Latvijas Universitāte
Fizikas un matemātikas fakultāte

Inese Bula

LINEĀRĀ PROGRAMMĒŠANA

LEKCIJU KONSPEKTS — 2008

SATURS

NODAĻA 1

LINEĀRĀS PROGRAMMĒŠANAS UZDEVUMS

1.1 KURSA PRASĪBAS

KURSA ANOTĀCIJA

Lineārā programmēšana kā patstāvīga disciplīna izveidojās iepriekšējā gadsimta četrdesmitajos-piecdesmitajos gados, t.i., apmēram tajā pašā laikā, kad parādījās pirmie datori. Jau daudzus gadu desmitus, attīstoties datoriem, paralēli attīstās arī lineārā programmēšana. Jo jaudīgāki kļūst datori, jo lineārās programmēšanas pielietojumu sfēra paplašinās. Šobrīd lineārās programmēšanas metodes izmanto gan ražošanā, lauksaimniecībā, transportā un celtniecībā, gan veselības aizsardzībā un kara lietās, gan psiholoģijā un socioloģijā.

REZULTĀTI

Pēc sekmīgas šī kursa apguves students pārzinās pamatpieņēmumus, ar kādiem operē lineārā programmēšana. Tas ļaus saprātīgi lietot lineārās programmēšanas algoritmus dažādu teorētisku un praktisku mērķu sasniegšanai.

PRASĪBAS KREDĪTPUNKTU IEGŪŠANAI

Studiju kursa gala atzīme tiek aprēķināta no semestra laikā rakstīto mazo kontroldarbu atzīmēm (jāuzraksta 3 kontroldarbi) - 10% un gala eksāmena (rakstisks pārbaudījums par teoriju un uzdevumiem par apgūto kursu) atzīmes - 90%.

LITERATŪRA
Mācību pamatliteratūra
1. S.A. Ašmanovs. Lineinoje programmirovanije, Maskava, "Nauka", 1981 (krievu val.).
2. C.H. Papadimitriou, K. Steiglitz. Combinatorial optimization, Prentice-Hall, 1982 (vai krievu val. Maskava, "Mir", 1985).
3. H.A. Taha. Operations Research. An Introduction. Macmillan, 1982 (vai krievu val. Maskava, "Mir", 1985, divas daļas).
4. D. Kļaviņš. Optimizācijas metodes ekonomikā I, II, Datorzinību Centrs, 2003.
Papildliteratūra
5. J.N. Franklin. Methods of mathematical economics, New York, 1980.
6. W. Krelle, H. P. Kuenzi. Lineare Programmierung, Zuerich, 1959.
7. I. Brīvers. Lineārā programmēšana, Banku Augstskola, Rīga, 2001.
8. V.G. Karmanov. Matematičeskoje programmirovanije, Maskava, "Nauka", 1986 (krievu val.).
9. S.R. Xačatrjan, M.V. Pinegina, V.P. Bujanov. Metodi i modeli rešenija ekonomičeskix zadač, Maskava, "Ekzamen", 2005 (krievu val.).
Periodika, interneta resursi
10. *http://en.wikipedia.org/wiki/Linear_programming*

1.2 KAS IR LINEĀRĀ PROGRAMMĒŠANA?

Matemātiskā programmēšana ir viena no matemātikas pamatsastāvdaļām, kura pēta vairāku argumentu funkciju nosacītā ekstrēma atrašanas metodes. Matemātiskā programmēšana atkarībā no risināmo uzdevumu veida savukārt iedalās lineāraja, nelineārajā, diskrētajā, dinamiskajā, ģeometriskajā, stohastiskajā programmēšanā. Klasiskajā matemātiskās analīzes kursā arī tiek apskatīts jautājums par funkcijas nosacītā ekstrēma atrašanu. Taču prakse ir parādījusi, ka daudzos gadījumos klasiskās metodes ir grūti lietojamas vai ir nepietiekamas. Reizē ar datoru parādīšanos ir tikuši izveidoti matemātiskie modeļi, kas ļauj abstrahēties no pētāmās parādības un dod iespēju veikt pētījumus ar matemātikas palīdzību. Idealizācija un abstrakcijas nedrīkst aiziet pārāk tālu no uzdevuma satura, lai konstruētais modelis nepazaudētu būtiskas modelējamā objekta īpašības. Lielā vairumā gadījumu pirmie tuvinātie modeļi reālajam objektam ir tādi, kuros sakarības starp mainīgajiem tiek aprakstītas lineāri. Līdz ar to varam precizēt lineārās programmēšanas jēdzienu. *Lineārā programmēšana* nodarbojas ar vairāku argumentu *lineāras* funkcijas nosacīto ekstrēmu atrašanas problēmām tādās kopās, kuras tiek aprakstītas ar *lineārām* vienādībām un/vai nevienādībām (t.i., *lineāriem* ierobežojumiem).

Lineārās programmēšanas attīstībā galvenie nopelni ir Krievijas un ASV zinātniekiem. Viens no pirmajiem pētījumiem lineārās programmēšanas virzienā ir 1939.gadā iznākušais krievu matemātiķa L.V.Kantoroviča darbs "Matemātiskās metodes ražošanas organizēšanā un plānošanā". 1947.gadā Dž. Dancigs (*G.B. Dantzig*) tika izstrādājis universālu lineārās programmēšanas uzdevumu atrisināšanas metodi — simpleksa metodi. Šo zinātnieku veikums ir novērtēts ar Nobela prēmiju ekonomikā. Termins "lineārā programmēšana" parādījies četrdesmito gadu beigās, kad 1949.gadā tika publicēta Danciga grāmata *"Programming in Linear Structures"*. Bet plašākai sabiedrībai ir pazīstama Danciga grāmata *"Linear Programming and Extensions"*, izdevniecība Princeton University Press, 1963, kuras tulkojums krievu valodā iznācis 1966.gadā izdevniecībā Progress, Maskavā.

Lai nonāktu pie praktiskas problēmas atrisinājuma, ir jānoiet noteikts secīgu soļu skaits, varam precizēt šos soļus vai etapus:

1. Uzdevuma nostādne.

2. Apskatāmā objekta vai procesa saturīga verbāla modeļa apraksts. Šajā etapā notiek objekta darbības mērķu formalizācija; iespējamo darbību noskaidrošana, kas varētu ietekmēt mērķi; ierobežojošo darbību apraksta izveidošana.

3. Matemātiskā modeļa konstruēšana, t.i., konstruētā verbālā modeļa pieraksts tādā formā, kura pētīšanu var veikt ar matemātiskiem paņēmieniem

4. Uzdevuma atrisināšana, kurš formulēts uz matemātiskā modeļa bāzes.

5. Iegūto rezultātu pārbaude. Tiek noskaidrota atbilstība starp iegūtajiem teorētiskajiem rezultātiem un praksē konstatētajiem. Var veikt pētījumus, kāda ir modelī neietverto faktoru iedarbība un izdarīt attiecīgas korekcijas.

6. Iegūtā atrisinājuma realizācija praksē.

Kaut arī lineārās programmēšanas ietvaros mēs saskarsimies ar modelēšanas dažādajiem etapiem, tomēr pamatā nodarbosimies ar ceturto etapu. Mūsu mērķis ir apgūt lineārās programmēšanas pamatjēdzienus, pamatproblemātiku, risināšanas algoritmus un analīzi.

1.3 LINEĀRU MODEĻU PIEMĒRI

Šajā apakšnodaļā apskatīsim dažus salīdzinoši vienkāršus piemērus, kuri atspoguļo kādu reālu situāciju. Apskatītos modeļus var klasificēt kā ekonomiski-matemātiskus, jo tajos būs ekonomisks rādītājs — izmaksas. Kāpēc tieši ekonomiskie modeļi? Izrādās, ka lineārā programmēšana ieguvusi īpašu lietojumu kā pētīšanas instruments tieši ekonomiski-matemātiskajos modeļos.

Lielāka uzmanība tiks pievērsta tieši pirmajam modelim — diētas uzdevumam. Tas darīts ar nolūku, lai lasītājs tiktu iepazīstināts ar uzdevumu formalizācijas metodiku.

Diētas uzdevums

Uzdevums ir sastādīt pēc iespējas ekonomiskāku (t.i., lētāku) ēdienkarti, kas apmierinātu noteiktas medicīniskās prasības. Šāds uzdevums var rasties tur, kur ir nepieciešams pabarot lielu cilvēku skaitu, piemēram, slimnīcās, armijā, skolā, u.t.t. Tāda problēma var rasties arī dažādu katastrofu gadījumā.

Pieņemsim, ka mums ir zināmi n dažādi pieejamie produkti (maize, sāls, cukurs, sviests, piens, kartupeļi, u.c.), kurus apzīmēsim ar burtiem F_1, F_2,..., F_n. Pie reizes ir zināms, cik daudz vitamīnu ir šajos produktos, cik minerālvielu, tauku, olbaltumvielu, ogļhidrātu un kāda ir kaloritāte. Pieņemsim, ka šādu vielu kopumā ir skaitā m un apzīmēsim šīs vielas ar N_1, N_2,..., N_m. Pieņemsim, ka katram produktam F_i, $i = 1, 2, ..., n$, ir zināmi tā vielu medicīniskie rādītāji, t.i., skaitliskie lielumi par vienā produkcijas vienībā (kilogramā vai gramā) esošajiem vielu daudzumiem. Tādā gadījumā var sastādīt tabulu, kura atspoguļo produktos esošo vielu daudzumus:

	F_1	F_2	$\dots$	F_j	$\dots$	F_n
N_1	a_{11}	a_{12}	$\dots$	a_{1j}	$\dots$	a_{1n}
N_2	a_{21}	a_{22}	$\dots$	a_{2j}	$\dots$	a_{2n}
$\dots$	$\dots$	$\dots$	$\dots$	$\dots$	$\dots$	$\dots$
N_i	a_{i1}	a_{i2}	$\dots$	a_{ij}	$\dots$	a_{in}
$\dots$	$\dots$	$\dots$	$\dots$	$\dots$	$\dots$	$\dots$
N_m	a_{m1}	a_{m2}	$\dots$	a_{mj}	$\dots$	a_{mn}

Šīs tabulas elementi veido matricu, kurai ir m rindiņas un n kolonnas. Apzīmēsim šo matricu ar A un nosauksim par *uztura matricu.*

Pieņemsim, ka mēs esam sastādījuši plānu $x = (x_1, x_2, ..., x_n)$ noteiktam laika posmam (piemēram, nedēļai). Tas nozīmē, ka mēs plānojam katram cilvēkam nedēļā x_1 vienības (kilogrammus vai gramus) produkta F_1, x_2 vienības produkta F_2,..., x_n vienības produkta F_n. Varam izrēķināt, kādu daudzumu vitamīnus, taukus, olbaltumvielas, u.c., dabūs cilvēks paredzētajā laika posmā. Viela N_1 šajā plānā ir šādā daudzumā

$$a_{11}x_1 + a_{12}x_2 + \ldots + a_{1n}x_n,$$

jo produkta F_1 x_1 vienībās (saskaņā ar uztura matricu) ir atrodamas $a_{11}x_1$ vienības vielas N_1; šim daudzumam tiek pievienotas $a_{12}x_2$ vienības vielas N_2, ko dod x_2 vienības produkta F_2 u.t.t. Analoģiski var noteikt pārējo vielu N_i daudzumus plānā $(x_1, x_2, \ldots, x_n)$.

Pieņemsim, ka ir noteiktas medicīniskās prasības, cik daudz nedēļas laikā katram cilvēkam nepieciešams patērēt uzturā atbilstošās vielas N_i, $i = 1, 2, ..., m$. Piemēram, vitamīnam C ir jābūt ne mazākam kā norādītais daudzums, kaloriju kopskaitam arī jābūt vismaz norādītajā daudzumā. Šīs prasības aprakstīsim ar vektoru $b = (b_1, b_2, \ldots, b_m)$, kur vektora b komponente i norāda minimālo nepieciešamo vielas N_i daudzumu plānā. Tas nozīmē, ka vektora x koordinātām x_i ir jāapmierina šāda ierobežojumu sistēma:

$$\begin{array}{c} a_{11}x_1 + a_{12}x_2 + \ldots + a_{1n}x_n \geq b_1, \\ a_{21}x_1 + a_{22}x_2 + \ldots + a_{2n}x_n \geq b_2, \\ \ldots \\ a_{m1}x_1 + a_{m2}x_2 + \ldots + a_{mn}x_n \geq b_m. \end{array} \tag{1.3.1}$$

Kaut arī uzdevuma jēga ir skaidra, tomēr īpaši jāatzīmē, ka visi mainīgie x_1, x_2,..., x_n ir nenegatīvi; formalizācijas procesā tas ir svarīgi, jo datoram, kas rēķinās uzdevumu, vispārīgā gadījumā tas ir jānorāda. Tāpēc ierobežojumiem (1.3.1) tiek pievienotas nevienādības

$$x_1 \geq 0,\ x_2 \geq 0,\ \ldots,\ x_n \geq 0. \tag{1.3.2}$$

Skaidrs, ka jebkuram plānam, kuru vēlamies pasludināt par nedēļas ēdienkarti, ir jāapmierina nevienādības (1.3.1) un (1.3.2). Bet tādu plānu var būt bezgalīgi daudz. Kuru no tiem izvēlēties, ir jāizdomā noteikts kritērijs. Un te palīgā ārstam var nākt grāmatvedis.

Pieņemsim, ka produktu F_1, F_2,..., F_n cenas ir atbilstoši c_1, c_2,..., c_n. Tādējādi visa plāna $x = (x_1, x_2, ..., x_n)$ izmaksas var aprēķināt ar izteiksmi

$$c_1x_1 + c_2x_2 + \ldots + c_nx_n. \tag{1.3.3}$$

Diētas uzdevumu gala variantā var formulēt šādi: starp visiem vektoriem $x = (x_1, x_2, ..., x_n)$, kas apmierina ierobežojumus (1.3.1) un (1.3.2), ir jāatrod tāds, ar kuru izteiksmes (1.3.3) vērtība būtu vismazākā.

Tālāk varam doto uzdevumu pierakstīt matemātiski ērtākā veidā. Atceroties, kā reizina matricu ar vektoru, varam sistēmas (1.3.1) kreiso pusi pierakstīt kā Ax, kur A ir uztura matrica. Pieņemsim, ka $u = (u_1, u_2, \ldots, u_m)$ un $v = (v_1, v_2, \ldots, v_m)$ ir divi vienāda garuma vektori. Rakstīsim $u \geq v$, ja $u_1 \geq v_1$, $u_2 \geq v_2$,..., $u_m \geq v_m$. Tādā gadījumā nevienādību sistēmu (1.3.1) var noformēt šādi:

$$Ax \geq b.$$

Pieraksts $x \geq 0$ nozīmēs, ka visas vektora $x = (x_1, x_2, ..., x_n)$ koordinātas ir nenegatīvas. Visbeidzot izteiksmi $c_1x_1 + c_2x_2 + \ldots + c_nx_n$ var pierakstīt kā vektoru $c = (c_1, c_2, \ldots, c_n)$ un x skalāro reizinājumu $< c, x >$. Izmantojot šos apzīmējumus, diētas uzdevumu var pārformulēt šādā matricu-vektoru formā:

$$\begin{aligned} &\min < c, x > \\ &Ax \geq b, \quad x \geq 0. \end{aligned} \tag{1.3.4}$$

Tātad mēs esam pietiekoši reālu uzdevumu pierakstījuši stingrā matemātiskā formā, kas ļauj izmantot matemātiskās metodes. No otras puses, formulās (1.3.1)-(1.3.4) pilnībā esam abstrahējušies no uzdevuma jēgas un, ja noskaidrosies, ka, sastādot modeli (1.3.4), ir izlaisti kaut kādi no satura viedokļa svarīgi momenti, tad nekāda matemātika mūs neglābs no iegūtajiem bezjēdzīgajiem rezultātiem. Skaidrs, ka modelis (1.3.4) neatspoguļo visas tās reālā uzdevuma puses, kuras modelim vajadzētu nomainīt. Piemēram, medicīniskie ierobežojumi par nepieciešamo vielas N_i daudzumu ēdienkartē var būt norādīti ne tikai, cik daudz šai vielai jābūt, bet cik lielu daudzumu nedrīkst pārsniegt. Šo papildus prasību nav grūti ievērot, pie tam modeļa (1.3.4) veids pārāk neizmainīsies. Bet vēl jau ir arī citi uzdevuma aspekti — ir ievēroti ārsta un ekonomista prasības un viedokļi, bet netiek ievērota paša cilvēka gaume. Var teikt pat vairāk, modelī (1.3.4) netiek ietverta prasība, lai ēdienkartes ēdieni būtu patiešām ēdami.

1945.gadā Dž. Štiglers (*G.Stigler*) tika publicējis konkrēta diētas uzdevuma rezultātus ar konkrētiem skaitļiem. Viņš bija apskatījis 77 produktu veidus, ietvēris 9 vielas (vitamīnu A, olbaltumvielas, ogļhidrātus, u.c.). Tika meklēta viena cilvēka diēta veselam gadam. Ar simpleksa metodi iegūtais rezultāts ietvēra 9 produktus: kviešu miltus, kukurūzu, iebiezināto pienu, augu eļļu, speķi, liellopu aknas, kāpostus, kartupeļus un spinātus. Optimālā diēta, kas sastādīta no minētajiem produktiem, maksāja 39 dolārus 67 centus tā laika cenās (domāts 1939.gads), bija pilnīgi bezgaršīga un tāpēc neēdama. Tanī pašā laikā diēta, ko bija saastādījis ārsts-dietologs un kura bija ēdama, maksāja 115 dolārus.

Neskatoties uz negatīvo pieredzi, nevajadzētu tūlīt domāt, ka sastādītais modelis nekam nav derīgs. Var modeli papildināt ar nosacījumiem, kas ierobežo katra produkta daudzumu, un ar nosacījumiem, kādi produkti ir lietojami kopā un kādi nē.

Eksistē vairākas citas sadzīviskas situācijas, kurās sasatādītais modelis (1.3.4) ir lietojams bez kādām īpašām izmaiņām. Piemēram, šis modelis sekmīgi tika izmantots, lai atrastu minimālo izmaksu ēdienkarti liellopiem (publicēts 1951.gadā, autors *F. V. Wangh, Journal of Farm Economics*). Zināmi sekmīgi lineārās programmēšanas risinājumu lietojumi tādu naftas produktu izgatavošanā, kuriem jāapmierina noteiktas tehniskas prasības (dots oktantskaitlis, cik viela ir gaistoša, u.t.t.) un kuri ir paši lētākie.

Transporta uzdevums

Pieņemsim, ka m dažādās atrašanās vietās S_1, S_2,..., S_m tiek ražota homogēna produkcija (ogles, cements vai kāda cita produkcija), pie tam vietā S_i tiek saražotas a_i šīs produkcijas vienības. Saražotā produkcija tiek izmantota n citās vietās Q_1, Q_2,..., Q_n, pie tam vietā Q_j tiek izmantotas b_j produkcijas vienības. Ir nepieciešams sastādīt tādu pārvadājumu plānu no vietām S_i, $i = 1, 2, ..., m$, uz vietām Q_j, $j = 1, 2, ..., n$, lai apmierinātu pieprasījumu pēc dotās produkcijas un minimizētu transporta izmaksas.

Pieņemsim, ka vienas produkcijas vienības (piemēram, 1 tonnas ogļu) pārvadāšana no vietas S_i uz vietu Q_j izmaksā c_{ij}. Izdarot vēl vienu būtisku pieņēmumu — pārvadājumu izmaksas ir tieši proporcionālas pārvadājamās produkcijas daudzumam, varam uzskatīt, ka, vedot x_{ij} produkcijas vienības no S_i uz Q_j, transporta izmaksas ir $c_{ij}\, x_{ij}$.

Par pārvadājumu plānu nosauksim tādus skaitļus (x_{ij}), $i = 1, 2, ..., m$, $j = 1, 2, ..., n$, kuri apmierina nosacījumus:

$$\begin{aligned} &\sum_{j=1}^{n} x_{ij} = a_i, \quad \sum_{i=1}^{m} x_{ij} = b_j, \\ &x_{ij} \geq 0, \quad i = 1, 2, ..., m, \quad j = 1, 2, ..., n. \end{aligned} \tag{1.3.5}$$

Vienādojumu (1.3.5) saturiskā jēga ir šāda: ja izvēlēts plāns (x_{ij}), tad no vietas S_i uz visām vietām Q_j tiek izvadāta produkcija daudzumā $\sum_{j=1}^{n} x_{ij}$, kuram vajadzētu būt tieši vienādam ar a_i. Savukārt vietā Q_j no visām vietām S_i tiek kopumā piegādāta produkcija daudzumā $\sum_{i=1}^{m} x_{ij}$, tam ir jāsakrīt ar b_j. Pie izvēlētā pārvadājumu plāna (x_{ij}) transporta izdevumi būs vienādi ar

$$\sum_{j=1}^{n} \sum_{i=1}^{m} c_{ij} x_{ij}. \tag{1.3.6}$$

Transporta uzdevumu var formulēt šādi: starp visiem iespējamajiem plāniem (x_{ij}), kas apmierina nosacījumus (1.3.5), jāatrod tāds, kas minimizē (1.3.6).

Transporta uzdevums (un tā modifikācijas) bija viens no pirmajiem lineārās programmēšanas uzdevumiem, kas tika plaši risināts bijušajā Padomju Savienībā. 1959.gadā tika risināts uzdevums par homogēnu kravu pārvadāšanu ar mērķi samazināt pārvadājumu vidējo ceļu garumu. Iegūtais rezultāts ļāva par 11,3% samazināt ceļa garumu.

