu zaměřené na didaktické problémy modelvání a simulace (sborník příspěvků). I. Vyškov: VVŠ PV, 1996

[16] JAMRICHOVÁ, Eva. Študijné materiály: MatLab[online]. c2005, Fakulta matematiky, fyziky a informatiky UK, Katedra matematickej analýzy a numerickéj matematiky, 2005 [cit. 22. červenec 2005]. Dostupný z WWW: http://hore.dnom.fmph.uniba.sk/personal/jamrichova/MatLab/matlab_win.html

[17] HROMÁDKA, Tomáš. MATLAB, Octave, Scilab[online].c2005[cit. 22. červenec 2005]. Seminár. MATLAB, Octave, Scilab. Dostupný z WWW: http://www.manualy.sk/seminar/Papers98/matlab/

[18] JANUŠKA, Karel. Programy pro PC[online].c2005[cit. 23. červenec 2005]. Programový systém LPS. Dostupný z WWW: http://cmp.felk.cvut.cz/~pisa/Public/ST_matlab.html

[19] BERKA, Milan. Operační výzkum II : Operační výzkum doktorandské studium 1999/2000 II. Ročník – INFO[online]. C1996,[cit. 27. červenec 2005]. Matematické programování. Dostupný z WWW: http://home.eunet.cz/berka/matempro.htm