Ražošanas plānošanas uzdevums

Šajā piemērā apskatīsim vienu konkrētu uzņēmumu, kuram ir jāsastāda ražošanas plāns. Uzņēmumam, ražojot n veidu produkcijas G_1, G_2,..., G_n, jāizmanto m resursi R_1, R_2,..., R_m. Par G_j ražošanas tehnoloģiju sauc vektoru (a_{ij}), $i = 1, 2, ..., m$, kas rāda, kāds daudzums resursa R_i ir nepieciešams, lai saražotu vienu vienību produkcijas G_j. Šos vektorus var izvietot tā saucamajā *tehnoloģijas matricā*:

	G_1	G_2	...	G_j	...	G_n
R_1	a_{11}	a_{12}	...	a_{1j}	...	a_{1n}
R_2	a_{21}	a_{22}	...	a_{2j}	...	a_{2n}
...	...	...	...	...	...	...
R_i	a_{i1}	a_{i2}	...	a_{ij}	...	a_{in}
...	...	...	...	...	...	...
R_m	a_{m1}	a_{m2}	...	a_{mj}	...	a_{mn}

Šī matrica A parāda uzņēmuma tehnoloģiskās iespējas.

Pieņemsim, ka doti resursu R_i, $i = 1, 2, ..., m$, ierobežojumi b_i, tos varam pierakstīt kā *resursu vektoru* $b = (b_1, b_2, ..., b_m)$. Par *ražošanas plānu* sauc vektoru $x = (x_1, x_2, ..., x_n)$, kurš norāda, kāds daudzums produkcijas G_1, G_2,..., G_n tiks saražots.

Modelī tiek izdarīts pieņēmums, ka ražošanas tehnoloģijas process ir lineārs, t.i., resursu izlietojums ir tieši proporcionāls produkcijas izlaides apjomam. Precīzāk, lai saražotu produkcijas G_j izlaidi daudzumā x_j vienības, nepieciešams iztērēt šādu resursu vektoru $(a_{1j}x_j, a_{2j}x_j, ..., a_{mj}x_j)$. Tādā gadījumā iztērēto resursu daudzums, kas nepieciešams, lai izpildītu plānu $x = (x_1, x_2, ..., x_n)$, ir aprakstāms ar vektoru

$$\begin{pmatrix} a_{11}x_1 + a_{12}x_2 + \ldots + a_{1n}x_n \\ a_{21}x_1 + a_{22}x_2 + \ldots + a_{2n}x_n \\ \ldots \\ a_{m1}x_1 + a_{m2}x_2 + \ldots + a_{mn}x_n \end{pmatrix}$$

vai matricu formā ar vektoru Ax. Resursu ierobežošanas nosacījums ir pierakstāms ar nevienādību $Ax \leq b$. Tas nozīmē, ka pie dota resursu vektora b var tikt realizēts jebkurš tāds prpdukcijas ražošanas plāns, kurš apmierina nosacījumus $Ax \leq b$ un $x \geq 0$. Tāds vektors x var nebūt viens vienīgs. Parādās iespēja izvēlēties kaut kādā nozīmē labāko plānu. No ekonomiskā viedokļa viens no pamatoptimizācijas nosacījumiem ir maksimālo ienākumu gūšana. Ja ir zināmas produkciju G_1, G_2,..., G_n cenu vektors $c = (c_1, c_2, ..., c_n)$, tad ražošanas plānošanas uzdevums ir matemātiski formulējams šādi:

$$\begin{gathered} \max < c, x > \\ Ax \leq b, \quad x \geq 0. \end{gathered} \tag{1.3.7}$$

Kaut arī modelis (1.3.7) atspoguļo noteiktas realitātes prasības, tas tomēr ir stipri idealizēts. Tajā netiek ietverts tāds ražošanā svarīgs faktors kā laiks. Modelī tiek pieņemts, ka visi resursi dotajā momentā ir pieejami. Tātad esam izslēguši no modeļa ražošanas dinamiku un resursu piegāžu ritmiskumu. Modelī netiek ietverti arī dzīvā darbaspēka izdevumi un daudzi citi rādītāji, kas pilnībā atspoguļotu reālo ražošanas procesu.

1.4 LINEĀRĀS PROGRAMMĒŠANAS UZDEVUMU DAŽĀDĀS FORMAS UN TO EKVIVALENCE

Neskatoties uz atšķirīgajām saturiskajām situācijām, kas tika aprakstītas iepriekš, atbilstošajiem ekstrēma matemātiskajiem uzdevumiem ir daudzas kopējas iezīmes. Visos šajos uzdevumos nepieciešams maksimizēt vai minimizēt vairāku argumentu lineāru funkciju. Pie tam mainīgo argumentu ierobežojumi ir lineāras nevienādības vai arī lineāras vienādības. Tiesa gan, nevienādības var būt uz dažādām pusēm — vienā gadījumā ierobežotajām izteiksmēm jābūt mazākām par kādu konstanti, citā jābūt lielākām par konstanti. Tomēr vienmēr var panākt, ka visu nevienādību zīmes ir vērstas vienā virzienā (nepieciešamības gadījumā abas nevienādības puses tiek pareizinātas ar -1).

Tiek izšķirtas *trīs* lineārās programmēšanas uzdevuma pamatformas atkarībā no ierobežojumu dažādajiem tipiem.

Lineārās programmēšanas standartuzdevums

Lineārās programmēšanas uzdevums ir dots *standartformā* (saka arī *normālformā*), ja tā nosacījumu sistēma satur tikai nevienādības $\leq$:

$$\begin{aligned}
&\max(c_1x_1 + c_2x_2 + \ldots + c_nx_n)\\
&a_{11}x_1 + a_{12}x_2 + \ldots + a_{1n}x_n \leq b_1,\\
&a_{21}x_1 + a_{22}x_2 + \ldots + a_{2n}x_n \leq b_2,\\
&\qquad\qquad\ldots\\
&a_{m1}x_1 + a_{m2}x_2 + \ldots + a_{mn}x_n \leq b_m,\\
&x_1 \geq 0,\ x_2 \geq 0, \ldots,\ x_n \geq 0,
\end{aligned}$$

(izvērstais pieraksts) vai matricu pierakstā (saīsinātais pieraksts)

$$\begin{gathered}
\max < c, x >\\
Ax \leq b, \quad x \geq 0,
\end{gathered}$$

kur $x = (x_1, x_2, ..., x_n)$, $c = (c_1, c_2, ..., c_n)$, $b = (b_1, b_2, ..., b_m)$, $A = (a_{ij})$ — koeficientu matrica. Vektoru c sauc par *lineārās formas koeficientu vektoru*, vektoru b sauc par *ierobežojumu vektoru.*

Lineārās programmēšanas kanoniskais uzdevums

Lineārās programmēšanas uzdevums ir dots *kanoniskā formā*, ja tas ir formulēts šādi:

$$\begin{aligned}
&\max(c_1x_1 + c_2x_2 + \ldots + c_nx_n)\\
&a_{11}x_1 + a_{12}x_2 + \ldots + a_{1n}x_n = b_1,\\
&a_{21}x_1 + a_{22}x_2 + \ldots + a_{2n}x_n = b_2,\\
&\qquad\quad \ldots\\
&a_{m1}x_1 + a_{m2}x_2 + \ldots + a_{mn}x_n = b_m,\\
&x_1 \geq 0,\ x_2 \geq 0, \ldots,\ x_n \geq 0,
\end{aligned}$$

vai matricu pierakstā

$$\max < c, x >$$
$$Ax = b, \quad x \geq 0.$$

Lineārās programmēšanas skaitļojamie algoritmi (simpleksa metode un tās modifikācijas) visbiežāk literatūrā ir aprakstītas tieši kanoniskās formas uzdevumiem.

Lineārās programmēšanas vispārīgais uzdevums

Vispārīgās formas gadījumā daļa ierobežojumu ir nevienādības, otra daļa vienādības, pie tam ne visi mainīgie ir nenegatīvi:

$$\begin{aligned}
&\max(c_1x_1 + c_2x_2 + \ldots + c_nx_n)\\
&a_{11}x_1 + a_{12}x_2 + \ldots + a_{1n}x_n \leq b_1,\\
&a_{21}x_1 + a_{22}x_2 + \ldots + a_{2n}x_n \leq b_2,\\
&\qquad\quad \ldots\\
&a_{k1}x_1 + a_{k2}x_2 + \ldots + a_{kn}x_n \leq b_k,\\
&a_{k+1,1}x_1 + a_{k+1,2}x_2 + \ldots + a_{k+1,n}x_n = b_{k+1},\\
&a_{k+2,1}x_1 + a_{k+2,2}x_2 + \ldots + a_{k+2,n}x_n = b_{k+2},\\
&\qquad\quad \ldots\\
&a_{m1}x_1 + a_{m2}x_2 + \ldots + a_{mn}x_n = b_m,\\
&x_1 \geq 0,\ x_2 \geq 0, \ldots,\ x_r \geq 0.
\end{aligned}$$

Šeit jāizpildās nosacījumam, ka $k \leq m$ un $r \leq n$. Standartuzdevumu varam iegūt kā vispārīgā uzdevumu speciālgadījumu, ja $k = m$, $r = n$, savukārt kanonisko uzdevumu dabūsim, ja $k = 0$, $r = n$.

Visas trīs uzdevuma formas ir ekvivalentas tādā nozīmē, ka no katras ar vienkāršiem pārveidojumiem var pāriet uz jebkuru citu. Tāpēc, ja mums ir zināms, kā atrast vienas noteiktas formas uzdevuma atrisinājumu, tad mēs pratīsim atrisināt arī citu formu uzdevumus.

Paskaidrosim teikto, parādot, kā no standartformas var tikt pie kanoniskās formas.

Pierādījums. Šajā gadījumā definēsim jaunus mainīgos z_1, z_2,..., z_m, t.i., tik daudzus jaunus mainīgos, cik ir ierobežojošo nevienādību, un apskatīsim šādu uzdevumu kanoniskajā formā:

$$\begin{aligned}
&\max(c_1x_1 + c_2x_2 + \ldots + c_nx_n + 0 \cdot z_1 + \ldots + 0 \cdot z_m)\\
&a_{11}x_1 + a_{12}x_2 + \ldots + a_{1n}x_n + z_1 = b_1,\\
&a_{21}x_1 + a_{22}x_2 + \ldots + a_{2n}x_n + z_2 = b_2,\\
&\qquad\qquad \ldots\\
&a_{m1}x_1 + a_{m2}x_2 + \ldots + a_{mn}x_n + z_m = b_m,\\
&x_1 \geq 0,\ x_2 \geq 0, \ldots,\ x_n \geq 0,\\
&z_1 \geq 0,\ z_2 \geq 0, \ldots,\ z_m \geq 0,
\end{aligned}$$

vai matricu pierakstā

$$\max < c, x >$$
$$Ax + z = b, \quad x \geq 0, \quad z \geq 0.$$

Pieņemsim, ka $(x_1^*, x_2^*, ..., x_n^*, z_1^*, z_2^*, ..., z_m^*)$ ir formulētā kanoniskā uzdevuma atrisinājums. Tādā gadījumā sākotnēji formulētā standartuzdevuma atrisinājums ir $(x_1^*, x_2^*, ..., x_n^*)$. Pārliecināsimies par to! Skaidrs, ka šis atrisinājums apmierina visus standartuzdevuma ierobežojumus: tā koordinātas ir nenegatīvas (jo tāda prasība ir arī kanoniskajā formā) un, tā kā z_1^*, z_2^*, ..., z_m^* ir nenegatīvi skaitļi, tad no vienādībām

$$a_{i1}x_1^* + a_{i2}x_2^* + \ldots + a_{in}x_n^* + z_i^* = b_i, \quad i = 1, 2, ..., m,$$

seko nevienādības

$$a_{i1}x_1^* + a_{i2}x_2^* + \ldots + a_{in}x_n^* \leq b_i, \quad i = 1, 2, ..., m.$$

Pieņemsim, ka vektors $x^* = (x_1^*, x_2^*, ..., x_n^*)$ nav standartuzdevuma atrisinājums. Tādā gadījumā eksistē cits vektors $\overline{x} = (\overline{x}_1, \overline{x}_2, ..., \overline{x}_n)$, kurš apmierina visus standartuzdevuma ierobežojumus un dod lielāku vērtību mērķa funkcijai nekā x^*:

$$c_1\overline{x}_1 + c_2\overline{x}_2 + \ldots + c_n\overline{x}_n > c_1x_1^* + c_2x_2^* + \ldots + c_nx_n^*.$$

Parādīsim, ka tas nav iespējams. Šajā nolūkā konstruēsim tādu vektoru, kas apmierina mūsu kanoniskā uzdevuma ierobežojumus un kurā mērķa (maksimizējamā) funkcija sasniedz lielāku vērtību nekā pie $(x_1^*, x_2^*, ..., x_n^*, z_1^*, z_2^*, ..., z_m^*)$, kas ir pretrunā ar šī vektora sākotnējo izvēli (tas ir jau kanoniskā uzdevuma atrisinājums!).

Apskatīsim vektoru $(\overline{x}_1, \overline{x}_2, ..., \overline{x}_n, \overline{z}_1, \overline{z}_2, ..., \overline{z}_m)$, kur

$$\overline{z}_i = b_i - (a_{i1}\overline{x}_1 + a_{i2}\overline{x}_2 + \ldots + a_{in}\overline{x}_n), \quad i = 1, 2, ..., m.$$

Tā kā $\overline{x} = (\overline{x}_1, \overline{x}_2, ..., \overline{x}_n)$ apmierina standartuzdevuma ierobežojumus, tad $\overline{z}_i \geq 0$, $i = 1, 2, ..., m$, tāpēc vektors $(\overline{x}_1, \overline{x}_2, ..., \overline{x}_n, \overline{z}_1, \overline{z}_2, ..., \overline{z}_m)$ savukārt apmierina visus papildus kanoniskā uzdevuma ierobežojumus un pie tam

$$<c, \overline{x}> \; > \; <c, x^*>,$$

kas ir pretrunā ar vektora $(x_1^*, x_2^*, ..., x_n^*, z_1^*, z_2^*, ..., z_m^*)$ optimalitāti. Iegūtā pretruna noslēdz pierādījumu (tātad lineārās programmēšanas standartuzdevumu var pārveidot par kanoniskās formas uzdevumu). ∎

Vēl vienkāršāk ir kanoniskās formas uzdevumu pārveidot par standartformas uzdevumu — nepieciešams tikai katru sistēmas vienādojumu aizvietot ar divām nevienādībām:

$$a_{i1}x_1 + a_{i2}x_2 + \ldots + a_{in}x_n = b_i \;\Leftrightarrow\; \begin{array}{l} a_{i1}x_1 + a_{i2}x_2 + \ldots + a_{in}x_n \leq b_i, \\ -a_{i1}x_1 - a_{i2}x_2 - \ldots - a_{in}x_n \leq -b_i. \end{array}$$

Ja lineārās programmēšanas vispārīgā uzdevuma mainīgajam x_j nav uzlikta prasība par nenegativitāti, tad, lai vispārīgo uzdevumu pārveidotu par standartuzdevumu, nepieciešams pieņemt, ka $x_j = u_j - v_j$, kur u_j, v_j ir divi jauni mainīgie ar nosacījumu $u_j \geq 0$, $v_j \geq 0$.

Ja nepieciešams minimizēt lineāro formu $c_1x_1 + c_2x_2 + ... + c_nx_n$, tad to var vienkārši pārveidot par maksimizācijas uzdevumu: nepieciešams atrast funkcijas $-c_1x_1 - c_2x_2 - ... - c_nx_n$ maksimumu.

Risinot lineārās programmēšanas uzdevumus, ir izveidojusies noteikta terminoloģija. Tā lineāro formu

$$<c, x> = c_1x_1 + c_2x_2 + \ldots + c_nx_n,$$

kuras maksimumu vai minimumu meklējam, sauc par *mērķa funkciju.* Vektoru

$$x = (x_1, x_2, \ldots, x_n),$$

kurš apmierina visus linārās programmēšanas uzdevuma ierobežojumus, sauc par *pieļaujamo vektoru* vai *pieļaujamo plānu.* Lineārās programmēšanas uzdevumu, kuram eksistē vismaz viens pieļaujamais plāns, sauc par *pieļaujamu uzdevumu.* Pieļaujamo plānu x^*, kurš dod mērķa funkcijas lielāko vērtību (vai mazāko vērtību minimizācijas uzdevuma gadījumā) salīdzinājumā ar citiem pieļaujamajiem plāniem x, t.i., $< c, x^* > \geq < c, x >$, sauc par *uzdevuma atrisinājumu* jeb *optimālo plānu.* Mērķa funkcijas maksimālo vērtību $d =< c, x^* >$ sauc par *uzdevuma vērtību.*

Vai lineāro modeļu uzdevumi no iepriekšējās nodaļas ir pieļaujami uzdevumi?

Skaidrs, ka diētas uzdevums, kura ierobežojumi aprakstīti ar nevienādībām (1.3.1) un (1.3.2), nebūs pieļaujams jebkurām skaitļu a_{ij} un b_i vērtībām. Patiešām, ja, piemēram, matricas A pirmajā rindā visi a_{1j} vienādi ar 0, bet $b_1 > 0$, tad pieļaujamo plānu kopa ir tukša. Nosacījums $a_{1j} = 0$, $j = 1, 2, ..., n$, nozīmē, ka viela N_1 netiek saturēta nevienā no apskatāmajiem produktiem. No šejienes seko nepieciešamais nosacījums, lai diētas uzdevums būtu pieļaujams: jebkurai no vielām N_i ir jāietilpst vismaz vienā no produktiem, t.i., matricas katrā rindā ir jābūt vismaz vienam pozitīvam skaitlim. Tā kā pēc koeficientu a_{ij} jēgas tie visi ir nenegatīvi, tad šis nosacījums ir arī pietiekams, — skaidrs, ka var paņemt vektoru x ar tik lielām koordinātām, lai visi ierobežojumi (1.3.1) būtu izpildīti.

Acīmredzot, lai transporta uzdevums būtu pieļaujams, ir nepieciešami un pietiekami, lai izpildās nosacījums

$$\sum_{j=1}^{n} b_j = D = \sum_{i=1}^{m} a_i,$$

t.i., preces pieprasījumam ir jāsakrīt ar tās piedāvājumu. Patiešām, ja plāns (x_{ij}) ir pieļaujams, tad summējot pirmo (1.3.5) vienādību pa i, bet otro vienādību pa j, iegūsim

$$\sum_{i=1}^{m} a_i = \sum_{i=1}^{m} \sum_{j=1}^{n} x_{ij} = \sum_{j=1}^{n} \sum_{i=1}^{m} x_{ij} = \sum_{i=1}^{n} b_j.$$

No otras puses, var pārliecināties, ja izpildās pieprasījuma un piedāvājuma vienādība, tad eksistē pieļaujamais plāns

$$x_{ij} = \frac{a_i b_j}{D}, \quad i = 1, 2, ..., m, \; j = 1, 2, ..., n.$$

Ražošanas plānošanas uzdevums (1.3.7) ir pieļaujams, jo triviālā veidā nulles vektors apmierina visus ierobežojumus.

Optimālo plānu ir jēga meklēt tikai lineārās programmēšanas pieļaujamajiem uzdevumiem. Tomēr jāatzīmē, ka ne jebkuram pieļaujamajam uzdevumam eksistē atrisinājums. Var būt tādas situācijas, ka pieļaujamo plānu kopa ir neierobežota un mērķa funkcija šajā kopā var būt arī neierobežota.

Piemēram, ja x_1 ir skaitlis (t.i., 1-dimensiju vektors), tad uzdevums

$$\max x_1$$
$$x_1 \leq -1, \quad x_1 \geq 0$$

nav pieļaujams. Bet uzdevums

$$\max x_1$$
$$x_1 \geq 0$$

ir pieļaujams, bet tam nav atrisinājuma.

1.5 UZDEVUMU ATRISINĀŠANAS PROBLĒMAS

Lineārās programmēšanas uzdevumu atrisinājumu meklēšanas pirmais solis būtu pirmām kārtām izmantot klasiskās matemātiskās analīzes un algebras metodes. Vienkāršākais uzdevums ir atrast diferencējamas funkcijas maksimumu (minimumu) reālo skaitļu intervālā. Pieņemsim, ka $f(x)$ ir viena argumenta funkcija, kas definēta intervālā $[a;b]$. Jāatrod funkcijas $f(x)$ maksimums šajā intervālā. Algoritms, kurš dod šī matemātiskās analīzes uzdevuma atrisinājumu, ir šāds: nepieciešams atrast atvasinājumu $f'(x)$, jāatrod visi vienādojuma $f'(x) = 0$ atrisinājumi x_1, $x_2, ...,$ x_k (t.i., stacionārie punkti), jāizskaitļo vērtības $f(x_1)$, $f(x_2)$,..., $f(x_k)$, $f(a)$, $f(b)$ un jāizvēlas starp tām lielākā. Var redzēt, ka vispārīgā gadījumā var rasties problēmas. Kā atrisināt vienādojumu $f'(x) = 0$? Vai šis uzdevums nebūs tikpat sarežģīts kā maksimuma atrašanas pamatproblēma? Ko darīt, ja stacionāro punktu ir ļoti daudz vai pat bezgalīgi daudz? Kā organizēt tādā gadījumā salīdzināšanu?

Vēl sarežģītāka situācija var izveidoties vairāku argumentu funkciju gadījumā. Pieņemsim, ka $x = (x_1, x_2, ..., x_n)$ ir n dimensiju vektors, X ir slēgta apakškopa n dimensiju telpā $\mathbf{R}^n$. Jāatrod funkcijas $f(x)$ maksimums kopā X. Ja funkcija ir diferencējama, tad, lai atrastu funkcijas $f(x)$ maksimumu kopā X, nepieciešams atrast visus sistēmas $\dfrac{\partial f(x)}{\partial x_i} = 0$, $i = 1, 2, ..., n$, atrisinājumus (vektorus!) x^1, x^2,..., x^k, tad jāizskaitļo funkcijas vērtības šajos punktos un *visos* kopas X robežas punktos, pēc tam jāizvēlas lielākā vērtība. Ja pat mums izdodas viegli atrisināt doto sistēmu, tad problēmas var rasties, lai izskaitļotu funkcijas f vērtības uz kopas X robežas. Jo kopas X robeža, kas atrodas n dimensiju telpā pie $n > 1$ sastāv no bezgalīgi daudziem punktiem (izņemot kādus kopas X triviālos gadījumus). Kā rīkoties šajā gadījumā, vispārīgu algoritmu klasiskā matemātiskā analīze nedod — katram uzdevumam tiek meklēta sava metode. Piemēram, ja kopa X tiek uzdota ar vienādību palīdzību, tad funkcijas ekstrēmu var meklēt ar Lagranža reizinātāju metodi, taču sistēmas atrisināšanas problēmas saglabājas joprojām. Lineārās programmēšanas uzdevumu gadījumā risināšanu var sarežģīt ierobežojošās nevienādības.

Ievērosim vēl vienu niansi, kas raksturīga tieši lineārās programmēšanas standartuzdevumam. Apzīmēsim ar X pieļaujamo vektoru kopu. Par cik mērķa funkcijas parciālie atvasinājumi ir vienādi ar vektora c koordinātām, kas noteikti vienlaicīgi visas nav 0 (jo tad nav ko optimizēt), tad mērķa funkcijai nav lokālo ekstrēmu (jo nav stacionāro punktu), līdz ar to maksimālās vērtības var tikt sasniegtas tikai uz kopas X robežas. Kopas X robeža tiešā veidā nav uzdota. Līdz ar to **lineārās programmēšanas uzdevuma atrisināšanas pamatproblēma ir kopas X robežas noteikšana un mērķa funkcijas vērtību izskaitļošanas tāda organizēšana (vērtību ir bezgalīgi daudz!), lai atrastu optimālo plānu**.

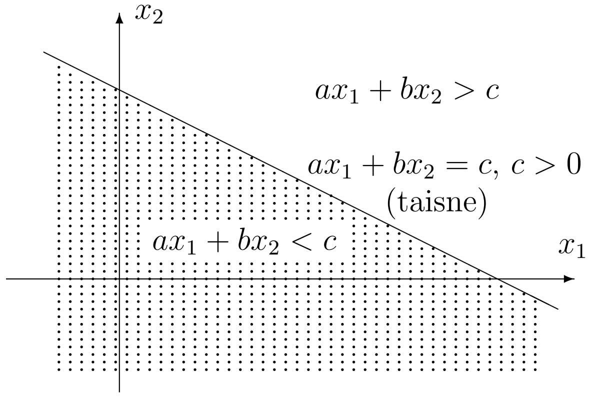
1.6 LINEĀRĀS PROGRAMMĒŠANAS UZDEVUMA ĢEOMETRISKĀ INTERPRETĀCIJA

Apskatīsim lineārās programmēšanas vienu vienkāršu piemēru.

Piemērs 1.6.1.

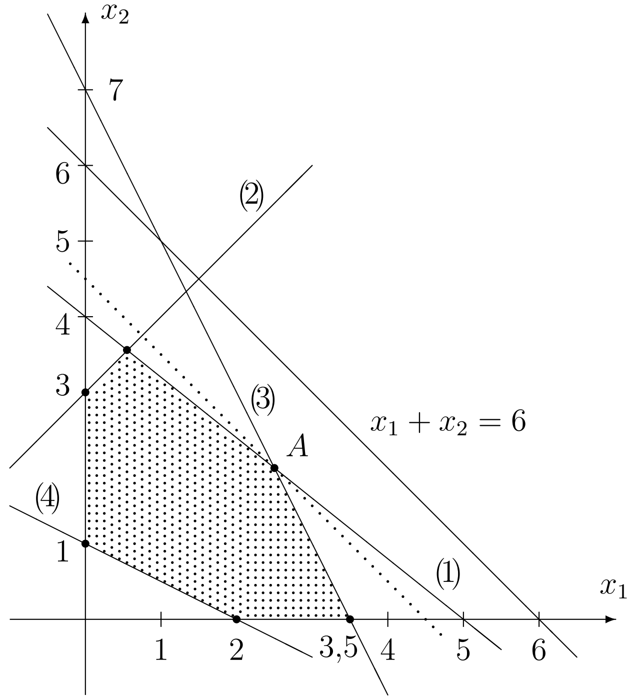
$$\begin{aligned}
&\max(x_1 + x_2)\\
&4x_1 + 5x_2 \le 20,\\
&-x_1 + x_2 \le 3,\\
&10x_1 + 5x_2 \le 35,\\
&-0,5x_1 - x_2 \le -1,\\
&x_1 \ge 0,\ x_2 \ge 0.
\end{aligned}$$

Vienādojums $ax_1 + bx_2 = c$ (a un b vienlaicīgi nav 0) plaknē definē taisni, kas plakni sadala divās pusplaknēs. Nevienādība $ax_1 + bx_2 \le c$ definē vienu no pusplaknēm (ieskaitot pašu taisni). Lai noteiktu, kura ir īstā pusplakne, nepieciešams izvēlēties vienu punktu plaknē (x_1^0, x_2^0), kurš neatrodas uz pašas taisnes, un noskaidrot vērtību $ax_1^0 + bx_2^0$, ja tā ir $< c$, tad tiek apskatīta punkta (x_1^0, x_2^0) puses pusplakne, ja $ax_1^0 + bx_2^0 > c$, tad mūs interesējošā ir tā pusplakne, kas atrodas taisnes otrā pusē raugoties no punkta (x_1^0, x_2^0) atrašanās vietas (skatīt 1.6.1.zīm.). Ja taisne neiet caur koordinātu sākumpunktu, tad izdevīgi ņemt $(x_1^0, x_2^0) = (0, 0)$.



1.6.1. zīm.

Šie apsvērumi ļauj vizualizēt piemērā formulētā lineārās programmēšanas uzdevuma pieļaujamo vektoru kopu. Tā attēlota 1.6.2.zīmējumā.



1.6.2. zīm.

Apskatīsim mērķa funkciju $x_1 + x_2$. 1.6.2.zīmējumā uzzīmēta funkcija $x_1 + x_2 = 6$, kuras grafiks nekrusto pieļaujamo vektoru kopu. Tas nozīmē, ka starp pieļaujamajiem punktiem nav tāda, kurš dotu mērķa funkcijas vērtību 6. Vienādojums $x_1 + x_2 = d$ pie dažādām d vērtībām apraksta paralēlu taišņu saimi. Visiem punktiem, kas atrodas uz vienas no šīm taisnēm, ir viena un tā pati mērķa funkcijas vērtība d, tāpēc dažkārt šīs taisnes sauc par *līmeņa līnijām.* Svarīga līmeņa līniju īpašība ir tāda, ka, veicot paralēlo pārbīdi, vienā virzienā līmaņa vērtība d tikai pieaug, bet otrā virzienā tikai samazinās. Lineārās programmēšanas uzdevumu var pārformulēt šādi: atrast tādu maksimālo vērtību d, ar kuru taisne $x_1 + x_2 = d$ šķērso pieļaujamo vektoru kopu. Dotajā piemērā, ja pārvieto taisni $x_1 + x_2 = 6$ paralēli sev aizvien tuvāk iepunktētajam apgabalam, tad pirmais punkts, kas tiks krustots, būs punkts A (1.6.2.zīmējumā atbilstošās mērķa funkcijas grafiks uzzīmēts kā punktota līnija), t.i., (1) un (3) taisnes krustpunkts. Atrisinot vienādojumu sistēmu

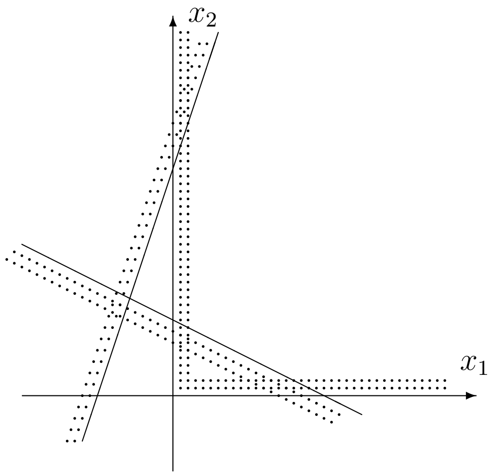
$$\begin{cases} 4x_1 + 5x_2 = 20, & (1) \\ 10x_1 + 5x_2 = 35, & (3) \end{cases}$$

atradīsim punkta A koordinātas $(2{,}5; 2)$, kas dod mērķa funkcijas vērtību 4,5. ∎

Šajā piemērā var ieraudzīt visas galvenās lineārās programmēšanas īpatnības: pieļaujamo punktu kopa ir izliekts daudzstūris, kas veidojas kā pusplakņu šķēlums, un mērķa funkcijas lielākā vērtība tiek sasniegta daudzstūra virsotnē. Vispārīgā gadījumā pieļaujamo punktu jeb vektoru kopa var būt:

⋆ tukša (tas nozīmē, ka nevienādību sistēma ir nesaderīga); piemērs apskatāms 1.6.3.zīmējumā;

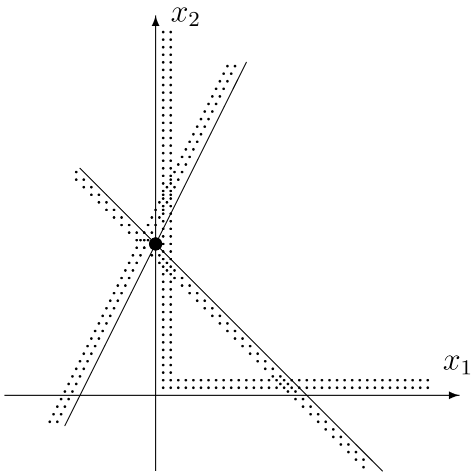
$$x_1 + 2x_2 \le 2,$$
$$-3x_1 + x_2 \ge 3,$$
$$x_1 \ge 0,\ x_2 \ge 0.$$



1.6.3. zīm.

⋆ viens punkts; piemērs apskatāms 1.6.4.zīmējumā;

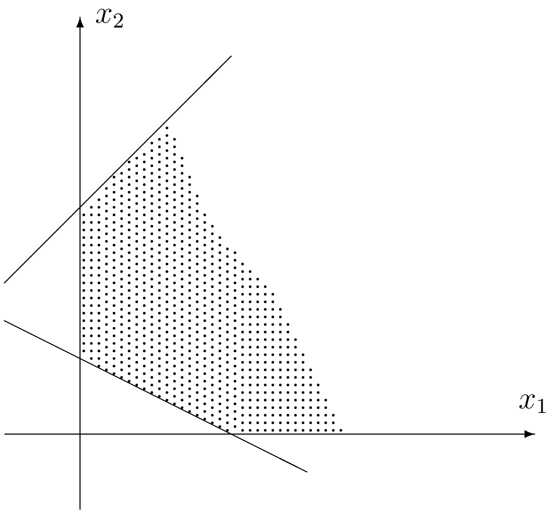
$$x_1 + x_2 \le 2,$$
$$-2x_1 + x_2 \ge 2,$$
$$x_1 \ge 0,\ x_2 \ge 0.$$



1.6.4. zīm.

⋆ izliekts daudzstūris; piemērs apskatīts iepriekš (1.6.2.zīmējums);
⋆ neierobežots izliekts apgabals; piemērs apskatāms 1.6.5.zīmējumā;

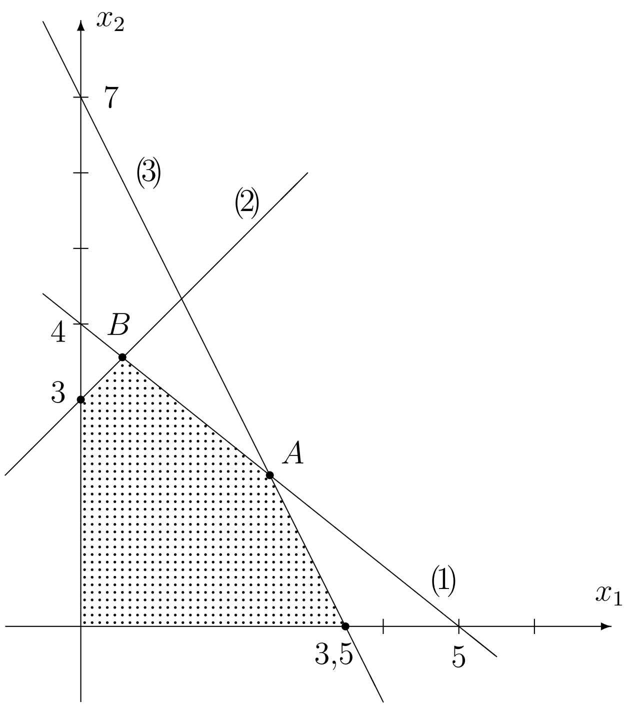
$$-x_1 + x_2 \le 3,$$
$$x_1 + 2x_2 \ge 2,$$
$$x_1 \ge 0,\ x_2 \ge 0.$$



1.6.5. zīm.

Pieļaujamo punktu kopas veids ietekmē lineārās programmēšanas uzdevuma atrisinājuma eksistences iespējamību. Atkarībā no kopas veida un pašas funkcijas uzdevumam var nebūt atrisinājuma, var būt tikai viens vienīgs atrisinājums (bet, piemēram, neierobežotas kopas gadījumā var neeksistēt mērķa funkcijas maksimums, bet var eksistēt minimums), kā arī iespējama situācija ar bezgalīgi daudziem atrisinājumiem. Piemēram, 1.6.6.zīmējumā parādīta tāda pieļaujamo vektoru kopa un tāda mērķa funkcija, ka lineārās programmēšanas uzdevumam ir bezgalīgi daudz atrisinājumu — par tādiem der visi nogriežņa AB punkti.

$$
\begin{array}{ll}
\max(4x_1 + 5x_2) & \\
4x_1 + 5x_2 \le 20, & (1) \\
-x_1 + x_2 \le 3, & (2) \\
10x_1 + 5x_2 \le 35, & (3) \\
x_1 \ge 0,\ x_2 \ge 0. &
\end{array}
$$



1.6.6. zīm.

Pieņemsim, ka lineārās programmēšanas uzdevums dots kanoniskā formā un $n - m = 2$. Šajā gadījumā divu brīvi izraudzītu nezināmo optimālās vērtības var noteikt grafiski. Pēc tam pārējo m nezināmo optimālās vērtības nosaka, izmantojot izteiksmes, kas iegūtas no uzdevuma nosacījumu sistēmas.

Ja $n - m = 2$, tad vispirms uzdevuma ierobežojumu sistēmu atrisina attiecībā pret m brīvi izraudzītiem nezināmiem, piemēram:

$$
\begin{array}{llll}
a'_{11}x_1 + a'_{12}x_2 + x_3 & & & = b'_1, \\
a'_{21}x_1 + a'_{22}x_2 & + x_4 & & = b'_2, \\
\quad \ldots & & & \\
a'_{m1}x_1 + a'_{m2}x_2 & & + x_n & = b'_m,
\end{array}
$$

jeb

$$\begin{aligned}
x_3 &= b_1' - (a_{11}'x_1 + a_{12}'x_2) \geq 0,\\
x_4 &= b_2' - (a_{21}'x_1 + a_{22}'x_2) \geq 0,\\
&\dots\\
x_n &= b_m' - (a_{m1}'x_1 + a_{m2}'x_2) \geq 0.
\end{aligned} \tag{1.6.1}$$

Ievietojot šo m nezināmo izteiksmes mērķa funkcijā un izdarot algebriskus pārveidojumus, iegūst pārveidotu mērķa funkciju, kas satur tikai divus nezināmos:

$$c_1'x_1 + c_2'x_2 + c_0.$$

Konstante c_0 maksimizāciju neiespaido, tāpēc risināšanā var izmantot šādu mērķa funkciju

$$c_1'x_1 + c_2'x_2.$$

Lineārās programmēšanas uzdevums iegūst šādu izskatu:

$$\begin{aligned}
&\max(c_1'x_1 + c_2'x_2)\\
&a_{11}'x_1 + a_{12}'x_2 \leq b_1',\\
&a_{21}'x_1 + a_{22}'x_2 \leq b_2',\\
&\quad\dots\\
&a_{m1}'x_1 + a_{m2}'x_2 \leq b_m',\\
&x_1 \geq 0,\ x_2 \geq 0.
\end{aligned}$$

Tā kā uzdevums satur tikai divus nezināmos, tad to var atrisināt grafiski. Rezultātā iegūsim optimālā atrisinājuma x^* koordinātas x_1^* un x_2^*, pārējās optimālā atrisinājuma x^* koordinātas x_3^*,..., x_n^* nosaka, izmantojot vienādības (1.6.1). Pēc tam aprēķina mērķa funkcijas maksimālo vērtību pēc formulas

$$c_1'x_1^* + c_2'x_2^* + c_0.$$

Piemērs 1.6.2. Atrisināsim grafiski lineārās programmēšanas uzdevumu

$$\max(5x_1 + 5x_2 + x_3 + x_4 + x_5)$$

$$\begin{array}{rrrrrll}
x_1 & +3x_2 & +2x_3 & +x_4 & & = 8, & (1)\\
2x_1 & +x_2 & +x_3 & & +x_5 & = 8, & (2)\\
x_1 & +x_2 & & +x_4 & & = 4, & (3)
\end{array}$$

$$x_1 \geq 0,\ x_2 \geq 0,\ x_3 \geq 0,\ x_4 \geq 0,\ x_5 \geq 0.$$

No vienādojuma (1) atskaitot (3), dabūsim

$$2x_2 + 2x_3 = 4 \text{ jeb } x_2 + x_3 = 2.$$

Varam aizvietot $x_2 + x_3$ vienādojumā (2) ar 2 un iegūt

$$2x_1 + 2 + x_5 = 8 \text{ jeb } 2x_1 + x_5 = 6.$$

Rezultātā ierobežojumu sistēma ir šāda

$$\begin{array}{rrrrrl} & x_2 & +x_3 & & & = 2, \\ x_1 & +x_2 & & +x_4 & & = 4, \\ 2x_1 & & & & +x_5 & = 6. \end{array}$$

No šīs sistēmas var izteikt nezināmos x_3, x_4 un x_5:

$$\begin{array}{l} x_3 = 2 - x_2 \geq 0, \\ x_4 = 4 - (x_1 + x_2) \geq 0, \\ x_4 = 6 - 2x_1 \geq 0. \end{array}$$

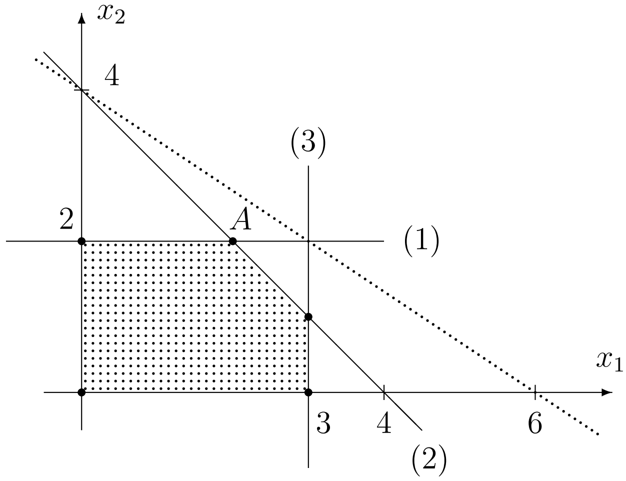
Ievietojot šo nezināmo izteiksmes mērķa funkcijas izteiksmē, veicot saīsināšanu, iegūsim mērķa funkciju, kas satur tikai divus nezināmos

$$5x_1+5x_2+x_3+x_4+x_5 = 5x_1+5x_2+2-x_2+4-x_1-x_2+6-2x_1 = 2x_1+3x_2+12.$$

Tātad grafiski jāmeklē atrisinājums šādam lineārās programmēšanas uzdevumam:

$$\begin{array}{ll} \max(2x_1 + 3x_2) & \\ x_2 \leq 2, & (1) \\ x_1 + x_2 \leq 4, & (2) \\ 2x_1 \leq 6, & (3) \\ x_1 \geq 0,\ x_2 \geq 0. & \end{array}$$

Uzdevuma grafiskais atrisinājums attēlots 1.6.7.zīmējumā. Punktotā līnija ir mērķa funkcijas grafiks pie vērtības (līmeņa) 6. Domās zīmējot paralēlās līmeņa līnijas, nonāksim pie secinājuma, ka mērķa funkcijas maksimālā vērtība tiks sasniegta punktā A. Punkta A koordinātas ir $x_1^* = 2$ un $x_2^* = 2$. Tādā gadījumā $x_3^* = 0$, $x_4^* = 4 - (2+2) = 0$, $x_5^* = 6 - 4 = 2$.



1.6.7. zīm.