Klíčové slova

Matice

Konvexní množina

Lineární programování

Simplexová metoda

Optimální řešení

Přípustné řešení

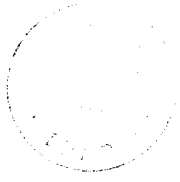
Degenerace

Dopravní úloha

Matlab

Simulink

LPS



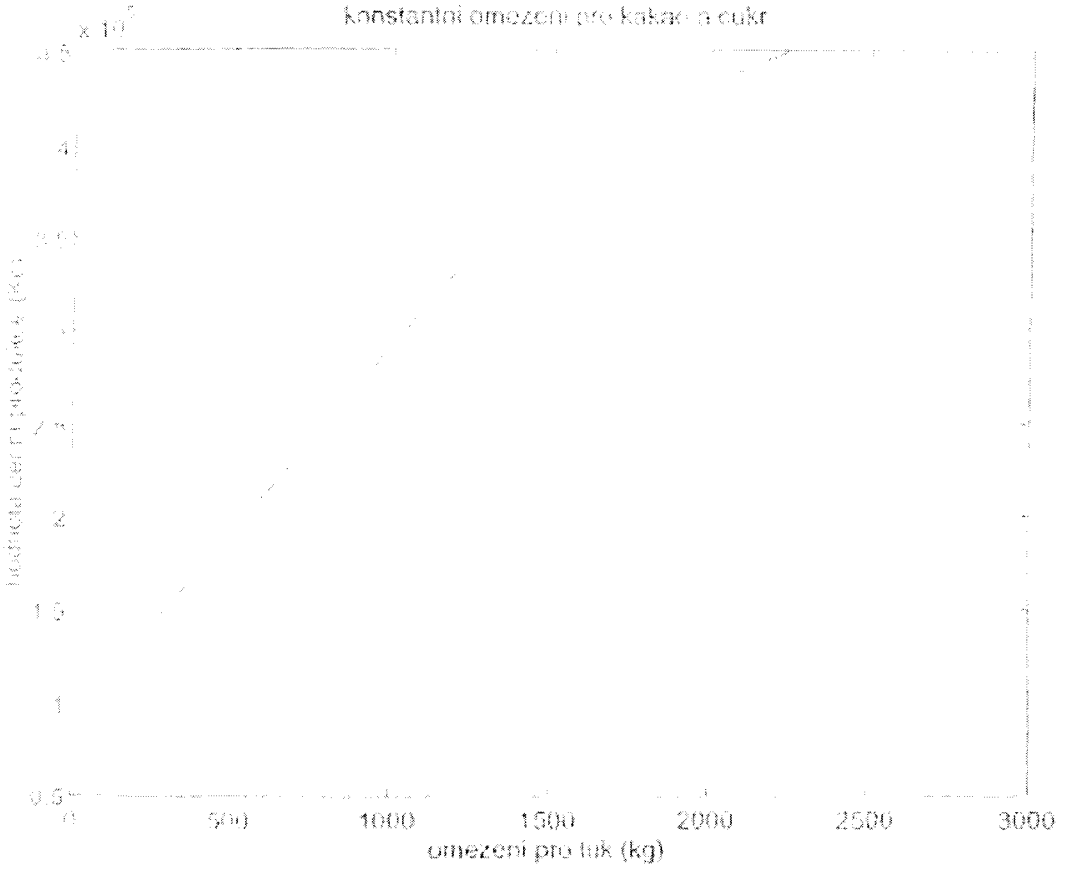
Resumé

Teorie lineárního programování obsahuje metody a algoritmy pro nalezení optimálního řešení. Cílem je určit a vytvořit výpočetní postupy, které řeší zformulované úlohy lineárního programování. Krok za krokem vede práce od matematických základů, přes metody řešení obecné úlohy lineárního programování, speciální úlohy a grafické řešení úloh až k ukázkám počítačových systémů vyvinutých pro řešení úloh matematického programování, jehož součástí je i lineární programování. Jsou zde uvedeny jak ukázky řešení jednotlivých problémů, tak i celá řešení konkrétních příkladů. Základní partie jako simplexová metoda , jsou popsány podrobně, zatímco jiné speciální úlohy lineárního programování mají více informativní charakter. V práci je upřednostněn teoretický přístup, praktické ukázky následují od čtvrté kapitoly. Téma lineárního programování je ukázáno teoreticky i prakticky, s ohledem na použití nejen pro odbornou veřejnost.

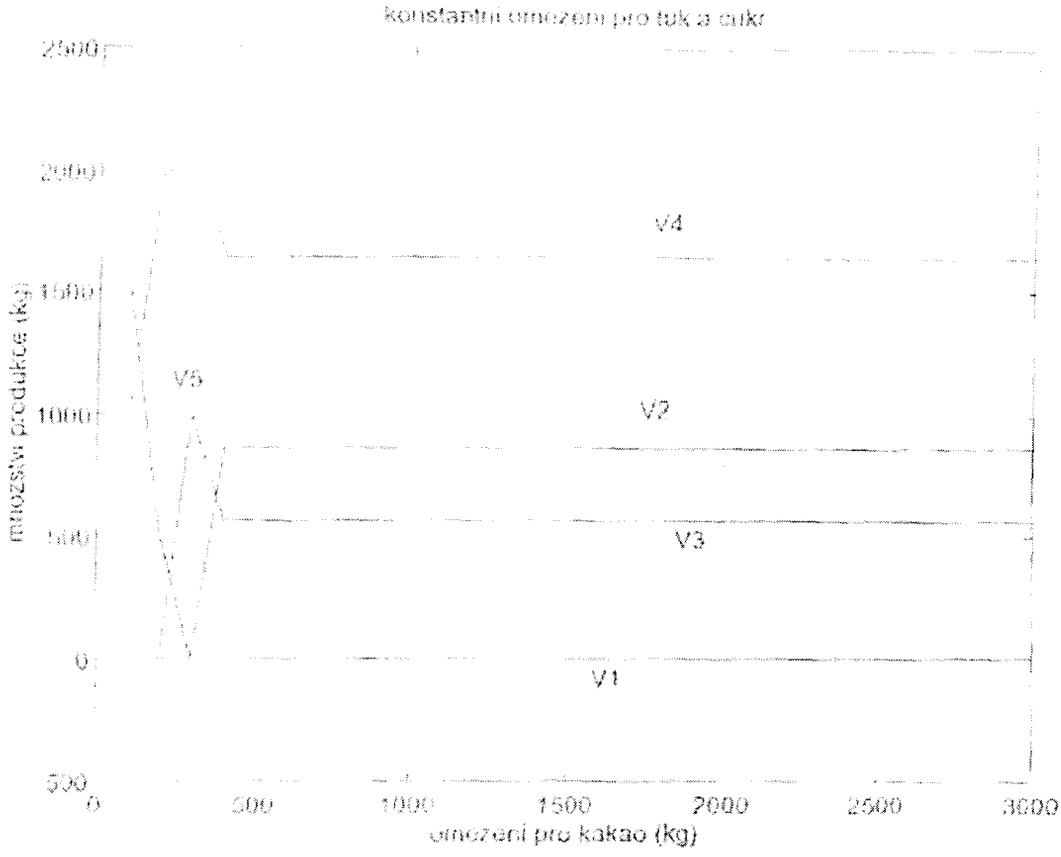
The theory of the linear programming includes the methods and algorithms for finding optimal solution. The object is to determine and create the computing progresses, which solve formulated problems of the linear programming. The tesis leads from a mathematical basis, through the methods of solutions of the linear programming, special problems and the graphical solution of problems to demonstration of computer systems, which are developed for problems of mathematical programming with its part linear programming. There are samples of the solution of each problem and also the solutions of concrete examples. The basic part as a simplex method is described in a detail. While other special problems of linear programming are more informative, in foreground is a theoretical ingress, the practical demonstrations begins from the fourth chapter. There is presented the linear programming in the practical and theoretical way, not only for public speciality.