Tātad sākumā dotā uzdevuma atrisinājums ir $x^* = (2; 2; 0; 0; 2)$ un mērķa funkcijas maksimālā vērtība ir $2x_1^* + 3x_2^* + 12 = 4 + 6 + 12 = 22$. ■

NODAĻA 2

SIMPLEKSA ALGORITMS

2.1 LINEĀRU VIENĀDOJUMU SISTĒMU ATRISINĀŠANA AR ŽORDĀNA-GAUSA METODI

Lielākā daļa no lineārās programmēšanas uzdevumu atrisināšanas skaitliskajām metodām izmanto ideju, kas lineāru vienādojumu sistēmu

$$\begin{aligned}
&a_{11}x_1 + a_{12}x_2 + ... + a_{1n}x_n = b_1,\\
&a_{21}x_1 + a_{22}x_2 + ... + a_{2n}x_n = b_2,\\
&\qquad ...\\
&a_{m1}x_1 + a_{m2}x_2 + ... + a_{mn}x_n = b_m
\end{aligned}$$

pārveido ērtākā pieraksta formā ar tā saucamo Žordāna-Gausa metodi, kas tiek mācīta algebras pamatkursos. Atgādināsim, kā tas tiek darīts.

Pirmajā sistēmas vienādojumā tiek sameklēts koeficients $a_{1i_1} \neq 0$. Ar šī koeficienta palīdzību visi koeficienti pie mainīgā x_{i_1} tiek pārveidoti par 0. Šajā nolūkā pirmais sistēmas vienādojums tiek pareizināts ar skaitli $-\dfrac{a_{ri_1}}{a_{1i_1}}$ un tiek pieskaitīts r-tajam sistēmas vienādojumam, $2 \leq r \leq m$. Pēc tam pirmais vienādojums tiek izdalīts ar a_{1i_1}. Šādus pārveidojumus sauc par *elementārajiem pārveidojumiem*, jo tie neizmaina sistēmas atrisinājumu kopu. Iegūtajā ekvivalentajā sistēmā mainīgais x_{i_1} ir atrodams tikai pirmajā vienādojumā un tam blakus ir koeficients 1. Šādu mainīgo x_{i_1} sauc par *bāzes mainīgo*.

Analoģiskas operācijas tiek izdarītas pēc kārtas ar katru sistēmas vienādojumu. Pie tam katrā reizē tiek izmainīti sistēmas vienādojumi un papildinās bāzes mainīgo saraksts.

Žordāna-Gausa metodes izmantošanas rezultātā tiek konstatēts, ka vai nu vienādojumu sistēma ir nesaderīga (kāda vienādojuma visi koeficienti ir 0, bet labās puses brīvais loceklis nav 0), vai arī tiek noskaidroti un atmesti visi "liekie" vienādojumi (kuriem visi koeficienti un brīvais loceklis ir 0). Otrajā gadījumā vienādojumu sistēma ir šādā izskatā

$$x_i + \sum_{j \in \omega} a_{ij} x_j = \alpha_{i0}, \quad i \in \sigma,$$

kur $\sigma = \{i_1, i_2, ..., i_k\}$ — bāzes mainīgo indeksu saraksts, $\omega = \{1, 2, ..., n\} \setminus \sigma$ — indeksu kopa mainīgajiem, kas neietilpst bāzes mainīgo sarakstā (turpmāk mainīgos, kas neietilpst bāzes mainīgo sarakstā, sauksim par *brīvajiem mainīgajiem*). Šajā gadījumā k ($k \leq m$) ir sākotnējās vienādojumu sistēmas koeficientu matricas A rangs.

Iegūto vienādojumu sistēmu sauksim par *reducēto sistēmu*, kurai atbilst bāzes mainīgie ar indeksu kopu σ.

Nav grūti ieraudzīt, ka par bāzes mainīgajiem x_{i_1}, x_{i_2},..., x_{i_k} var būt tikai tāds mainīgo komplekts, kuriem atbilstošā matricas A kolonnu sistēma $\{a^{i_1}, a^{i_2}, ..., a^{i_k}\}$ ir lineāri neatkarīga.

Reducētā vienādojumu sistēma ir ērtāka par sākotnējo, jo dod tiešu visu atrisinājumu kopas aprakstu. Piešķirot brīvajiem mainīgajiem x_j, $j \in \omega$, patvaļīgas vērtības, bāzes mainīgos var izrēķināt pēc formulām

$$x_i = \alpha_{i0} - \sum_{j \in \omega} \alpha_{ij} x_j, \quad i \in \sigma,$$

tādējādi atrodam jebkuru sistēmas atrisinājumu. Speciālgadījumā viens no atrisinājumiem ir vektors $x^0 = (x_1^0, x_2^0, ..., x_n^0)$, kur

$$x_i^0 = \begin{cases} \alpha_{i0}, & \text{ja } i \in \sigma, \\ 0, & \text{ja } i \in \omega. \end{cases}$$

Piemērs 2.1.1. Apskatīsim iepriekšējas nodaļas pēdējā piemēra vienādojumu sistēmu

$$\begin{cases} x_1 & +3x_2 & +2x_3 & +x_4 & & = 8 \\ 2x_1 & +x_2 & +x_3 & & +x_5 & = 8 \\ x_1 & +x_2 & & +x_4 & & = 4 \end{cases}$$

Veicot Žordāna-Gausa izslēgšanas metodes soļus, iegūsim reducēto vienādojumu sistēmu šādā formā

$$\begin{cases} x_3 & & +x_2 & = 2 \\ x_4 & +x_1 & +x_2 & = 4 \\ x_5 & +2x_1 & & = 6 \end{cases}$$

Šajā gadījumā bāzes mainīgie ir x_3, x_4 un x_5, bet brīvie mainīgie ir x_1 un x_2. Izmantojot iepriekšējos apzīmējums, $\sigma = \{3, 4, 5\}$ un $\omega = \{1, 2\}$, $k = 3$ ir matricas rangs. Sākotnējās vienādojumu sistēmas kolonnas

$$a^3 = \begin{pmatrix} 2 \\ 1 \\ 0 \end{pmatrix}, \quad a^4 = \begin{pmatrix} 1 \\ 0 \\ 1 \end{pmatrix}, \quad a^5 = \begin{pmatrix} 0 \\ 1 \\ 0 \end{pmatrix}$$

veido lineāri neatkarīgu vektoru sistēmu. Par to var pārliecināties tiešā veidā, izrēķinot determinantu

$$\Delta = \begin{vmatrix} 2 & 1 & 0 \\ 1 & 0 & 1 \\ 0 & 1 & 0 \end{vmatrix} = -2 \neq 0.$$

Vispārīgā gadījumā (ja nav uzlikti nekādi ierobežojumi par mainīgo nenegativitāti) sākotnējās sistēmas atrisinājums ir

$$\begin{aligned} &x_3 = 2 - x_2, \\ &x_4 = 4 - (x_1 + x_2), \\ &x_4 = 6 - 2x_1, \\ &x_1,\, x_2 \in \mathbf{R}. \end{aligned}$$

Speciālgadījumā viens no atrisinājumiem ir vektors $x^0 = (0, 0, 2, 4, 6)$. ∎

Pieņemsim, ka matricas A rangs ir m (t.i., $k = m$). Ja apzīmējam ar

$$A_\sigma = (a^{i_1}, a^{i_2}, ..., a^{i_m})$$

matricu, kas sastāv no matricas A kolonnām un atbilst bāzes mainīgajiem, tad A_σ ir kvadrātiska nedeģenerēta (jeb nesingulāra) matrica ar izmēriem $m \times m$. Šajā gadījumā Žordāna-Gausa pārveidojumu gala rezultāts iegūstams, pareizinot sākotnējo sistēmu ar matricu A_σ^{-1}, tādējādi reducētā sistēma ir šāda

$$A_\sigma^{-1} A x = A_\sigma^{-1} b.$$

Iegūto faktu var pārbaudīt tiešā veidā. Apzīmēsim $m \times n$ matricas $A_\sigma^{-1}A$ elementus ar β_{ij}, $i = 1, 2, ..., m$, $j = 1, 2, ..., n$. No inversās matricas definīcijas seko, ja $i \in \sigma$, tad $\beta_{ij} = 0$, ja $j \notin \sigma$, tad $\beta_{ij} = 1$. Apzīmēsim vektoru $A_\sigma^{-1}b$ ar $\beta^0 = (\beta_{10}, \beta_{20}, ..., \beta_{m0})$. Tādā vienādojumu sistēma $A_\sigma^{-1}Ax = A_\sigma^{-1}b$ iegūst izskatu

$$x_i + \sum_{j \in \omega} \beta_{ij} x_j = \beta_{i0}, \quad i \in \sigma.$$

Reizinot vienādojumu sistēmua ar nedeģenerētu matricu, atrisinājumu kopa nemainās; viegli ieraudzīt, ka $\beta_{ij} = \alpha_{ij}$ visiem i, j.

Lai noskaidrotu lineārās formas $< c, x >$ uzvedību vektoru kopā, kas ir sistēmas $Ax = b$ atrisinājumi, izveidosim papildus jaunu mainīgo x_0 un apskatīsim vienādojumu sistēmu

$$x_0 - < c, x >= 0, \quad Ax = b.$$

Tā kā vienādojums $x_0 - < c, x >= 0$ atrodas pirmais un koeficients pie papildus mainīgā x_0 ir atšķirīgs no 0, tad, izmantojot Žordāna-Gausa metodi, var šo mainīgo vienmēr iekļaut bāzes mainīgo sarakstā. Ja $\sigma = \{i_1, i_2, ..., i_m\}$ ir bāzes mainīgo indeksu saraksts vienādojumam $Ax = b$ un ir jau atrasta atbilstošā reducētā sistēma (uzskatīsim, ka matricas A rangs ir m), tad lai atrastu reducēto sistēmu, kas atbilst sarakstam $\{0, i_1, i_2, ..., i_m\}$, nepieciešams mainīgo x_0 izteikt caur brīvajiem mainīgajiem

$$x_0 =< c, x >= \sum_{j=1}^{n} c_j x_j = \sum_{i \in \sigma} c_i x_i + \sum_{j \in \omega} c_j x_j.$$

Ievietojot šajā izteiksmē bāzes mainīgo x_i, $i \in \sigma$, izteiksmes, iegūsim

$$x_0 = \sum_{i \in \sigma} c_i \alpha_{i0} - \sum_{j \in \omega} \left(\sum_{i \in \sigma} c_i \alpha_{ij} - c_j \right) x_j.$$

Izmantojot iepriekš definēto vektoru x^0, ievērosim, ka $\sum\limits_{i \in \sigma} c_i \alpha_{i0} =< c, x^0 >$. Izmantosim apzīmējumu

$$\alpha_{0j} = \sum_{i \in \sigma} \alpha_{ij} - c_j, \quad j \in \omega,$$

tad reducētā sistēma pierakstāma šādā izskatā

$$\begin{aligned} x_0 + \sum_{j \in \omega} \alpha_{0j} x_j &=< c, x^0 >, \\ x_i + \sum_{j \in \omega} \alpha_{ij} x_j &= \alpha_{i0}, \quad i \in \sigma. \end{aligned} \tag{2.1.1}$$

2.2 LINEĀRĀS ALGEBRAS UN IZLIEKTĀS ANALĪZES PAMATJĒDZIENI

Šajā paragrāfā īsi izklāstīsim tās definīcijas un teorēmas no lineārās algebras un izliektās analīzes, kas plaši tiek izmantotas, lai risinātu lineārās programmēšanas problēmas.

Fundamentāls lineārās algebras jēdziens ir *lineārā* (reālā) *telpa*. Ar šo jēdzienu saprot noteiktu elementu (kurus sauc par vektoriem) kopu, kuriem ir definēta saskaitīšanas un reizināšanas ar reālu skaitli (skalāru) operācijas, pie tam iegūtie rezultāti pieder tai pašai kopai. Speciālgadījumā lineāra telpa ir reālo skaitļu ass, plakne, trīs dimensiju telpa.

Par vektoru a^1, $a^2, ...,$ a^m *lineāro kombināciju* ar (reāliem) koeficientiem λ_1, $\lambda_2, ...,$ λ_m sauc vektoru $\lambda_1 a^1 + \lambda_2 a^2 + ... + \lambda_m a^m$.

Lineāras telpas vektoru sistēmu a^1, $a^2, ...,$ a^m sauc par *lineāri atkarīgu*, ja eksistē tādi reāli skaitļi λ_1, $\lambda_2, ...,$ λ_m, kas vienlaicīgi visi nav 0, bet to lineārā kombinācija $\lambda_1 a^1 + \lambda_2 a^2 + ... + \lambda_m a^m$ ir vienāda ar 0 vektoru. Pretējā gadījumā sistēmu a^1, $a^2, ...,$ a^m sauc par *lineāri neatkarīgu*, t.i., doto vektoru lineārā kombinācija ir vienāda ar nulles vektoru tikai tad, ja visi koeficienti λ_1, $\lambda_2, ...,$ λ_m ir vienādi ar 0.

Maksimālo iespējamo vektoru skaitu, kuri var veidot lineāri neatkarīgu vektoru sistēmu dotajā lineārajā telpā, sauc par *telpas dimensiju*, bet jebkurus lineāri neatkarīgus vektorus dimensijas skaitā sauc par *telpas bāzi.* Lineāro telpu parasti apzīmē ar $\mathbf{R}^n$, kur n — telpas dimensija. Jebkuru dotās lineārās telpas apakškopu, kurai pašai piemīt lineāras telpas īpašības, sauc par *lineāru apakštelpu.* Kopu H, kuru iegūst ar lineāras apakštelpas $L \subset \mathbf{R^n}$ pārbīdi par vektoru $a \in \mathbf{R}^n$: $H = L + a$, sauc par *afīnu telpu* (vai kopu). Jebkurai lineārai telpai vai apakštelpai noteikti pieder nulles vektors, bet afīnai telpai nav obligāti jāsatur nulles vektors. Par plaknes apakštelpas piemēru der jebkura taisne, kura iet caur koordinātu sākumpunktu, bet afīnā kopa var būt jebkura taisne plaknē. Afīnas kopas raksturīga īpašība ir tāda, ka tai pieder jebkura taisne, kas savieno kopas divus patvaļīgus punktus. Afīnas kopas dimensija sakrīt ar tās lineārās apakštelpas dimensiju, kuras pārbīde ir afīnā kopa.

Aplūkosim lineāru telpu $\mathbf{R}^n$. $\mathbf{R}^n$ afīnās kopas ar dimensiju viens sauc par taisnēm, bet ar dimensiju $n - 1$ — par hiperplaknēm. Tā, piemēram, parastā plakne ir trīsdimensiju telpas $\mathbf{R}^3$ hiperplakne, bet taisne ir plaknes $\mathbf{R}^2$ hiperplakne.

Definīcija 2.2.1. Lineāras telpas $\mathbf{R}^n$ apakškopu V sauc par *izliektu*, ja tā satur taisnes nogriezni, kas savieno jebkurus divus kopas V punktus, t.i.,

$$\forall a,\ b \in V\ \forall \lambda \in [0;1]:\ x = (1-\lambda)a + \lambda b \in V.$$

Izliektu kopu šķēlums ir atkal izliekta kopa. Vektoru a^1, $a^2, ...,$ a^m lineāro kombināciju

$$\sum_{i=1}^{m} \lambda_i a^i$$

sauc par *izliektu*, ja $\lambda_i \geq 0$, $i = 1, ..., m$ un

$$\sum_{i=1}^{m} \lambda_i = 1.$$

Kopu, kas satur kopas M punktu visas izliektās kombinācijas, sauc par dotās kopas *izliekto čaulu.* Var pierādīt, ka kopas M izliektā čaula ir mazākā izliektā kopa, kas satur kopu M.

Galīgas punktu kopas izliekto čaulu sauc par *izliektu daudzskaldni*, bet galīga skaita slēgtu apakštelpu netukšo šķēlumu sauc par *izliektu daudzskalņa kopu*, kas atšķirībā no izliektā daudzskaldņa var nebūt ierobežota kopa. Visos apskatītajos iepriekšējās nodaļas piemēros lineārās programmēšanas uzdevuma pieļaujamo plānu kopas bija plaknes izliektas daudzskaldņa kopas. Šāda šo kopu reprezentācija literatūrā tiek saukta par *lineāras programmēšanas uzdevuma pirmo ģeometrisko interpretāciju.*

Teorēma 2.2.1. Pieņemsim, ka dots lineārās programmēšanas uzdevums kanoniskā formā

$$\max < c, x >$$
$$Ax = b, \quad x \geq 0.$$

Dotā uzdevuma pieļaujamo plānu kopa ir izliekta.

Pierādījums. Izvēlamies divus patvaļīgus pieļaujamos plānus x^1 un x^2, fiksēsim $\lambda \in [0;1]$. Tad

$$Ax^1 = b,\ x^1 \geq 0,$$
$$Ax^2 = b,\ x^2 \geq 0.$$

Pēc izliektas kopas definīcijas ir jāpierāda, ka plānu x^1 un x^2 izliektā lineārā kombinācija

$$x = (1-\lambda)x^1 + \lambda x^2$$

ir pieļaujams plāns. Pārbaudīsim to:

$$Ax = A[(1-\lambda)x^1 + \lambda x^2] = (1-\lambda)Ax^1 + \lambda Ax^2 = (1-\lambda)b + \lambda b = b. \blacksquare$$

Definīcija 2.2.2. Izliektas kopas V punktu v sauc par *stūra* (vai malējo) punktu, ja tas nav iekšējs punkts nevienam no nogriežņiem, kura galapunkti pieder kopai V, t.i., izliektas kopas $V \subset \mathbf{R}^n$ punktu x sauc par *stūra* punktu, ja to nevar uzrakstīt formā $x = (1-\lambda)x^1 + \lambda x^2$, kur $x^1 \neq x^2$, $x^1 \in V$, $x^2 \in V$ un $0 < \lambda < 1$.

Izliekta daudzskaldņa stūra punkti ir tā virsotnes, bet pats tas ir savu virsotņu izliekta čaula. Ne jebkurai kopai eksistē stūra punkti. Piemēram, koordinātu plaknes slēgtajai augšējai pusplaknei ir bezgalīgi daudz robežpunktu (visi taisnes punkti, kuri definē šo pusplakni), bet tai nav neviena stūra punkta. Savukārt riņķim visi riņķa līnijas punkti ir stūra punkti. Izliektam daudzstūrim (to var nosaukt arī par izliektu daudzskaldni) plaknē visas virsotnes ir stūra punkti.

Kopu K sauc par *konusu* ar virsotni punktā x_0, ja $x_0 \in K$, un no tā, ka kaut kāds punkts x_1 pieder kopai K, seko, ka K satur staru, kurš sākas punktā x_0 un iet caur punktu x_1, t.i.,

$$\{x \in \mathbf{R}^n \,|\, x = (1-\lambda)x_0 + \lambda x_1,\ \lambda \geq 0\} \subset K$$

vai

$$\{x \in \mathbf{R}^n \,|\, x = x_0 + \lambda(x_1 - x_0),\ \lambda \geq 0\} \subset K.$$

Galīga skaita staru, kas iziet no viena punkta, izliekto čaulu sauc par *izliektu daudzskaldņa konusu* ar virsotni dotajā punktā.

2.3 LINEĀRĀS PROGRAMMĒŠANAS PAMATTEORĒMAS

Šajā paragrāfā apskatīsim dažas teorēmas, kas atspoguļo lineārās programmēšanas uzdevumu nozīmīgas īpašības un kuras kalpo par bāzi uzdevumu atrisināšanas metodēs. Šīs teorēmas vispārina tās īpašības, kuras tikām novērojuši divu mainīgo gadījumā, gadījumam, kad mainīgo skaits ir patvaļīgs.

Teorēma 2.3.1. Ja mērķa funkcija z sasniedz maksimālo vērtību pieļaujamo plānu kopas D noteiktā punktā, tad tā sasniedz šo maksimālo vērtību arī kādā no kopas D stūra punktiem.

Pierādījums. Mēs pierādījumu veiksim gadījumā, ja kopa D ir izliekts daudzskaldnis (tātad ierobežota kopa). Ierobežotām un izliektām kopām izpildās šāda īpašība (to šeit nepierādīsim): ja D ir slēgta ierobežota izliekta kopa, kurai ir galīgs skaits stūra punktu, tad jebkuru punktu $x \in D$ var pierakstīt kā kopas D stūra punktu izliektu kombināciju.

Pieņemsim, ka x^1, $x^2, ...,$ x^m ir kopas D stūra punkti, bet x^* ir punkts, kurā mērķa funkcija z sasniedz maksimālo vērtību.

Pēc iepriekš formulētās īpašības punktu x^* var pierakstīt kā stūra punktu x^1, $x^2, ...,$ x^m izliekto kombināciju

$$x^* = \sum_{i=1}^{m} \lambda_i x^i, \text{ kur } \sum_{i=1}^{m} \lambda_i = 1, \ \lambda_i \geq 0, \ i = 1, 2, ..., m.$$

Tā kā punktā x^* funkcija $z = cx$ sasniedz maksimālo vērtību, tad jebkuram citam punktam $x \in D$: $cx^* \geq cx$. Ievērojot funkcijas z lineāro veidu, varam pierakstīt vienādību

$$c \cdot \left(\sum_{i=1}^{m} \lambda_i x^i \right) = \sum_{i=1}^{m} \lambda_i (cx^i),$$

tāpēc

$$cx^* = c \sum_{i=1}^{m} \lambda_i x^i = \sum_{i=1}^{m} \lambda_i (cx^i) \leq \sum_{i=1}^{m} \lambda_i (cx^r) = cx^r \sum_{i=1}^{m} \lambda_i = cx^r, \qquad (2.3.1)$$

kur x^r ir tas stūra punkts, kurā mērķa funkcija sasniedz lielāko vērtību starp visiem citiem stūra punktiem, t.i.,

$$cx^r = \max_{1 \leq i \leq m} \{ \, cx^i \, \}.$$

Izteiksmē (2.3.1) iegūts, ka $cx^* \leq cx^r$, bet punkts x^* ir mērķa funkcijas maksimuma punkts, tāpēc $cx^* \geq cx^r$. No šejienes seko, ka $cx^* = cx^r$, t.i., eksistē vismaz viens stūra punkts, kurā mērķa funkcija sasniedz maksimālo vērtību. ■

Teorēma 2.3.2. Ja mērķa funkcija z sasniedz maksimālo vērtību vairākos kopas D punktos, tad šo vērtību tā sasniedz jebkurā šo punktu izliektās čaulas punktā.

Pierādījums. Pieņemsim, ka funkcija z maksimālo vērtību sasniedz punktos $\overline{x}^1, \overline{x}^2, ..., \overline{x}^s$, t.i., $c\overline{x}^i = \max\limits_{x \in D} z = z^*$, $i = 1, 2, ..., s$. Apskatīsim šo punktu patvaļīgu izliektu kombināciju

$$x^* = \sum_{i=1}^{s} \lambda_i \overline{x}^i, \text{ kur } \sum_{i=1}^{s} \lambda_i = 1, \ \lambda_i \geq 0, \ i = 1, 2, ..., s. \tag{2.3.2}$$

Atradīsim mērķa funkcijas vērtību šajā punktā

$$z(x^*) = cx^* = \sum_{i=1}^{s} c(\lambda_i \overline{x}^i) = \sum_{i=1}^{s} \lambda_i (c\overline{x}^i) = \sum_{i=1}^{s} \lambda_i z^* = z^* \sum_{i=1}^{s} \lambda_i = z^*. \ \blacksquare$$

2.4 ATBALSTA PLĀNI

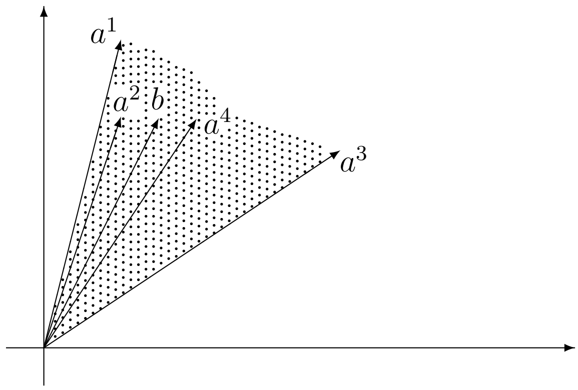
Apskatīsim lineārās programmēšanas kanonisko uzdevumu

$$\max < c, x >$$
$$Ax = b, \quad x \geq 0.$$

Apzīmēsim ar a^j matricas A kolonnas un apskatīsim tās kā telpas $\mathbf{R}^m$ vektorus. Tādā gadījumā jebkurš lineārās programmēšanas uzdevuma kanoniskās formas pieļaujamais plāns (n-dimensiju vektors) ir uztverams kā nenegatīva lineāra kombinācija no kolonnām a^j, kas vienāda ar kolonnu $b \in \mathbf{R}^m$

$$x_1 a^1 + x_2 a^2 + ... + x_n a^n = b, \ x_i \geq 0, \ i = 1, 2, ..., n.$$

Šādu lineārās programmēšanas uzdevuma kanoniskās formas pierakstu sauc par pierakstu *vektoru formā*. Vektorus a^j, $j = 1, 2, ..., n$, sauc par *prasību vektoriem* un vektoru b par *ierobežojumu vektoru*. Kolonnu a^j nenegatīvo lineāro kombināciju kopa no ģeometrijas viedokļa ir uztverama kā izliekts daudzskaldņa konuss, kas "uzvilkts" uz vektoru a^j sistēmas telpā $\mathbf{R}^m$ (piemēram, skatīt 2.4.1.zīm.)



2.4.1. zīm.

Līdz ar to jautājums par uzdevuma pieļaujamā plāna eksistenci ir pielīdzināms jautājumam par vektora b piederību dotajam konusam, bet pieļaujamā plāna $x \in D$ koordinātas x_j ir uztveramas kā izvirzījuma koeficienti pie prasību vektoriem a^j ar uzdevuma ierobežojumu vektoru b.

Šādu lineārās programmēšanas uzdevuma kanoniskās formas reprezentāciju sauc par *lineārās programmēšanas uzdevuma otro ģeometrisko interpretāciju.*

Tālāk mēs pieņemsim, ka vienādojumu skaits lineārās programmēšanas kanoniskajā formā ir mazāks vai vienāds ar mainīgo skaitu ($m \leq n$). Ja tas tā nav, tad sistēma $Ax = b$ ir vai nu nesaderīga (tad pieļaujamo plānu kopa D ir tukša), vai arī satur liekus (lineāri atkarīgus) vienādojumus.

Ja matricas A kaut kādas kolonnas a^{j_1}, a^{j_2},..., a^{j_m} veido lineāri neatkarīgu sistēmu, tad šī sistēma ir telpas $\mathbf{R}^m$ bāze. Šī sistēma ir pietiekoša, lai vektoru b izteiktu kā uzrādīto kolonnu lineāro kombināciju. Tas nozīmē, ka pārējās kolonnas dotajā izvirzījumā būs ar nulles koeficientiem. Ja pie tam lineārās kombinācijas koeficienti būs nenegatīvi , tad mēs iegūsim tā saucamo atbalsta plānu x, kuram ir ne vairāk kā m nenulles koordinātas.

Pieņemsim, ka dots lineārās programmēšanas uzdevums kanoniskā formā

$$\begin{gathered} \max < c, x > \\ Ax = b, \quad x \geq 0. \end{gathered} \tag{2.4.1}$$

Apzīmēsim ar A sistēmas ierobežojumu matricu. Pieņemsim, ka

$$\beta = \{ a^{j_1}, a^{j_2}, ..., a^{j_m} \}$$

ir matricas A lineāri neatkarīgo kolonnu sistēma, kas veido telpas $\mathbf{R}^m$ bāzi.

Apzīmēsim kolonnu, kas ietilpst sistēmā β, indeksu kopu ar

$$N(\beta) = \{j_1, j_2, ..., j_m\}.$$

Definīcija 2.4.1. Dotā uzdevuma (2.4.1) pieļaujamo plānu x sauc par *atbalsta plānu*, ja tā koordinātas, kas atbilst bāzes kolonnām un tiek sauktas par *bāzes koordinātām*, ir lielākas vai vienādas ar 0: $x_j \geq 0$, $j \in N(\beta)$, bet visas pārējās koordinātas (*nebāzes* vai *brīvās koordinātas*) ir vienādas ar 0: $x_j = 0$, $j \notin N(\beta)$.

Definīcija 2.4.2. Atbalsta plānu x sauc par *nedeģenerētu*, ja visas tā bāzes koordinātas ir stingri pozitīvas, un par *deģenerētu* pretējā gadījumā.

Vispirmām kārtām apskatīsim rezultātus ar nedeģenerētiem atbalsta plāniem.

Nākamā teorēma izskaidro atbalsta plāna jēdzienu pirmās ģeometriskās interpretācijas terminos. Šis rezultāts attiecas arī uz deģenerētajiem atbalsta plāniem.

Teorēma 2.4.1. Pieļaujamais plāns ir atbalsta plāns tad un tikai tad, ja tas ir stūra punkts.

Pierādījums. Vispirms pierādīsim, ka jebkurš atbalsta plāns ir pieļaujamo plānu kopas D stūra punkts.

Vienkāršības labad pieņemsim, ka bāzes vektori ir matricas A pirmās m kolonnas, t.i., $\beta = \{\, a^1, a^2, ..., a^m \,\}$. Tad teorēmas apgalvojumu var pārformulēt šādi: ja eksistē tāds n-dimensiju vektors

$$x = (x_1, x_2, ..., x_m, 0, ..., 0), \quad x_j \geq 0, \ j = 1, ..., m,$$

ka $x_1 a^1 + x_2 a^2 + ... + x_m a^m = b$, tad x ir kopas D stūra punkts.

Pierādījumu veiksim no pretējā, t.i., pieņemsim, ka apskatāmais atbalsta plāns nav kopas D stūra punkts. Tādā gadījumā to var izteikt kā divu atšķirīgu pieļaujamo plānu x^1 un x^2 izliektu kombināciju:

$$x = \lambda x^1 + (1 - \lambda) x^2, \quad 0 < \lambda < 1.$$

Koordinātu formā pēdējo apgalvojumu var pierakstīt arī šādi:

$$x_j = \lambda x_j^1 + (1 - \lambda) x_j^2, \quad 0 < \lambda < 1, \quad j = 1, 2, ..., n.$$

Tā kā vektora x pēdējās $(n - m)$ koordinātas pēc pieņēmuma ir vienādas ar 0, bet skaitļi x_j^1, $x_j^2 \geq 0$ un λ, $(1 - \lambda) > 0$, tad tās pašas koordinātas ir vienādas ar 0 arī vektoros x^1 un x^2. Ievērojot, ka x^1 un x^2 ir pieļaujamie plāni, tad ir jāizpildās vienādībām

$$a^1x_1^1 + a^2x_2^1 + ... + a^mx_m^1 = b, \qquad (2.4.2)$$
$$a^1x_1^2 + a^2x_2^2 + ... + a^mx_m^2 = b. \qquad (2.4.3)$$

Atņemot (2.4.3) no (2.4.2), iegūsim

$$a^1(x_1^1 - x_1^2) + a^2(x_2^1 - x_2^2) + ... + a^k(x_m^1 - x_m^2) = 0.$$

Tā kā vektori $a^1, a^2, ..., a^m$ ir lineāri neatkarīgi, tad koeficienti $x_1^1 - x_1^2 = 0$, $x_2^1 - x_2^2 = 0$,..., $x_m^1 - x_m^2 = 0$, no šejienes seko, ka $x^1 = x^2$. Tā ir pretruna ar pieņēmumu, ka x^1 un x^2 ir kopas D divi atšķirīgi punkti.

Tagad pierādīsim pretējo apgalvojumu: ja x ir kopas D stūra punkts, tad tas ir uzdevuma (2.4.1) atbalsta plāns.

Pieņemsim, ka x ir pieļaujamo plānu kopas D stūra punkts. Tātad to nevar izteikt kā divu citu plānu izliektu lineāru kombināciju, ja $0 < \lambda < 1$. Pieņemsim, ka plāna x pirmās s koordinātas ir pozitīvas, bet pārējās $n - s$ ir vienādas ar 0

$$x = (x_1, x_2, ..., x_s, 0, 0, ..., 0).$$

Vispirms apskatīsim gadījumu, kad $s \leq m$. Tā kā x ir pieļaujamais plāns, tad tas apmierina prasību sistēmu, tāpēc

$$\sum_{i=1}^{s} x_i a^i = b. \qquad (2.4.4)$$

Ja vektoru sistēma a^i, $i = 1, 2, ..., s$, ir lineāri neatkarīga, tad saskaņā ar Definīciju 2.4.1 plāns x ir atbalsta plāns, un pierādījums pabeigts.

Pieņemsim, ka vektoru sistēma a^i, $i = 1, 2, ..., s$, nav lineāri neatkarīga. Tad saskaņā ar vektoru sistēmas lineārās atkarības definīciju summā

$$\sum_{i=1}^{s} \alpha_i a^i = 0 \qquad (2.4.5)$$

vismaz viens no koeficientiem α_i nav nulle.

Pareizināsim (2.4.5) abas puses ar koeficientu $\varepsilon > 0$ un attiecīgi pieskaitīsim un atņemsim no vienādības (2.4.4), iegūsim

$$\begin{cases} \sum\limits_{i=1}^{s} x_i a^i + \varepsilon \sum\limits_{i=1}^{s} \alpha_i a^i = b, \\ \sum\limits_{i=1}^{s} x_i a^i - \varepsilon \sum\limits_{i=1}^{s} \alpha_i a^i = b \end{cases}$$

jeb

$$\begin{cases} \sum\limits_{i=1}^{s}(x_i + \varepsilon\alpha_i)a^i = b, \\ \sum\limits_{i=1}^{s}(x_i - \varepsilon\alpha_i)a^i = b. \end{cases} \tag{2.4.6}$$

Izvēlamies $\varepsilon > 0$ tik mazu, lai izpildītos nosacījumi

$$\begin{cases} x_i + \varepsilon\alpha_i > 0, & i = 1, 2, ..., s, \\ x_i - \varepsilon\alpha_i > 0, & i = 1, 2, ..., s. \end{cases} \tag{2.4.7}$$

Pamatojoties uz pieļaujamo plānu definīciju, no (2.4.6) un (2.4.7) izriet, ka atšķirīgie vektori

$$x^1 = (x_1 + \varepsilon\alpha_1,\ x_2 + \varepsilon\alpha_2,\ ...,\ x_s + \varepsilon\alpha_s,\ 0, 0, ..., 0),$$

$$x^2 = (x_1 - \varepsilon\alpha_1,\ x_2 - \varepsilon\alpha_2,\ ...,\ x_s - \varepsilon\alpha_s,\ 0, 0, ..., 0)$$

ir uzdevuma pieļaujamie plāni. Saskaitot x^1 un x^2, iegūst

$$x^1 + x^2 = 2x \text{ jeb}$$

$$x = \frac{1}{2}x^1 + \frac{1}{2}x^2,$$

t.i., x ir plānu x^1 un x^2 izliekta lineāra kombinācija ar koeficientu $\lambda = \frac{1}{2}$. Tas ir pretrunā ar doto, ka x ir stūra punkts, tāpēc pieņēmums, ka vektoru sistēma a^i, $i = 1, 2, ..., s$, nav lineāri neatkarīga, ir aplams. Tādā gadījumā vektoru sistēma, kura atbilst stūra punkta x pozitīvajām koordinātām, ir lineāri neatkarīga, tāpēc x ir atbalsta plāns.

Ja pieņem, ka $s > m$, tad vektoru sistēma a^i, kas atbilst plāna x pozitīvajām koordinātām, ir lineāri atkarīga, jo matricas A rangs ir m. Varam spriest tieši tāpat kā iepriekš un konstruēt divus pieļaujamus plānus, kuru izliekta lineāra kombinācija ir x. Iegūtā pretruna parāda, ka nevienādība $s > m$ nav iespējama. ■

Teorēma 2.4.2. Ja pieļaujamo plānu kopa nav tukša, tad tā satur vismaz vienu atbalsta plānu.

Pierādījums. Ja pieļaujamo plānu kopa nav tukša, tad tā satur vismaz vienu plānu, piemēram,

$$x = (x_1,\ x_2,\ ...,\ x_m,\ 0,\ ...,\ 0),\ x_i > 0,\ i = 1, 2, ..., m. \tag{2.4.8}$$

Plāna x pozitīvajām koordinātām atbilst kolonnu vektoru sistēma a^i, $i = 1, 2, ..., m$. Pieņemsim, ka no šiem vektoriem r veido lineāri neatkarīgu vektoru sistēmu. Iespējami divi gadījumi:

a) $r = m$, b) $r < m$.

a) Ja $r = m$, tad saskaņā ar Definīciju 2.4.1 plāns x ir atbalsta plāns, teorēmas pierādījums pabeigts.

b) Pieņemsim, ka $r < m$, un pierādīsim, ka bez plāna x eksistē vēl citi pieļaujamie plāni, no kuriem vismaz viens ir atbalsta plāns. Tā kā x ir plāns, tad tas apmierina nosacījumu sistēmu $Ax = b$. Ievērojot (2.4.8), iegūsim

$$\sum_{i=1}^{m} x_i a^i = b. \tag{2.4.9}$$

Tā kā $r < m$, tad vektoru sistēma a^i, $i = 1, 2, ..., m$, ir lineāri atkarīga. Saskaņā ar vektoru sistēmas lineārās atkarības definīciju izteiksmē

$$\sum_{i=1}^{m} \alpha_i a^i = 0 \tag{2.4.10}$$

vismaz viens no koeficientiem α_i nav nulle. Varam pieņemt, ka izteiksmē (2.4.10) ir vismaz viens pozitīvs koeficients (ja ir negatīvs, tad vienādības abas puses var pareizināt ar -1 un dabūt pozitīvu koeficientu). Pareizinām (2.4.10) abas puses ar $\varepsilon > 0$ un attiecīgi atņemam no vienādības (2.4.9), iegūsim

$$\sum_{i=1}^{m} (x_i - \varepsilon\alpha_i) a^i = b. \tag{2.4.11}$$

No pēdējās vienādības (2.4.11) seko, ka katram $\varepsilon > 0$, kas apmierina nevienādības

$$x_i - \varepsilon\alpha_i \geq 0,\ i = 1, 2, ..., m, \tag{2.4.12}$$

atbilst pieļaujams uzdevuma plāns

$$x^\alpha = (x_1 - \varepsilon\alpha_1,\ x_2 - \varepsilon\alpha_2,\ ...,\ x_m - \varepsilon\alpha_m,\ 0,\ ...,\ 0). \tag{2.4.13}$$

Lai izpildītos nevienādības (2.4.12), ε vērtībai jāapmierina nevienādības

$$0 < \varepsilon \leq \min_{i \in \{1,2,...,m\}} \frac{x_i}{\alpha_i}, \quad \alpha_i > 0.$$

Definēsim

$$\varepsilon_0 = \min_{i \in \{1,2,...,m\}} \frac{x_i}{\alpha_i} = \frac{x_s}{\alpha_s}, \quad \alpha_s > 0. \tag{2.4.14}$$

Saskaņā ar (2.4.12) un (2.4.14)

$$x_i - \varepsilon_0 \alpha_i > 0, i = \{1, 2, ..., m\} \setminus \{s\}, \quad x_s - \varepsilon_0 \alpha_s = 0.$$

Rezultātā ir atrasts uzdevuma pieļaujamais plāns x^0 ar $m-1$ pozitīvu koordinātu. Ja vektoru sistēmas a^i, $i = \{1, 2, ..., m\} \setminus \{s\}$, rangs $r = m - 1$, tad x^0 ir atbalsta plāns un teorēmas pierādījums pabeigts. Ja $r < m - 1$, tad var atkārtot iepriekšējos spriedumus tik ilgi, kamēr pēc galīga soļu skaita būs atrasts atbalsta plāns. ■

Ievērojot tikko pierādītās teorēmas un iepriekšējā paragrāfa Teorēmas 2.3.1 un 2.3.2, varam izdarīt secinājumu

Sekas 2.4.1. Ja lineārās programmēšanas uzdevumam eksistē optimālais plāns, tad eksistē arī optimālais atbalsta plāns.

2.5 SIMPLEKSA METODES PAMATETAPI

No iepriekš apskatītajām lineārās programmēšanas uzdevumu īpašībām var izdarīt slēdzienu, ka vispārīgā gadījumā atrisinājuma atrašana reducējama uz pieļaujamo plānu kopas stūra punktu caurskatīšanu vai, kas ir tas pats, atbalsta plānu caurskatīšanu. Jātzīmē, ka tāda pilnā pārlase reālos daudzdimensiju uzdevumos ir iespējama tikai teorētiski, bet praktiski nav realizējama pat pie iespaidīgas skaitļojamās tehnikas. Par lineārās programmēšanas uzdevumu atrisināšanas klasisku metodi ir kļuvusi simpleksa metode, kuru dažkārt literatūrā sauc arī par uzlaboto plānu virknes metodi. Šo metodi 1947.gadā izstrādājis Dž.Dancigs.

No skaitļošanas redzes viedokļa optimalitātes kritērijs simpleksa metodē tiek realizēts ar brīvo jeb nebāzes kolonnas vektoru speciāliem novērtējumiem attiecībā pret esošajiem bāzes vektoriem (kolonnām).

Ērtības labad izveidosim noteiktu apzīmējumu sistēmu. Ņemot vērā, ka simpleksa metode ir iterāciju process, ar q apzīmēsim esošās iterācijas kārtu. Tādā gadījumā bāzes kolonnu kopa, kas tiek iegūta q-tajā iterācijā, tiks apzīmēta ar $\beta^{(q)}$

$$\beta^{(q)} = \{\, a^{j_1},\, a^{j_2}, ...,\, a^{j_m}\, \}.$$

Lai vieglāk atšķirtu iterācijas kārtu no matricas un vektoru koordinātām, iterācijas kārta tiks pierakstīta apaļajās iekavās. Indeksi kolonnām, kuras ietilpst bāzē, tiks apzīmēti ar

$$N(\beta^{(q)}) = \{\, j_1,\, j_2, ...,\, j_m \,\}.$$

Pie tam $N_r(\beta^{(q)}) = j_r$ ir tās kolonnas indekss, kura atrodas bāzes r-tajā pozīcijā. Tad esošais atbalsta plāns x ir izskatā

$$x = (x_1, x_2, ..., x_n), \text{ kur } x_j = 0, \text{ ja } j \notin N(\beta^{(q)}).$$

Ar $\Delta(\beta^{(q)})$ apzīmēsim matricu, kas sastādīta no matricas A kolonnām $\{\, a^{j_1},\, a^{j_2}, ...,\, a^{j_m} \,\}$, kuras veido bāzi. Skaidrs, ka $\Delta(\beta^{(q)})$ ir nedeģenerēta matrica un tai eksistē inversā matrica $\Delta^{-1}(\beta^{(q)})$. Tā kā $x_j = 0$, ja $j \notin N(\beta^{(q)})$, tad

$$Ax = \Delta(\beta^{(q)})\, x^\beta = b,$$

kur $x^\beta = (x_{j_1}, x_{j_2}, ..., x_{j_m})$ — tās x koordinātas, kuras atbilst bāzei. Tādā gadījumā

$$x^\beta = \Delta^{-1}(\beta^{(q)})\, b.$$

Līdz ar to dotā bāze $\beta^{(q)}$ viennozīmīgi nosaka vektoru x^β, no kura savukārt var atrast vektoru $x = (x_1, x_2, ..., x_n)$ šādi

$$x_j = \begin{cases} 0, & j \notin N(\beta^{(q)}), \\ x^\beta_j, & j \in N(\beta^{(q)}), \end{cases} \quad j = 1, 2, ..., n.$$

Par kolonnas a^j paplašināto kolonnu nosauksim tādu kolonnu, kurai klāt pievienots atbilstošais mērķa funkcijas koeficients c_j (tas tiks saukts par paplašinātās kolonnas vektora 0-to koordinātu), tātad

$$\overline{a}^j = (c_j, a^j) = (c_j, a_{1,j}, a_{2,j}, ..., a_{m,j})^T.$$

Paplašinātais ierobežojumu vektors ir

$$\overline{b} = (0, b_1, b_2, ..., b_m)^T.$$

(Ar pakāpi T apzīmējam šeit transponēto vektoru.)