Přílohy

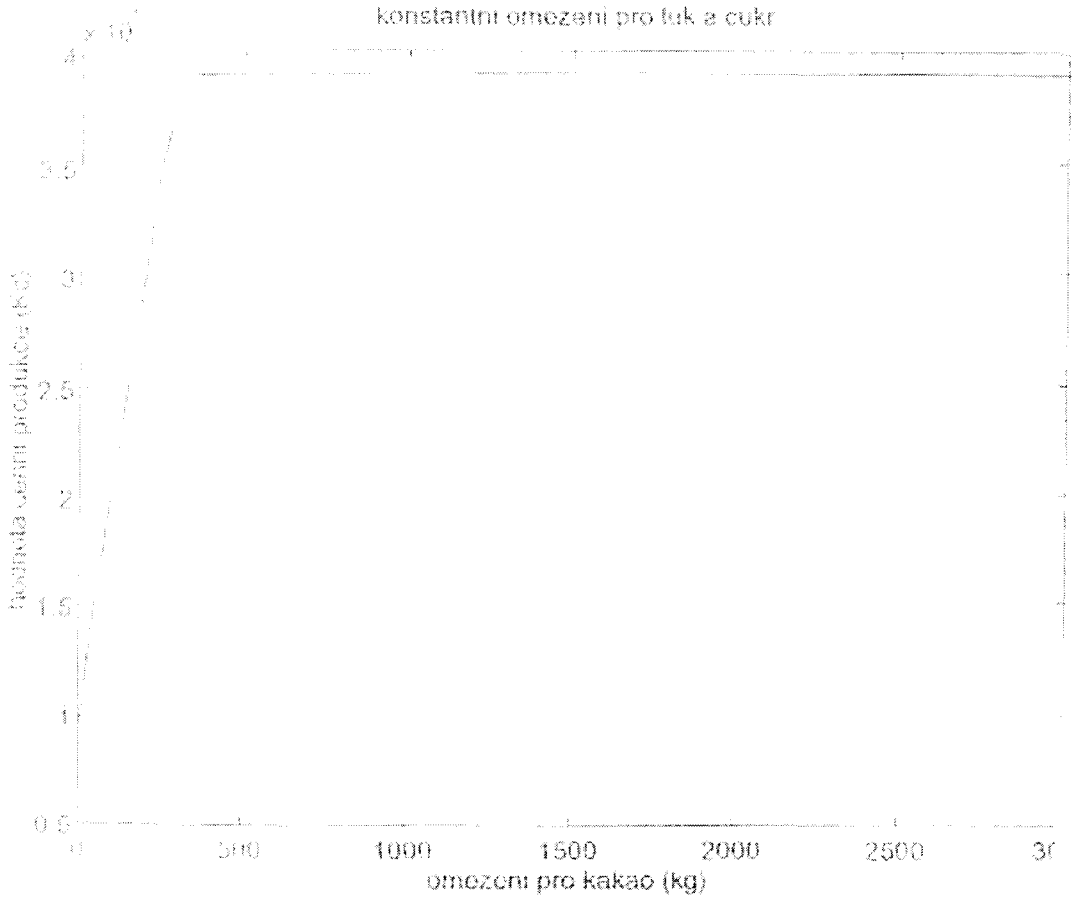
Obrázek 5.1 :