Ar $\overline{\Delta}(\beta^{(q)})$ apzīmēsim matricu, kas sastādīta no paplašinātajām kolonnām un papildināta no kreisās puses ar speciāla veida kolonnu

$$\overline{\Delta}(\beta^{(q)}) = \begin{pmatrix} -1 & c_{j_1} & c_{j_2} & \cdots & c_{j_m} \\ 0 & a_{1,j_1} & a_{1,j_2} & \cdots & a_{1,j_m} \\ & & & \cdots & \\ 0 & a_{m,j_1} & a_{m,j_2} & \cdots & a_{m,j_m} \end{pmatrix}.$$

Matricas kolonnas veido lineāri neatkarīgu vektoru sistēmu telpā $\mathbf{R}^{m+1}$, tāpēc tai eksistē inversā matrica $\overline{\Delta}^{-1}(\beta^{(q)})$.

Matricu

$$\overline{A}(\beta^{(q)}) = \overline{\Delta}^{-1}(\beta^{(q)})\, \overline{A} \tag{$*$}$$

nosauksim par simpleksa tabulu, kas atbilst bāzei $\beta^{(q)}$, kur

$$\overline{A} = \begin{pmatrix} 0 & c_1 & c_2 & ... & c_n \\ b & a^1 & a^2 & ... & a^n \end{pmatrix}.$$

Dotās matricas nultās rindas elementus apzīmēsim ar $a_{0,j}(\beta^{(q)})$, $j = 0, 1, 2, ..., n$ (visu rindu kopumā ar $a_0(\beta^{(q)})$). Pārējos matricas elementus apzīmēsim ar $a_{i,j}(\beta^{(q)})$, $i \in N(\beta^{(q)})$, $j = 0, 1, 2, ..., n$. Kolonnas vektorus bez 0-tā elementa apzīmēsim ar

$$a^i(\beta^{(q)}) = \begin{pmatrix} a_{j_1,i} \\ a_{j_2,i} \\ ... \\ a_{j_m,i} \end{pmatrix}, \quad i = 0, 1, 2, ..., n.$$

Simpleksa tabulai $\overline{A}(\beta^{(q)})$ ir uzskatāma jēga: ja $z, y \in \mathbf{R}^{m+1}$, tad pieraksts $y = \overline{\Delta}(\beta^{(q)})\, z$ nozīmē, ka vektors z reprezentē vektora y koordinātas bāzē

$$\begin{pmatrix} -1 \\ 0 \\ ... \\ 0 \end{pmatrix}, \begin{pmatrix} c_{j_1} \\ a_{1,j_1} \\ ... \\ a_{m,j_1} \end{pmatrix}, \begin{pmatrix} c_{j_2} \\ a_{1,j_2} \\ ... \\ a_{m,j_2} \end{pmatrix} ..., \begin{pmatrix} c_{j_m} \\ a_{1,j_m} \\ ... \\ a_{m,j_m} \end{pmatrix}.$$

No ($*$) varam iegūt formulu $\overline{\Delta}(\beta^{(q)})\, \overline{A}(\beta^{(q)}) = \overline{A}$, kura atklātā veidā izskatās šādi

$$\begin{pmatrix} -1 & c_{j_1} & c_{j_2} & ... & c_{j_m} \\ 0 & a_{1,j_1} & a_{1,j_2} & ... & a_{1,j_m} \\ & & & ... & \\ 0 & a_{m,j_1} & a_{m,j_2} & ... & a_{m,j_m} \end{pmatrix} \cdot \begin{pmatrix} a_{0,0}(\beta^{(q)}) & a_{0,1}(\beta^{(q)}) & ... & a_{0,n}(\beta^{(q)}) \\ a_{j_1,0}(\beta^{(q)}) & a_{j_1,1}(\beta^{(q)}) & ... & a_{j_n,n}(\beta^{(q)}) \\ & & ... & \\ a_{j_m,0}(\beta^{(q)}) & a_{j_m,1}(\beta^{(q)}) & ... & a_{j_m,n}(\beta^{(q)}) \end{pmatrix} =$$

$$= \begin{pmatrix} 0 & c_1 & c_2 & ... & c_n \\ b_1 & a_{11} & a_{12} & ... & a_{1n} \\ & & & ... & \\ b_m & a_{m1} & a_{m2} & ... & a_{mn} \end{pmatrix}.$$

Tad izmantojot iepriekšējos apzīmējumus un pierakstu matricai

$$\overline{\Delta}(\beta^{(q)}) = \begin{pmatrix} -1 & c_{j_1} & c_{j_2} & ... & c_{j_m} \\ 0 & a_{1,j_1} & a_{1,j_2} & ... & a_{1,j_m} \\ & & & ... & \\ 0 & a_{m,j_1} & a_{m,j_2} & ... & a_{m,j_m} \end{pmatrix} = \begin{pmatrix} -1 & c^{\beta} \\ 0 & \beta^{(q)} \end{pmatrix},$$

formulu $\overline{\Delta}(\beta^{(q)})\,\overline{A}(\beta^{(q)}) = \overline{A}$ matricu veidā var noformēt šādi:

$$\begin{pmatrix} -1 & c^{\beta} \\ 0 & \beta^{(q)} \end{pmatrix} \cdot \begin{pmatrix} a_{0,0}(\beta^{(q)}) & a_{0,1}(\beta^{(q)}) & ... & a_{0,n}(\beta^{(q)}) \\ a^0(\beta^{(q)}) & a^1(\beta^{(q)}) & ... & a^n(\beta^{(q)}) \end{pmatrix} = \begin{pmatrix} 0 & c_1 & c_2 & ... & c_n \\ b & a^1 & a^2 & ... & a^n \end{pmatrix}.$$

No matricu reizināšanas seko, ka

(1) $\beta^{(q)}\, a^0(\beta^{(q)}) = b$,

(2) $-a_{0,0}(\beta^{(q)}) + c_{\beta}\, a^0(\beta^{(q)}) = 0$,

(3) $\beta^{(q)}\, a^i(\beta^{(q)}) = a^i,\ i = 1, 2, ..., n$,

(4) $c_{\beta}\, a^i(\beta^{(q)}) - a_{0,i}(\beta^{(q)}) = c_i,\ i = 1, 2, ..., n$.

Pirmā (1) vienādība nozīmē, ka

$$a^0(\beta^{(q)}) = x^{\beta},$$

t.i., $a^0(\beta^{(q)})$ sakrīt ar atbalsta plāna tām koordinātām, kas atbilst bāzei $\beta^{(q)}$. Tā kā atbalsta plāna x pārējās koordinātas ir 0, tad (2) var pierakstīt kā

$$a_{0,0}(\beta^{(q)}) = c_{\beta}\, x^{\beta} = c\, x.$$

Trešā vienādība (3) nozīmē, ka $a^i(\beta^{(q)})$ ir vektora a^i koordinātas bāzē $\beta^{(q)}$; no(4) seko, ka

$$a_{0,i}(\beta^{(q)}) = c_{\beta}\, a^i(\beta^{(q)}) - c_i,\ i = 1, 2, ..., n.$$

Atbalsta plāna optimalitātes kritērijs simpleksa metodē ir šāds:

atbalsta plāns ir optimāls, ja

$$\forall j \in \{1, 2, ..., n\} :\ a_{0,j}(\beta^{(q)}) \geq 0;$$

atbalsta plāns nav optimāls pretējā gadījumā, t.i.,

$$\exists i \in \{1, 2, ..., n\} :\ a_{0,i}(\beta^{(q)}) < 0.$$

Vērtības $a_{0,j}(\beta^{(q)})$ sauc par *matricas A kolonnu novērtējumiem* attiecībā pret esošo bāzi.

Ja esošā bāze nav optimāla, tad simpleksa metode paredz pāreju uz citu bāzi. Tas tiek izdarīts tā, ka viena kolonna tiek izslēgta no bāzes, bet cita nāk tās vietā. Lai nodrošinātu mērķa funkcijas vērtību palielināšanos, bāzē tiek likts tas kolonnas vektors, kuram ir negatīvs novērtējums. Ja tādu kolonnu ir vairāk kā viena, tad tiek rekomendēts izvēlēties to kolonnu, kurai ir lielākais novērtējums pēc absolūtās vērtības. Vienlaicīgi nepieciešams pieņemt lēmumu, kuru kolonnu vajag izslēgt no bāzes. Tas jādara tā, lai jauniegūtā bāze būtu pieļaujama. Doto prasību var viegli ilustrēt gadījumam $m = 2$. Piemēram, 2.4.1.zīmējumā $\{a^2, a^3\}$ veido pieļaujamo bāzi, bet vektoru sistēma $\{a^3, a^4\}$ nav pieļaujama, jo vektora b izvirzījums ar šiem vektoriem saturēs vienu negatīvu plāna koordinātu, kas ir pretrunā ar lineārās programmēšanas kanonisko formu.

Teorēma 2.5.1. Ja atbalsta plānam x^* visu kolonnu vektoru novērtējumi ir nenegatīvi $(a_0(\beta^{(q)}) \geq 0)$, tad tas ir optimālais plāns.

Pierādījums. Pieņemsim, ka $x^* = (x_1^*, x_2^*, ..., x_m^*, 0, ..., 0)$, $x_i^* > 0$, $i = 1, 2, ..., m$, ir nedeģenerēts atbalsta plāns un $a_{0,j}(\beta^{(q)}) \geq 0$, $j = 1, 2, ..., n$. Pieņemsim, ka atbalsta plāna bāzi veido vektori $\beta^{(q)} = \{a^1, a^2, ..., a^m\}$. Tā kā atbalsta plāns x^* apmierina prasību sistēmu $Ax = b$, tad

$$\sum_{i=1}^{m} x_i^* a^i = b. \tag{2.5.1}$$

Mērķa funkcijas vērtība atbalsta plānam x^* ir

$$z(x^*) = \sum_{i=1}^{m} c_i x_i^*. \tag{2.5.2}$$

Izraudzīsimies brīvi patvaļīgu pieļaujamo plānu $x = (x_1, x_2, ..., x_n)$. Tā kā šis plāns ir pieļaujams, tad tas apmierina prasību sistēmu $Ax = b$, t.i.,

$$\sum_{j=1}^{n} x_j a^j = b. \tag{2.5.3}$$

mērķa funkcijas vērtība plānam x ir

$$z(x) = \sum_{j=1}^{n} c_j x_j. \tag{2.5.4}$$

Izteiksim vektorus a^j kā atbalsta plāna x^* bāzes vektoru $\beta^{(q)}$ lineāras kombinācijas

$$\sum_{i=1}^{m} a_{ij}(\beta^{(q)})a^i = a^j,\ j = 1, 2, ..., n. \qquad (2.5.5.)$$

Pēc simpleksa tabulas vektoru a^j novērtējumi ir

$$a_{0,j}(\beta^{(q)}) = \sum_{i=1}^{m} c_i a_{ij}(\beta^{(q)}) - c_j,\ j = 1, 2, ..., n. \qquad (2.5.6)$$

Pēc dotā $a_0(\beta^{(q)}) \geq 0$, tāpēc no (2.5.6) iegūstam nevienādības

$$\sum_{i=1}^{m} c_i a_{ij}(\beta^{(q)}) \geq c_j,\ j = 1, 2, ..., n. \qquad (2.5.7)$$

Pārveidosim (2.5.3), izmantojot (2.5.5)

$$\sum_{j=1}^{n} x_j a^j = \sum_{j=1}^{n} x_j \left(\sum_{i=1}^{m} a_{ij}(\beta^{(q)})a^i \right) = b.$$

Mainām summēšanas kārtību

$$\sum_{i=1}^{m} \left(\sum_{j=1}^{n} x_j\, a_{ij}(\beta^{(q)}) \right) a^i = b. \qquad (2.5.8)$$

Pārveidosim (2.5.4), izmantojot nevienādību (2.5.7)

$$z(x) = \sum_{j=1}^{n} c_j x_j \leq \sum_{j=1}^{n} \left(\sum_{i=1}^{m} c_i a_{ij}(\beta^{(q)}) \right) x_j.$$

Atkal samainīsim summēšanas kārtību

$$z(x) \leq \sum_{i=1}^{m} c_i \left(\sum_{j=1}^{n} x_j a_{ij}(\beta^{(q)}) \right). \qquad (2.5.10)$$

No (2.5.1) un (2.5.8) iegūsim

$$\sum_{j=1}^{n} x_j\, a_{ij}(\beta^{(q)}) = x_i^*,\ i = 1, 2, ..., m. \qquad (2.5.11)$$

Savietojot kopā (2.5.10), (2.5.11) un (2.5.2), iegūsim

$$z(x^*) = \sum_{i=1}^{m} c_i x_i^* = \sum_{i=1}^{m} c_i \left(\sum_{j=1}^{n} x_j a_{ij}(\beta^{(q)}) \right) \geq z(x).$$

Tā kā x var būt jebkurš patvaļīgais plāns, tad x^* ir optimālais plāns. ■

Lemma 2.5.1. Ja vektoru sistēma a^i, $i = 1, 2, ..., m$, ir lineāri neatkarīga, vektors a^n ir šo vektoru lineāra kombinācija, t.i., $a^n = \sum\limits_{i=1}^{m} y_{in} a^i$, pie tam $y_{mn} \neq 0$, tad arī vektoru sistēma a^i, $i = 1, 2, ..., m-1$, un a^n ir lineāri neatkarīga.

Pierādījums. Pierādījumu veiksim no pretējā — ja vektoru sistēma a^i, $i = 1, 2, ..., m-1$, un a^n ir lineāri atkarīga, tad saskaņā ar lineāri atkarīgas vektoru sistēmas definīciju izteiksmē

$$\sum_{i=1}^{m-1} \alpha_i a^i + \alpha_n a^n = 0$$

vismaz viens no koeficientiem α_j, $j = 1, 2, ..., m-1, n$, nav nulle. Pārveidosim šo izteiksmi, ievērojot, ka $a^n = \sum\limits_{i=1}^{m} y_{in} a^i$. Rezultātā iegūsim

$$\sum_{i=1}^{m-1} \alpha_i a^i + \alpha_n \sum_{i=1}^{m} y_{in} a^i = 0 \text{ jeb}$$

$$\sum_{i=1}^{m-1} (\alpha_i + \alpha_n y_{in}) a^i + \alpha_n y_{mn} a^m = 0.$$

Iegūtajā izteiksmē vai nu $\alpha_n y_{mn} \neq 0$, vai $\alpha_i + \alpha_n y_{in} \neq 0$ vismaz vienam indeksam, jo pēc dotā $y_{mn} \neq 0$ un pēc pieņēmuma vai nu $\alpha_n \neq 0$, vai $\alpha_i \neq 0$ vismaz vienam indeksam. Līdz ar to vektoru sistēma a^i, $i = 1, 2, ..., m$, ir lineāri atkarīga, kas ir pretrunā ar doto. Tāpēc varam secināt, ka pieņēmums ir nepareizs un vektoru sistēmai a^i, $i = 1, 2, ..., m-1$, un a^n ir jābūt lineāri neatkarīgai. ■

Teorēma 2.5.2. Ja atbalsta plānam x vismaz vienas kolonnas vektora novērtējums ir negatīvs, tad iespējams atrast jaunu atbalsta plānu x' ar lielāku mērķa funkcijas vērtību kā dotajam plānam x vai arī var secināt, ka mērķa funkcija ir neierobežota.

Pierādījums. Pieņemsim, ka dots atbalsta plāns (ar n koordinātām)

$$x = (x_1,\, x_2,\, ...,\, x_m,\, 0,\, ...,\, 0), \quad x_i > 0,\ i = 1, 2, ..., m.$$

Pieņemsim, ka šī atbalsta plāna bāzi veido vektoru sistēma $\beta^{(q)} = \{a^1,\, a^2,\, ...,\, a^m\}$. Atbalsta plāns x apmierina prasību sistēmu $Ax = b$:

$$\sum_{i=1}^{m} x_i a^i = b, \tag{2.5.12}$$

tā mērķa funkcijas vērtība ir

$$z(x) = \sum_{i=1}^{m} c_i x_i. \tag{2.5.13}$$

Izteiksim vektorus a^j kā atbalsta plāna x bāzes vektoru $\beta^{(q)}$ lineāras kombinācijas $\sum\limits_{i=1}^{m} a_{ij}(\beta^{(q)})a^i = a^j,\ j = 1, 2, ..., n$. Pēc simpleksa tabulas vektoru a^j novērtējumi ir $a_{0,j}(\beta^{(q)}) = \sum\limits_{i=1}^{m} c_i a_{ij}(\beta^{(q)}) - c_j,\ j = 1, 2, ..., n$. Pēc dotā vismaz viena vektora novērtējums ir negatīvs. Pieņemsim, ka

$$a_{0,n}(\beta^{(q)}) = \sum_{i=1}^{m} c_i a_{in}(\beta^{(q)}) - c_n < 0.$$

Lai aprēķinātu $a_{0,n}(\beta^{(q)})$, izmanto kolonnas vektora a^n izvirzījuma koeficientus, kurus atrod no izteiksmes

$$\sum_{i=1}^{m} a_{in}(\beta^{(q)})a^i = a^n. \tag{2.5.14}$$

Iespējami divi gadījumi:

a) vismaz viens koeficients $a_{in}(\beta^{(q)})$ ir pozitīvs, pieņemsim, ka tas ir $a_{mn}(\beta^{(q)}) > 0$;

b) $a_{in}(\beta^{(q)}) \leq 0,\ i = 1, 2, ..., m$.

Izpētīsim šos abus gadījumus precīzāk.

a) gadījums. Reizinām vienādības (2.5.14) abas puses ar skaitli ε un pēc tam atņemam attiecīgi no (2.5.12). Rezultātā iegūsim

$$\sum_{i=1}^{m} x_i a^i - \varepsilon \sum_{i=1}^{m} a_{in}(\beta^{(q)})a^i = b - \varepsilon a^n \quad \text{jeb}$$

$$\sum_{i=1}^{m}(x_i - \varepsilon a_{in}(\beta^{(q)}))a^i + \varepsilon a^n = b. \qquad (2.5.15)$$

Ja $\varepsilon > 0$ un $x_i - \varepsilon a_{in}(\beta^{(q)}) \geq 0$, $i = 1, 2, ..., m$, tad no (2.5.15) seko, ka

$$x_\varepsilon = (x_1 - \varepsilon a_{1n}(\beta^{(q)}),\ x_2 - \varepsilon a_{2n}(\beta^{(q)}),\ ...,\ x_m - \varepsilon a_{mn}(\beta^{(q)}),\ 0, ...,\ 0,\ \varepsilon),$$

kur 0 ir skaitā $n - (m + 1)$, ir pieļaujamais plāns.

Lai izpildītos nosacījumi

$$x_i - \varepsilon a_{in}(\beta^{(q)}) \geq 0, \quad i = 1, 2, ..., m, \quad \varepsilon > 0, \qquad (2.5.16)$$

konstante ε jāizvēlas šādi

$$0 < \varepsilon \leq \min_i \frac{x_i}{a_{in}(\beta^{(q)})}, \quad a_{in}(\beta^{(q)}) > 0.$$

Pieņemsim, ka

$$\varepsilon_0 = \min_i \frac{x_i}{a_{in}(\beta^{(q)})} = \frac{x_m}{a_{mn}(\beta^{(q)})}, \quad a_{mn}(\beta^{(q)}) > 0. \qquad (2.5.17)$$

Ja mēs izvēlamies šo ε_0 pēc formulām (2.5.17), tad

$$x_i - \varepsilon_0 a_{in}(\beta^{(q)}) > 0, i = 1, 2, ..., m - 1, \quad x_m - \varepsilon a_{mn}(\beta^{(q)}) = 0. \qquad (2.5.18)$$

Plānu, kuru iegūsim ar ε_0, apzīmēsim ar x'

$$x' = (x'_1,\ x'_2,\ ...,\ x'_{m-1},\ 0,\ ...,\ 0,\ x'_n), \text{ kur}$$

$$x'_i = x_i - \varepsilon_0 a_{in}(\beta^{(q)}), i = 1, 2, ..., m - 1, \quad x'_n = \varepsilon_0 = \frac{x_m}{a_{mn}(\beta^{(q)})}. \qquad (2.5.19)$$

Plāna x' pozitīvajām koordinātām atbilst vektori a^i, $i = 1, 2, ..., m - 1$, a^n. Vektoru sistēma a^i, $i = 1, 2, ..., m$, ir lineāri neatkarīga, jo tā ir atbalsta plāna x bāze. Vektors a^n ir bāzes vektoru lineāra kombinācija, un pēc dotā $a_{mn}(\beta^{(q)}) > 0$. Ņemot vērā Lemmu 2.5.1, arī vektoru sistēma a^i, $i = 1, 2, ..., m - 1$, a^n ir lineāri neatkarīga, tāpēc x' ir atbalsta plāns.

Aprēķināsim mērķa funkcijas vērtību šim atbalsta plānam x'

$$z(x') = \sum_{i=1}^{m-1} c_i x'_i + c_n x'_n. \qquad (2.5.20)$$

Pārveidosim (2.5.20), izmantojot (2.5.19), (2.5.13) un kolonnu vektoru novērtējumus $a_{0,j}(\beta^{(q)}) = \sum_{i=1}^{m} c_i a_{ij}(\beta^{(q)}) - c_j,\ j = 1, 2, ..., n,$

$$z(x') = \sum_{i=1}^{m-1} c_i \left(x_i - \tfrac{x_m}{a_{mn}(\beta^{(q)})} a_{in}(\beta^{(q)}) \right) + c_n \tfrac{x_m}{a_{mn}(\beta^{(q)})} + c_m x_m - c_m x_m =$$
$$= \sum_{i=1}^{m-1} c_i x_i + c_m x_m - \tfrac{x_m}{a_{mn}(\beta^{(q)})} \left(\sum_{i=1}^{m-1} c_i a_{in}(\beta^{(q)}) + c_m a_{mn}(\beta^{(q)}) - c_n \right) =$$
$$= \sum_{i=1}^{m} c_i x_i - \tfrac{x_m}{a_{mn}(\beta^{(q)})} \left(\sum_{i=1}^{m} c_i a_{in}(\beta^{(q)}) - c_n \right) = z(x) - \tfrac{x_m}{a_{mn}(\beta^{(q)})} a_{0n}(\beta^{(q)}).$$

Tātad

$$z(x') = z(x) - \frac{x_m}{a_{mn}(\beta^{(q)})} a_{0n}(\beta^{(q)}) \text{ jeb} \tag{2.5.21}$$

$$z(x') = z(x) - \varepsilon_0 a_{0n}(\beta^{(q)}). \tag{2.5.22}$$

Tā kā $\varepsilon_0 > 0$ un $a_{0n}(\beta^{(q)}) < 0$, tad no (2.5.22) iegūsim nevienādību

$$z(x') > z(x).$$

Tas nozīmē, ka esam atraduši jaunu atbalsta plānu x' ar lielāku mērķa funkcijas vērtību nekā sākumā dotajam atbalsta plānam x.

b) gadījums. Ja $a_{0n}(\beta^{(q)}) < 0$ un $a_{in}(\beta^{(q)}) \leq 0,\ i = 1, 2, ..., m,$ tad (2.5.16) izpildās jebkuram $\varepsilon > 0$. Šajā gadījumā plānam x_ε ir $m+1$ pozitīva koordināta

$$x_i - \varepsilon a_{in}(\beta^{(q)}) > 0,\ i = 1, 2, ..., m, \quad \varepsilon > 0.$$

Izvēloties pēc patikas lielu ε, var atrast plānu ar neierobežotu mērķa funkcijas vērtību. Tas nozīmē, ka uzdevumam nav atrisinājuma. ■

Tātad, lai nākamā bāze būtu pieļaujama, jāievēro esošās bāzes kolonnas izslēgšanas šāds likums:

atbilstošajai kolonnai l, kura pretendē uz iekļaušanu bāzē, un ierobežojumu vektoram b tiek apskatīti skaitļi

$$\lambda_i = \frac{b_i(\beta^{(q)})}{a_{i,l}(\beta^{(q)})},\ \text{kur } i \in \{1, ..., m \,|\, a_{i,l}(\beta^{(q)}) > 0\,\},$$

un tiek atrasta tā rinda r, kurai

$$\lambda_r = \min_i \{\lambda_i\}.$$

Iegūtais indekss r definē kolonnas indeksu sistēmā $N(\beta^{(q)})$, kuru nepieciešams izslēgt no bāzes.

Tādējādi, ja q-tajā iterācijā bāzē bija iekļautas kolonnas ar indeksiem

$$N(\beta^{(q)}) = \{j_1, j_2, ..., j_{r-1}, j_r, j_{r+1}, ..., j_m\},$$

tad $q+1$ iterācijā bāze tiks izveidota no kolonnām ar indeksiem

$$N(\beta^{(q+1)}) = \{j_1, j_2, ..., j_{r-1}, j_l, j_{r+1}, ..., j_m\}.$$

Atsevišķi jāapskata gadījums, ja kolonna $a^l(\beta^{(q)})$, kas pretendē uz iekļaušanu bāzē, nesatur pozitīvas koordinātas, t.i., $a^l(\beta^{(q)}) \leq 0$. Tas nozīmē, ka uzdevuma mērķa funkcija nav ierobežota pieļaujamo vērtību kopā, t.i., var sasniegt pēc patikas lielu vērtību. Tas nozīmē, ka skaitļošanas process tiek pabeigts, jo nav optimālā plāna. Situācija, ja $a^l(\beta^{(q)}) \leq 0$, ğeometriski atbilst tam, ka ordinātu ass atrodas vidū konusam, kurš veidojas kā paplašināto vektoru $\overline{a}^j$ sistēmas izliektā čaula, bet taisne, kas novilkta caur vektora b galapunktu paralēli aplikātas asij, vienreiz "ieejot" konusā, vairs to nekad neatstāj.

Vispārīgi runājot, pēc pārejas no bāzes $\beta^{(q)}$ uz bāzi $\beta^{(q+1)}$ mēs varam no jauna noformēt matricas $\overline{\Delta}(\beta^{(q+1)})$, $\overline{\Delta}^{-1}(\beta^{(q+1)})$ un, izskaitļojot $\overline{A}(\beta^{(q+1)}) = \overline{\Delta}^{-1}(\beta^{(q+1)})\,\overline{A}$, izdarīt slēdzienu par bāzes optimalitāti. Taču $\beta^{(q+1)}$ no $\beta^{(q)}$ atšķiras tikai ar vienu kolonnu, tāpēc no skaitļošanas tehnikas viedokļa racionālāk ir uzreiz pāriet no $\overline{A}(\beta^{(q)})$ un $\overline{b}(\beta^{(q)})$ uz $\overline{A}(\beta^{(q+1)})$ un $\overline{b}(\beta^{(q+1)})$. Tas ir iespējams, jo tāda tipa matricai kā $\overline{A}(\beta^{(q)})$ kolonnas, kas atbilst bāzes vektoriem, sastāv no nullēm, izņemot vienu elementu, kas vienāds ar 1. Šī vieninieka atrašanās vieta ir nodefinēta ar kolonnas indeksu bāzē $N(\beta^{(q)})$. Tāpēc, lai iegūtu matricu $\overline{A}(\beta^{(q+1)})$, ir pietiekami veikt lineāras operācijas ar matricas $\overline{A}(\beta^{(q)})$ rindām tā, lai atbilstošo kolonnu pārveidotu "bāzes" vektora izskatā.

Šajā nolūkā tiek izmantota Žordāna-Gausa metode. Dotajā gadījumā tas nozīmē to, ka mums nepieciešams ar šiem pārveidojumiem dabūt skaitli 1 koordinātas $a_{r,l}(\beta^{(q)})$ vietā (koordinātu $a_{r,l}(\beta^{(q)})$ šādā gadījumā sauc par *vadošo*) un 0 visu pārējo kolonnas $a^l(\beta^{(q)})$ koordinātu vietās. Vieninieku dabūsim, ja dalīsim r-to rindu ar vadošo koordinātu, bet nulles iegūsim, ja jauniegūto r-to rindu pareizināsim ar atbilstošiem koeficientiem un atskaitīsim šīs rindas no atbilstoštošām matricas $\overline{A}(\beta^{(q)})$ rindām.

Dotos pārveidojumus ar $\overline{A}(\beta^{(q)})$ un $\overline{b}(\beta^{(q)})$ elementiem formāli var aprakstīt ar šādām formulām:

$$a_{r,j}(\beta^{(q+1)}) = \frac{a_{r,j}(\beta^{(q)})}{a_{r,l}(\beta^{(q)})},\ j = 1, ..., n; \tag{2.5.1}$$

$$b_r(\beta^{(q+1)}) = \frac{b_r(\beta^{(q)})}{a_{r,l}(\beta^{(q)})}; \tag{2.5.2}$$

$$a_{i,j}(\beta^{(q+1)}) = a_{i,j}(\beta^{(q)}) - a_{i,l}(\beta^{(q)})\frac{a_{r,j}(\beta^{(q)})}{a_{r,l}(\beta^{(q)})},\ i = 0, 1, ..., m,\ i \neq r,\ j = 1, ..., n; \tag{2.5.3}$$

$$b_i(\beta^{(q+1)}) = b_i(\beta^{(q)}) - a_{i,l}(\beta^{(q)})\frac{b_r(\beta^{(q)})}{a_{r,l}(\beta^{(q)})},\ i = 0, 1, ..., m,\ i \neq r. \tag{2.5.4}$$

Īpaši jāatzīmē vektora $\overline{b}(\beta^{(q)})$ koordinātu nozīme. Tā nultā koordināta $\overline{b}_0(\beta^{(q)})$ saskaņā ar konstrukciju satur mērķa funkcijas vērtību, ko tā sasniedz pie esošā plāna

$$z(x(\beta^{(q)}) = \overline{b}_0(\beta^{(q)}), \tag{2.5.5}$$

bet pārējās koordinātas ir šī plāna nenulles koordinātas

$$x_{N_i(\beta^{(q)})}(\beta^{(q)}) = \overline{b}_i(\beta^{(q)}),\ i = 1, ..., m. \tag{2.5.6}$$

Paragrāfa noslēgumā dosim slēdzienu par iepriekš apskatītajiem jautājumiem: *maksimizācijas uzdevuma atrisināšanas ar simpleksa metodi algoritma shēma.* Šī shēma ietver vienreizēju 0-tā etapa izpildi un atkārtotu galīgā skaitā I-etapu (standarta iterāciju).

0-etaps. *Atbalsta plāna atrašana.* 0-etapa rezultāts ir atbalsta plāns $x(\beta^{(q)})$, tam atbilstošā matrica $\overline{A}(\beta^{(q)})$ un vektors $\overline{b}(\beta^{(q)})$, kuri tiks tālāk izmantoti pirmajā iterācijā. Definējam tekošo iterāciju $q = 1$ un pārejam uz I-etapu.

I-etaps. *Algoritma standartiterācija* — tiek izpildīta kārtējam atbalsta plānam $x(\beta^{(q)})$.

1°. Tekošā atbalsta plāna optimalitātes pārbaude: tiek veikta rindas novērtējumu $a_0(\beta^{(q)})$ caurskatīšana. Iespējami divi varianti:

1'. $a_0(\beta^{(q)}) \geq 0$ — plāns, kas atbilst esošajai uzdevuma bāzei, ir **optimāls**. Skaitļošanas process ir pabeigts. Saskaņā ar formulām (2.5.5) un (2.5.6) ir atrodams uzdevuma optimālais plāns $x^* = x(\beta^{(q)})$ un mērķa funkcijas vērtība $z(x^*) = z(x(\beta^{(q)}))$.

1". novērtējumu rindā $a_0(\beta^{(q)})$ eksistē vismaz viens elements $a_{0,j}(\beta^{(q)}) < 0$. Seko, ka plāns $x(\beta^{(q)})$ **nav optimāls**. Tiek izvēlēta kolonna ar indeksu l, kurai ir mazākais negatīvais novērtējums (vai maksimālais pēc absolūtās vērtības)

$$a_{0,l}(\beta^{(q)}) = \min_{j:\, a_{0,j}(\beta^{(q)})<0} \{a_{0,j}(\beta^{(q)})\}.$$

Šo kolonnu sauc par *vadošo* un tā ir jāiekļauj tekošajā bāzē. Jāpāriet uz algoritma punktu 2°.

2°. *Kolonnas, kura izslēdzama no bāzes, noteikšana.* Tiek apskatīta vadošā kolonna $a^l(\beta^{(q)})$. Iespējami divi varianti:

2'. Visiem $i = 1, 2, ..., m$: $a_{i,l}(\beta^{(q)}) \leq 0$. Tiek izdarīts slēdziens par *mērķa funkcijas neierobežotību* un skaitļošanas process tiek pabeigts.

2". Eksistē vismaz viena rinda ar indeksu $i \in \{1, ..., m\}$, kurai $a_{i,l}(\beta^{(q)}) > 0$. Saskaņā ar likumu

$$\lambda_r = \min_{i\in\{1,...,m|a_{i,l}(\beta^{(q)})>0\}} \frac{b_i(\beta^{(q)})}{a_{i,l}(\beta^{(q)})}$$

tiek atrasts r un tās kolonnas indekss $N_r(\beta^{(q)}) = j_r$, kura tiks izslēgta no bāzes. Jāpāriet uz algoritma punktu 3°.

3°. *Matricas $\overline{A}$ un kolonnas $\overline{b}$ elementu caurskatīšana* attiecībā pret jauno bāzi. Saskaņā ar formulām (2.5.1) – (2.5.4) veicam matricas $\overline{A}$ un kolonnas $\overline{b}$ pārrēķināšanu attiecībā pret jauno bāzi $\beta^{(q+1)}$, kura izveidojas, izslēdzot kolonnu a^r un iekļaujot kolonnu a^l bāzē $\beta^{(q)}$. Definējam tekošo iterāciju $q := q + 1$ un pārejam uz algoritma pirmo soli.

2.5.1 Simpleksa metodes realizācija tabulas veidā

Lai nodrošinātu simpleksa metodes algoritma racionalitāti un uzskatāmību, to ērti pierakstīt tabulu virknes veidā. Atšķirīgos literatūras avotos tabulas tiek noformētas dažādi. Tomēr tas nedrīkstētu pārāk mulsināt lasītāju, jo tabulas visos gadījumos tiek veidotas pēc vieniem un tiem pašiem principiem. Šajā darbā mēs izmantosim tabulu, kas vispārīgā veidā izskatīsies šādi (2.5.1.zīmējums):

<table>
<tr><td rowspan="2">$T^{(q)} =$</td><td></td><td>$b_0(\beta^{(q)})$</td><td>$a_0(\beta^{(q)})$</td></tr>
<tr><td>$N(\beta^{(q)})$</td><td>$b(\beta^{(q)})$</td><td>$A(\beta^{(q)})$</td></tr>
</table>

2.5.1. zīm.

Simpleksa tabula $T^{(q)}$, kas attēlota 2.5.1.zīmējumā, atbilst lineārās programmēšanas uzdevuma kanoniskās formas pieļaujamajai bāzei $\beta^{(q)}$, kas iegūta q-tajā iterācijā. Kolonna $N(\beta^{(q)})$ satur bāzes kolonnu indeksus (tajā secībā, kādā tie ieiet bāzē), kolonna $b(\beta^{(q)})$ satur ierobežojuma vektora koordinātas attiecībā pret tekošo bāzi $\beta^{(q)}$, $A(\beta^{(q)})$ satur uzdevuma matricas komponentes attiecībā pret tekošo bāzi $\beta^{(q)}$. Rindā $a_0(\beta^{(q)})$ atrodas kolonnu tekošie novērtējumi, bet tabulas rūtiņa $b_0(\beta^{(q)})$ satur mērķa funkcijas vērtību, kas tiek sasniegta ar esošo plānu.

Piemērs 2.5.1. Atrisināsim lineārās programmēšanas uzdevumu kanoniskajā formā

$$\max(50x_1 - 10x_2 + 6x_3 + 40x_4 - 30x_5) \tag{2.5.7}$$

$$\begin{array}{rrrrrl} x_1 & & & +x_4 & +3x_5 & = 12, \\ 2x_1 & +x_2 & & +3x_4 & & = 14, \\ -2x_1 & & +3x_3 & -4x_4 & & = 17, \end{array} \tag{2.5.8}$$

$$x_1,\, x_2,\, x_3,\, x_4,\, x_5 \geq 0.$$

Kā redzams no uzdevuma, tad matricas kolonnas ar indeksiem $\{2, 3, 5\}$ ir lineāri neatkarīgas. Lai labāk saprastu tālākos pierakstus, varam nosacījumu sistēmu pierakstīt šādi:

$$\begin{array}{rrrrrl} 2x_1 & +x_2 & & +3x_4 & & = 14, \\ -2x_1 & & +3x_3 & -4x_4 & & = 17, \\ x_1 & & & +x_4 & +3x_5 & = 12. \end{array}$$

Lineārās neatkarības dēļ ir iespējams ierobežojuma vektora izvirzījums ar dotajām kolonnām ar pozitīviem koeficientiem. Tas savukārt nozīmē, ka kolonnas $\{2, 3, 5\}$ veido telpas $\mathbf{R}^3$ bāzi. No kolonnām, kuras ieiet bāzē, tiek izveidota matrica $\overline{\Delta}(\beta^{(1)})$

$$\overline{\Delta}(\beta^{(1)}) = \begin{pmatrix} -1 & -10 & 6 & -30 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 3 & 0 \\ 0 & 0 & 0 & 3 \end{pmatrix}.$$

Lai atrastu tās inverso $\overline{\Delta}^{-1}(\beta^{(1)})$ matricu, atgādināsim, kā vispār tiek atrasta dotas matricas M inversā matrica. Viens no iespējamajiem variantiem ir šādas formulas izmantošana

$$M^{-1} = \frac{1}{det\,M}\begin{pmatrix} M_{11} & M_{21} & ... & M_{n1} \\ M_{12} & M_{22} & ... & M_{n2} \\ & & ... & \\ M_{1n} & M_{2n} & ... & M_{nn} \end{pmatrix},$$

kur M_{ij} ir matricas M elementa m_{ij} algebriskais papildinājums

$$M_{ij} = (-1)^{i+j}\begin{vmatrix} m_{11} & ... & m_{1\,j-1} & m_{1\,j+1} & ... & m_{1n} \\ m_{21} & ... & m_{2\,j-1} & m_{2\,j+1} & ... & m_{2n} \\ & & & & ... & \\ m_{i-1\,1} & ... & m_{i-1\,j-1} & m_{i-1\,j+1} & ... & m_{i-1\,n} \\ m_{i+1\,1} & ... & m_{i+1\,j-1} & m_{i+1\,j+1} & ... & m_{i+1\,n} \\ & & & & ... & \\ m_{n1} & ... & m_{n\,j-1} & m_{n\,j+1} & ... & m_{nn} \end{vmatrix}, i, j = 1, 2, ..., n,$$

(t.i., $(-1)^{i+j}$ pareizināts ar $(n-1)$-mās kārtas determinantu, kas iegūts, matricas M determinantā izsvītrojot to rindu un to kolonnu, kas satur elementu m_{ij} (i-to rindu un j-to kolonnu)).

Matricas $\overline{\Delta}(\beta^{(1)})$ inversā matrica ir

$$\overline{\Delta}^{-1}(\beta^{(1)}) = \begin{pmatrix} -1 & -10 & 2 & -10 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & \frac{1}{3} & 0 \\ 0 & 0 & 0 & \frac{1}{3} \end{pmatrix}.$$

Izmantojot formulu $\overline{A}(\beta^{(q)}) = \overline{\Delta}^{-1}(\beta^{(q)})\overline{A}$, iegūsim

$$\overline{A}(\beta^{(1)}) = \overline{\Delta}^{-1}(\beta^{(1)})\overline{A} =$$

$$= \begin{pmatrix} -1 & -10 & 2 & -10 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & \frac{1}{3} & 0 \\ 0 & 0 & 0 & \frac{1}{3} \end{pmatrix} \times \begin{pmatrix} 0 & 50 & -10 & 6 & 40 & -30 \\ 14 & 2 & 1 & 0 & 3 & 0 \\ 17 & -2 & 0 & 3 & -4 & 0 \\ 12 & 1 & 0 & 0 & 1 & 3 \end{pmatrix} =$$

$$= \begin{pmatrix} -226 & -84 & 0 & 0 & -88 & 0 \\ 14 & 2 & 1 & 0 & 3 & 0 \\ \frac{17}{3} & -\frac{2}{3} & 0 & 1 & -\frac{4}{3} & 0 \\ 4 & \frac{1}{3} & 0 & 0 & \frac{1}{3} & 1 \end{pmatrix}.$$

Izmantojot iegūtās vērtības $\overline{A}(\beta^{(1)})$, varam aizpildīt tabulu $T^{(1)}$

$$T^{(1)} = \begin{array}{c|c|cccccc} & \text{-226} & \text{-84} & 0 & 0 & \text{-88} & 0 \\ \hline 2 & 14 & 2 & 1 & 0 & \mathbf{3} & 0 \\ 3 & \frac{17}{3} & -\frac{2}{3} & 0 & 1 & -\frac{4}{3} & 0 \\ 5 & 4 & \frac{1}{3} & 0 & 0 & \frac{1}{3} & 1 \end{array}$$

Tā kā novērtējumu rindas $a_0(\beta^{(1)})$ pirmās un ceturtās kolonnas elementi ir negatīvi $a_{0,1}(\beta^{(1)}) = -84$, $a_{0,4}(\beta^{(1)}) = -88$, tad plāns $x(\beta^{(1)}) = (0, 14, \frac{17}{3}, 0, 4)$ nav optimāls un mērķa funkcijas vērtība $z(x(\beta^{(1)})) = -226$ var tikt uzlabota. Sekojot simpleksa metodes algoritma 1" .punkta rekomendācijām, par bāzē iekļaujamās kolonnas indeksu izvēlēsimies $l = 4$ (jo $|-88| > |-84|$). Apskatīsim *vadošo* kolonnu

$$a^4(\beta^{(1)}) = \left(\begin{array}{c} 3 \\ -\frac{4}{3} \\ \frac{1}{3} \end{array} \right).$$

Tā satur divas pozitīvas koordinātas. Izmantojot algoritma 2".punkta rekomendāciju, iegūsim $\lambda_1 = \frac{14}{3}$, $\lambda_3 = \frac{4}{\frac{1}{3}} = 12$, tāpēc $r = 1$. Tādēļ kolonna ar indeksu $N_2(\beta^{(1)}) = 2$ tiks izslēgta no bāzes. Tādējādi iegūsim kārtējo uzdevuma atbalsta plānu $N(\beta^{(2)}) = \{4, 3, 5\}$ Elements $a_{1,4}(\beta^{(1)}) = 3$ ir vadošais. Izmantojot formulas (2.5.1)-(2.5.4), veicam pāreju uz nākošo simpleksa tabulu, kas atbilst otrajai iterācijai $T^{(2)}$ (definējam q=2).