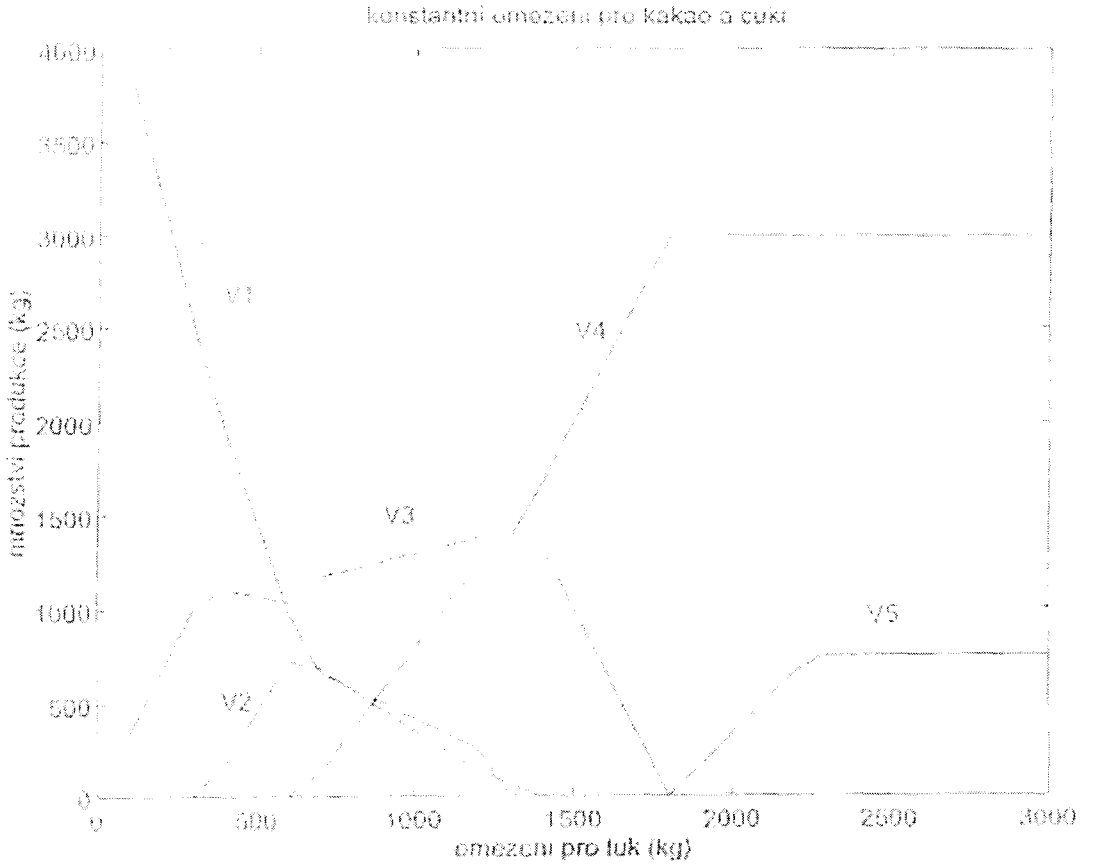
Obrázek 5.2 :


konstantní omezení pro tuk a cukr
2500
2000
1500
1000
500
0
množství produkce (kg)
V4
V5
V2
V3
V1
0
500
1000
1500
2000
2500
3000
omezení pro kakao (kg)

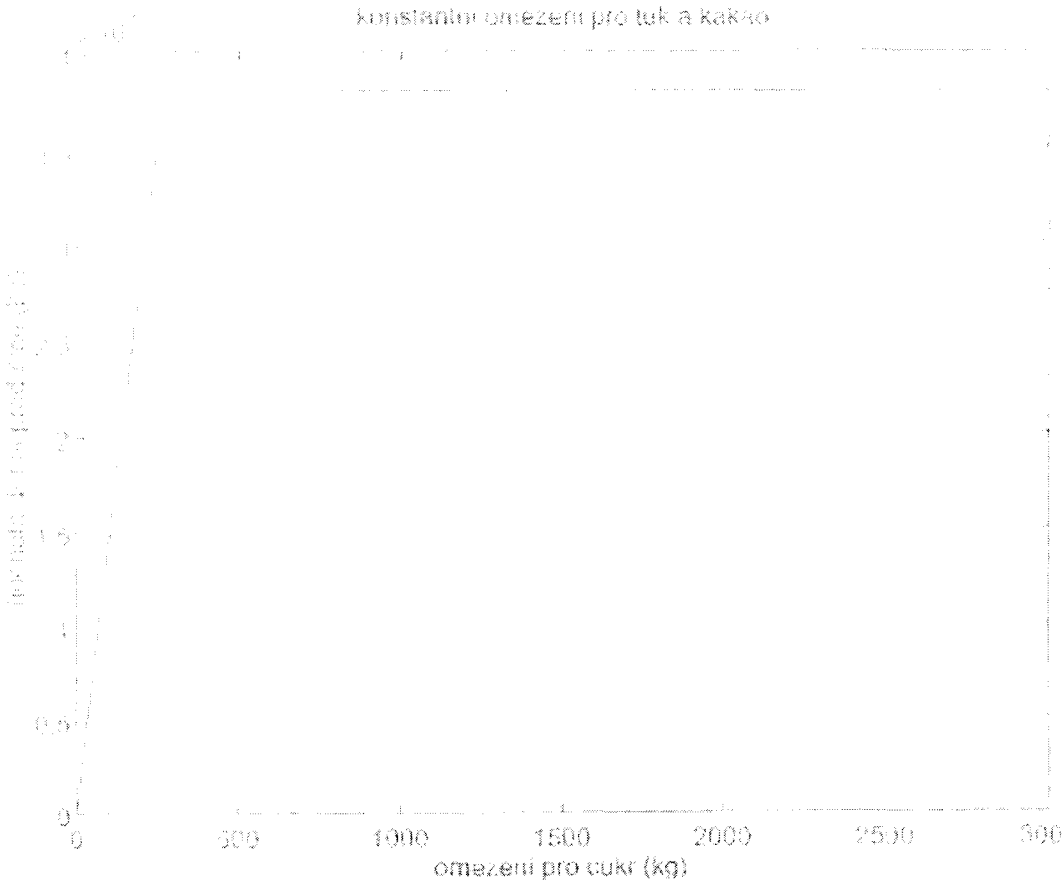
Obrázek 5.3 :


konstantní omezení pro tuk a cukr
x 10^4
4
3.5
3
2.5
2
1.5
1
0.5
0
500
1000
1500
2000
2500
omezení pro kakao (kg)

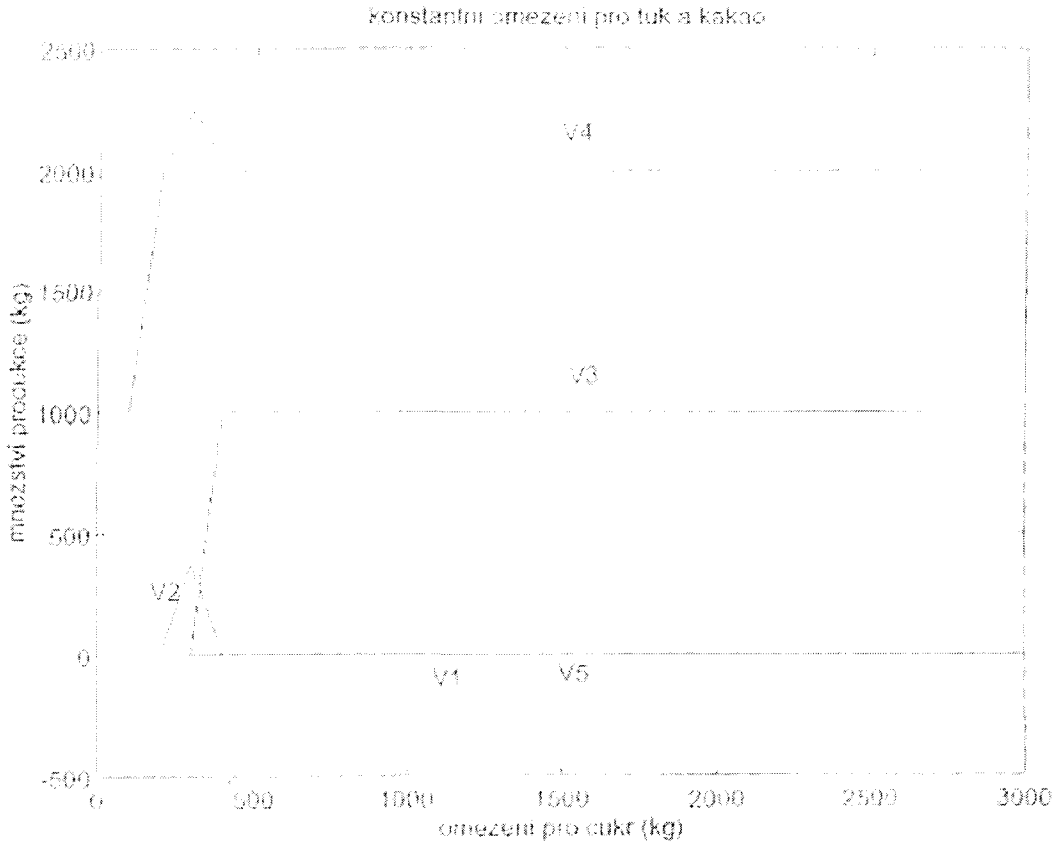
Obrázek 5.4 :


konstantní omezení pro kakao a cukr
množství produkce (kg)
4000
3500
3000
2500
2000
1500
1000
500
0
V1
V2
V3
V4
V5
0
500
1000
1500
2000
2500
3000
omezení pro luk (kg)

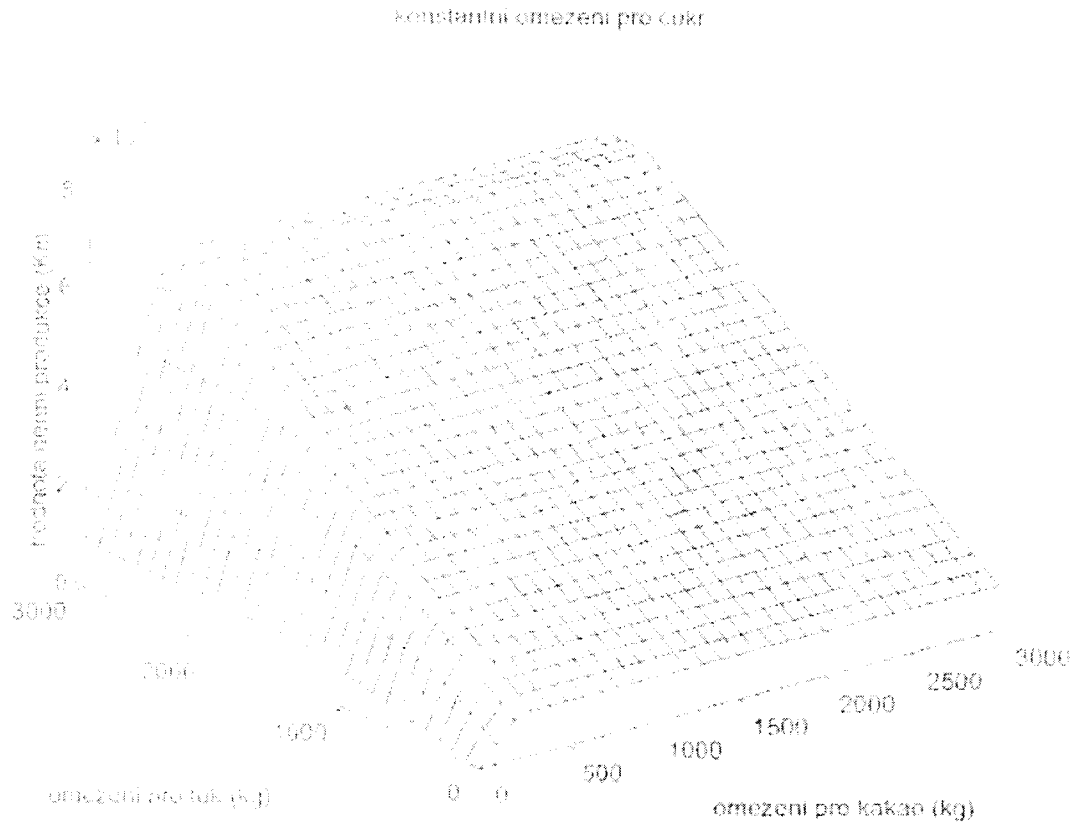
Obrázek 5.5 :


konstantní omezení pro tuk a kakao
0
500
1000
1500
2000
2500
omezení pro cukr (kg)

Obrázek 5.6 :


konstantní omezení pro tuk a kakao
množství produkce (kg)
omezení pro cukr (kg)
V1
V2
V3
V4
V5
2500
2000
1500
1000
500
0
-500
0
500
1000
1500
2000
2500
3000

Obrázek 5.7 :


konstantní omezení pro cukr
3000
2500
2000
1500
1000
500
0
omezení pro kakao (kg)

Anotace

Malá, Hana: Lineární programování, diplomová práce. Brno, MU 2005, s. 64

Cílem práce je objasnit a ukázat metody lineárního programování. Teoreticky i prakticky jsou zde uvedeny algoritmy pro výpočet úloh lineárního programování. Pro lepší srozumitelnost se v práci věnuji matematickému základu i grafickým ukázkám.

The objective of the thesis is to explain and to demonstrate the methods of the linear programming. There is presented an algorithm for the computation of the linear programming problem in the theoretical and practical way. For a better comprehensibility, I go in for a mathematical basis and graphical samples in the thesis.

Klíčová slova: Matice, Konvexní množina, Lineární programování, Simplexová metoda, Optimální řešení, Přípustné řešení, Degenerace, Dopravní úloha, Matlab, Simulink, LPS