$$T^{(2)} = \begin{array}{c|c|cccccc} & \frac{554}{3} & \text{-}\frac{76}{3} & \frac{88}{3} & 0 & 0 & 0 \\ \hline 4 & \frac{14}{3} & \mathbf{\frac{2}{3}} & \frac{1}{3} & 0 & 1 & 0 \\ 3 & \frac{107}{9} & \frac{2}{9} & \frac{4}{9} & 1 & 0 & 0 \\ 5 & \frac{22}{9} & \frac{1}{9} & -\frac{1}{9} & 0 & 0 & 1 \end{array}$$

Otrās iterācijas galarezultāts rāda, ka bāzē ir jāiekļauj pirmā kolonna a^1 un jāizslēdz ceturtā a^4, vadošais elements ir $a_{1,1}(\beta^{(2)}) = \frac{2}{3}$. Veicot iterāciju $q = 3$, iegūsim tabulu $T^{(3)}$.

$$T^{(3)} = \begin{array}{c|c|cccccc} & 362 & 0 & 42 & 0 & 38 & 0 \\ \hline 1 & 7 & 1 & \frac{1}{2} & 0 & \frac{3}{2} & 0 \\ 3 & \frac{31}{3} & 0 & \frac{1}{3} & 1 & -\frac{1}{3} & 0 \\ 5 & \frac{5}{3} & 0 & -\frac{1}{6} & 0 & -\frac{1}{6} & 1 \end{array}$$

Kā redzams no $T^{(3)}$, novērtējumu rinda satur tikai nenegatīvas vērtības, tāpēc iegūtā bāze $N(\beta^{(3)}) = \{1, 3, 5\}$ ir optimālā. Galarezultātā esam ieguvuši, ka uzdevuma (2.5.7)-(2.5.8) optimālais plāns ir

$$x^* = (7, 0, \frac{31}{3}, 0, \frac{5}{3}),$$

mērķa funkcijas maksimālā vērtība ir

$$z^* = z(x^*) = 362. \blacksquare$$

Piemērs 2.5.2. Atrisināsim lineārās programmēšanas uzdevumu kanoniskajā formā

$$\max(5x_1 - 2x_2 - x_3 + 7x_4 - 14x_5) \tag{2.5.9}$$

$$\begin{array}{rrrrrl} 4x_1 & +x_2 & & +x_4 & -3x_5 & = 8, \\ -7x_1 & & +x_3 & -2x_4 & +4x_5 & = 3, \end{array} \tag{2.5.10}$$

$$x_1,\ x_2,\ x_3,\ x_4,\ x_5 \geq 0.$$

Matricas kolonnas ar indeksiem $\{2, 3\}$ ir lineāri neatkarīgas, t.i., kolonnas ar indeksiem $N(\beta^{(1)}) = \{2, 3\}$ veido bāzes kolonnas. No kolonnām, kuras ieiet bāzē, tiek izveidota matrica $\overline{\Delta}(\beta^{(1)})$ un tās inversā $\overline{\Delta}^{-1}(\beta^{(1)})$

$$\overline{\Delta}(\beta^{(1)}) = \begin{pmatrix} -1 & -2 & -1 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}, \quad \overline{\Delta}^{-1}(\beta^{(1)}) = \begin{pmatrix} -1 & -2 & -1 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}.$$

Atrodam matricu $\overline{A}(\beta^{(1)})$:

$$\overline{A}(\beta^{(1)}) = \overline{\Delta}^{-1}(\beta^{(1)})\overline{A} =$$

$$= \begin{pmatrix} -1 & -2 & -1 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \times \begin{pmatrix} 0 & 5 & -2 & -1 & 7 & -14 \\ 8 & 4 & 1 & 0 & 1 & -3 \\ 3 & -7 & 0 & 1 & -2 & 4 \end{pmatrix} =$$

$$= \begin{pmatrix} -19 & -6 & 0 & 0 & -7 & 16 \\ 8 & 4 & 1 & 0 & 1 & -3 \\ 3 & -7 & 0 & 1 & -2 & 4 \end{pmatrix}.$$

Varam aizpildīt tabulu $T^{(1)}$

$$T^{(1)} = \begin{array}{c|c|ccccc} & -19 & -6 & 0 & 0 & -7 & 16 \\ \hline 2 & 8 & 4 & 1 & 0 & \mathbf{1} & -3 \\ 3 & 3 & -7 & 0 & 1 & -2 & 4 \end{array}$$

Tā kā $-7 < -6$, tad vadošā ir 4.kolonna. Šajā kolonnā ir pozitīvs 1, kas nozīmē, ka tas būs vadošais elements un izslēgta no bāzes tiks 2.kolonna. Varam sarēķināt otrās iterācijas tabulu.

$$T^{(2)} = \begin{array}{c|c|ccccc} & 37 & 22 & 7 & 0 & 0 & \mathbf{-5} \\ \hline 4 & 8 & 4 & 1 & 0 & 1 & \mathbf{-3} \\ 3 & 19 & 1 & 2 & 1 & 0 & \mathbf{-2} \end{array}$$

Tā kā novērtējumu rindā $-5 < 0$, tad plāns nav optimāls. Nepieciešams 5.kolonnu iekļaut bāzē, bet diemžēl $a^5 = \begin{pmatrix} -3 \\ -2 \end{pmatrix} \leq 0$; 2'.rekomendācija saka, ka mērķa funkcija ir neierobežota un skaitļošanas process ir jāpabeidz. Tātad dotajam uzdevumam nav atrisinājuma. ■

2.6 SIMPLEKSA METODES GALĪGUMS

Ja visi atbalsta plāni nav deģenerēti, tad simpleksa metode ir galīga. Tā kā uzdevumam ir tikai galīgs skaits atbalsta plānu un ja katrā iterācijā notiek pāreja tikai uz tādu atbalsta plānu, kuram mērķa funkcijas vērtība ir lielāka nekā iepriekšējam atbalsta plānam, tad optimālais plāns, ja tāds vispār eksistē, tiek atrasts pēc galīga skaita iterācijām. Tātad, ja ir stingra monotonitāte, tad simpleksa metodes algoritms ir galīgs.

Apskatīsim situācijas, kad dotam lineārās programmēšanas uzdevumam kanoniskajā formā

$$\begin{gathered} \max < c, x > \\ Ax = b, \quad x \geq 0. \end{gathered} \tag{2.6.1}$$

var būt deģenerētri atbalsta plāni. Šajā gadījumā simpleksa metodē var rasties problēmas, kuras ar noteiktiem paņēmieniem var novērst.

Ja matricas A rangs ir mazāks par m, tad visi uzdevuma (2.6.1) atbalsta plāni ir deģenerēti. Kā atrisināt šo situāciju, apskatīsim nākamajā paragrāfā. Bet pat tad, ja matricas rangs ir m, uzdevumam (2.6.1) var būt deģenerēti atbalsta plāni.

Piemērs 2.6.1. Lineārās programmēšanas uzdevumam kanoniskajā formā

$$\max(x_1 + 3x_2 + 2x_3 + 4x_4 + x_5)$$

$$\begin{array}{cccccc} x_1 & +\frac{3}{2}x_2 & & +x_4 & & =\frac{3}{4}, \\ & \frac{3}{8}x_2 & +x_3 & +x_4 & & =\frac{11}{16}, \\ & \frac{9}{8}x_2 & & & +x_5 & =\frac{9}{16}, \\ x_1, & x_2, & x_3, & x_4, & x_5 & \geq 0 \end{array}$$

vektors $x = (\frac{3}{4}, 0, \frac{11}{16}, 0, \frac{9}{16})$ ir nedeģenerēts atbalsta plāns, bet $x = (0, \frac{1}{2}, \frac{1}{2}, 0, 0)$ ir deģenerēts atbalsta plāns. ■

Turpmākajā pieņemsim, ka matricas A rangs ir vienāds ar m. Bet atbalsta plānam ir tikai $l < m$ pozitīvas koordinātas $x^0_{i_r}$, $r = 1, 2, ..., l$, $x^0_i = 0$, ja $i \neq i_r$, $r = 1, 2, ..., l$, un $\{a^{i_1}, a^{i_2}, ..., a^{i_l}, a^{i_l+1}, ..., a^{i_m}\}$ ir kaut kāda atbalsta plāna x^0 bāze. Šajā gadījumā ir iespējama tāda situācija, ka pēc sākotnējās simpleksa tabulas $\overline{A}(\beta^{(q)})$ sastādīšanas var izpildīties visi simpleksa metodes pirmās iterācijas soļi, bet pēc tam, fiksējot kolonnu, kas izslēdzama no bāzes, un rindu, piemēram s, kurai atbilsošā kolonna tiktu iekļauta bāzē, atbilstošais elements $b_s(\beta^q)$ var izrādīties 0. Tas nozīmē, ka mērķa funkcijas vērtība nepalielināsies. Līdz ar to simpleksa metodes algoritmam nav stingras monotonitātes. Ja vairākās pēc kārtas sekojošās iterācijās atbilsošais kolonnas $b(\beta^q) = 0$, tad uzdevuma risināšanas gaitā mainās tikai viena un tā paša deģenerētā atbalsta plāna bāzes, bet mērķa funkcijas vērtība nemainās. Šajā gadījumā teorētiski iespējams atgriezties pie deģenerētā atbalsta plāna sākotnējās bāzes. Tādējādi veidojas *cikls* un uzdevums risināšanas algoritms var tikt atkārtots bezgalīgi daudzas reizes.

Piemērs 2.6.2. Pieņemsim, ka kaut kādā iterācijā esam nonākuši pie simpleksa tabulas šādā izskatā:

$$T^{(q)} = \begin{array}{c|c|cccccccc} & 3 & -\frac{3}{4} & 20 & -\frac{1}{2} & 6 & 0 & 0 & 0 \\ \hline 5 & 0 & \mathbf{\frac{1}{4}} & \text{-8} & \text{-1} & 9 & 1 & 0 & 0 \\ 6 & 0 & \frac{1}{2} & \text{-12} & -\frac{1}{2} & 3 & 0 & 1 & 0 \\ 7 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 1 \end{array}$$

$-\frac{3}{4}$ ir mazākais negatīvais novērtējums, tāpēc bāzē tiks iekļauta pirmā kolonna. Tā kā $b_5 = b_6 = 0$, tad $\lambda_5 = \lambda_6 = 0$, tāpēc nevar pateikt viennozīmīgi, kuru izslēgt no bāzes kolonnām. Dažos priekšraksos tiek ierosināts izslēgt to, kuras indekss ir mazākais. Tātad izslēgsim piekto kolonnu. Tā rīkosimies arī citkārt šajā piemērā. Rezultātā iegūsim iterāciju tabulu virkni, kas atgriežas atpakaļ pie sākotnējā atbalsta plāna. Vadošais elements tabulās iezīmēts treknāk.

$$T^{(q+1)} = \begin{array}{c|c|ccccccc} & 3 & 0 & -4 & -\frac{7}{2} & 33 & 3 & 0 & 0 \\ \hline 1 & 0 & 1 & -32 & -4 & 36 & 4 & 0 & 0 \\ 6 & 0 & 0 & \mathbf{4} & \frac{3}{2} & -15 & -2 & 1 & 0 \\ 7 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 1 \end{array}$$

$$T^{(q+2)} = \begin{array}{c|c|ccccccc} & 3 & 0 & 0 & -2 & 18 & 1 & 1 & 0 \\ \hline 1 & 0 & 1 & 0 & \mathbf{8} & -84 & -12 & 8 & 0 \\ 2 & 0 & 0 & 1 & \frac{3}{8} & -\frac{15}{4} & -\frac{1}{2} & \frac{1}{4} & 0 \\ 7 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 1 \end{array}$$

$$T^{(q+3)} = \begin{array}{c|c|ccccccc} & 3 & \frac{1}{4} & 0 & 0 & -3 & -2 & 3 & 0 \\ \hline 3 & 0 & \frac{1}{8} & 0 & 1 & -\frac{21}{2} & -\frac{3}{2} & 1 & 0 \\ 2 & 0 & -\frac{3}{64} & 1 & 0 & \mathbf{\frac{3}{16}} & \frac{1}{16} & -\frac{1}{8} & 0 \\ 7 & 1 & -\frac{1}{8} & 0 & 0 & \frac{21}{2} & \frac{3}{2} & -1 & 1 \end{array}$$

$$T^{(q+4)} = \begin{array}{c|c|ccccccc} & 3 & -\frac{1}{2} & 16 & 0 & 0 & -1 & 1 & 0 \\ \hline 3 & 0 & -\frac{5}{2} & 56 & 1 & 0 & \mathbf{2} & -6 & 0 \\ 4 & 0 & -\frac{1}{4} & \frac{16}{3} & 0 & 1 & \frac{1}{3} & -\frac{2}{3} & 0 \\ 7 & 1 & \frac{5}{2} & -56 & 0 & 0 & -2 & 6 & 1 \end{array}$$

$$T^{(q+5)} = \begin{array}{c|c|ccccccc} & 3 & -\frac{7}{4} & 44 & \frac{1}{2} & 0 & 0 & -2 & 0 \\ \hline 5 & 0 & -\frac{5}{4} & 28 & \frac{1}{2} & 0 & 1 & -3 & 0 \\ 4 & 0 & -\frac{1}{6} & -4 & -\frac{1}{6} & 1 & 0 & \mathbf{\frac{1}{3}} & 0 \\ 7 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 1 \end{array}$$

$$T^{(q+6)} = \begin{array}{c|c|ccccccc} & 3 & -\frac{3}{4} & 20 & -\frac{1}{2} & 6 & 0 & 0 & 0 \\ \hline 5 & 0 & \frac{1}{4} & -8 & -1 & 9 & 1 & 0 & 0 \\ 6 & 0 & \frac{1}{2} & -12 & -\frac{1}{2} & 3 & 0 & 1 & 0 \\ 7 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 1 \end{array}$$

Esam nonākuši pie tādas pašas tabulas kā sākumā (ieciklojušies). ■

Lai optimālo plānu atrastu pēc galīga skaita iterācijām, jānovērš cikla rašanās. To var panākt, lietojot speciālas metodes, kuras nodrošina simpleksa metodes algoritma galīgumu arī tad, ja uzdevumam ir deģenerēti atbalsta plāni. Praktiskās situācijās, kad lineārās programmēšanas uzdevums tiek risināts ar datora palīdzību, ieciklošanās gadījumi netiek novēroti. Kāpēc? Tas saistīts ar noapaļošanā uzkrātajām kļūdām, kuru rezultātā praktiski nonākt atpakaļ pie tā paša plāna nav iespējams. Tomēr teorētiski pastāv iespēja iecikloties. 1976.gadā ir ticis izstrādāts algoritms, kas novērš ieciklošanās iespēju. Tā autors ir beļģu matemātiķis *R.G.Bland.*

Teorēma 2.6.1 (Blenda algoritms). Ja kolonnu, kas tiks iekļauta bāzē simpleksa algoritmā, izvēlas pēc likuma

$$j = \min\{j \mid a_{0,j}(\beta^{(q)}) < 0\}$$

(tiek izvēlēta kolonna ar negatīvu novērtējumu, kuras indekss ir vismazākais), bet rindu izvēlas pēc likuma

$$s = \min\{\lambda_{sj} \mid \lambda_{sj} = \frac{b_s}{a_{sj}(\beta^{(q)})},\ \text{kur } a_{sj}(\beta^{(q)}) > 0\}$$

(nenotektības gadījumā no bāzes tiks izslēgta kolonna ar mazāko indeksu), tad šis simpleksa algoritms darbu beidz pēc galīga skaita iterācijām.

Pierādījums. Parādīsim, ka pieņēmums par cikla eksistenci noved pie pretrunas. Cikls nozīmē, ka pēc galīga skaita bāzes vektoru maiņām mēs atgriežamies pie tā paša pieļaujamā atrisinājuma. Pie tam mērķa funkcijas z vērtībai ir jābūt visu laiku vienai un tai pašai visa cikla laikā. Kā arī vērtībai b_s, kas atbilst katrai maiņai, ir jābūt vienādai ar 0, jo pretējā gadījumā $\lambda_{sj} > 0$, no šejienes sekotu, ka z dilst. No šejienes seko, ka ka nultā kolonna b ir nemainīga visa cikla laikā.

Atmetot rindiņas un kolonnas, kuras nesatur cikla laikā vadošos elementus, iegūsim jaunu lineārās programmēšanas uzdevumu, kurš ieciklojas un kuram cikla laikā visi $a_{s0} = 0$, kā arī mērķa funkcijas vērtība visu laiku ir viena un tā pati.

Pieņemsim, ka q ir lielākais indekss tam mainīgajam, kuru iekļaujam bāzē cikla laikā. Apskatīsim divas tabulas: tabulu T_1, pēc kuras iegūšanas a^q tiek iekļauts bāzē, un tabulu T_2, pēc kuras iegūšanas vektors a^q tiek izslēgts no bāzes.

$$T_1 = \begin{array}{c|c|ccc} & b_0 & \dots & & a_{0q} < 0 \\ \hline & 0 & & & \\ & 0 & & & \\ & \dots & & & \\ & 0 & & & \end{array}$$

$$T_2 = \begin{array}{c|c|ccccc}
 & \hat{b}_0 & ... & \hat{a}_{0p} < 0 & ... & \hat{a}_{0q} \\
\hline
 & 0 & & & & \\
 & ... & & & & \\
q & 0 & ... & \hat{a}_{qp} > 0 & ... & +1 \\
 & ... & & & & \\
 & 0 & & & &
\end{array}$$

Apzīmēsim tabulas T_1 elementus ar a_{ij} un tabulas T_2 elementus ar $\hat{a}_{ij}$, atbilstoši N un $\hat{N}$ veido T_1 un T_2 bāzes. Pieņemsim, ka pēc tabulas T_2 iegūšanas p-tā kolonna tiek iekļauta bāzē.

Lemma 2.6.1. Pieņemsim, ka $\bar{c}$ ir novērtējumu rinda jebkurā tabulā X_1 ar vienības bāzi (nav obligāti jāatbilst pieļaujamam atrisinājumam, t.i., daži b_{i0} var būt arī negatīvi). Pieņemsim, ka y ir vienādību sistēmas $Ay = b$ atrisinājums, kas var nebūt pieļaujams atrisinājums (t.i., daži y_j var būt negatīvi). Pieņemsim, ka f ir mērķa funkcijas vērtība, kas atbilst tabulai X_1, bet g ir mērķa funkcijas vērtība, kas atbilst atrisinājumam y. Tad $\bar{c}y = g - f$.

Pierādījums. Pakāpenisku pārveidojumu ceļā iegūsim

$$\bar{c}y = (c - z)y = cy - zy = g - c_B B^{-1} Ay = g - c_B B^{-1} b = g - f,$$

jo $B^{-1}b$ ir tabulas X_1 nultā kolonna. ■

Teorēmas 2.6.1 pierādījuma turpinājums. Izmantojot Lemmu 2.6.1, konstruēsim divus atrisinājumus. Tabulas T_1 gadījumā izmantosim pieļaujamo atrisinājumu x_0 (aizstājot T_1 ar X_1). Tabulas T_2 atrisinājumu y definēsim šādi:

$$y_j = \begin{cases} 1, & j = p, \\ -\hat{a}_{ip}, & a^j \in \hat{N}, \\ 0, & \text{pretējā gadījumā.} \end{cases}$$

Ievērosim, kaut arī y nav ne bāzes, ne pieļaujams atrisinājums, tomēr tas ir sistēmas $Ay = b$ atrisinājums, tāpēc apmierina Lemmas 2.6.1 nosacījumus. Pie tam mērķa funkcijas vērtība ar atrisinājumu y ir vienāda ar $f + \hat{a}_{0p}$, tāpēc Lemmas 2.6.1 slēdziens dod vērtējumu $\bar{c}y = \hat{a}_{0p} < 0$. Nevienādība seko no tā, ka p-tā kolonna tiek iekļauta tabulā T_2, tāpēc novērtējumam $\hat{a}_{0p}$ ir jābūt negatīvam.

Saskaņā ar vadošo kolonnu tabulā T_1, iegūsim, ka

$$\bar{c}_j \geq 0, \text{ ja } j < q, \ \bar{c}_j < 0, \text{ ja } j = q,$$

bet saskaņā ar vadošo kolonnu tabulā T_2

$$y_j = \begin{cases} -\hat{a}_{ip} < 0, & j = q, \\ 0,\ 1 \text{ vai } -\hat{a}_{ip} \geq 0, & j < q. \end{cases}$$

Tādējādi

$$\bar{c}y = \sum_{j<q} \bar{c}_j y_j + \bar{c}_q y_q \geq \bar{c}_q y_q > 0.$$

Esam ieguvuši pretrunu. ■

Piemēra 2.6.2 turpinājums. Vai Piemērā 2.6.2 izvairīsimies no ieciklošanās, ja izmantosim Blenda algoritmu? Salīdzinājumā ar iepriekšējo risinājuma gaitu, būtiska atšķirība būs pāreja uz tabulu $T^{(q+5)}$. Iepriekš par vadošo elementu izvēlējāmies 2 (piektā kolonna un pirmā rinda).

$$T^{(q+4)} = \begin{array}{c|c|ccccccc} & 3 & -\frac{1}{2} & 16 & 0 & 0 & -1 & 1 & 0 \\ \hline 3 & 0 & -\frac{5}{2} & 56 & 1 & 0 & \mathbf{2} & -6 & 0 \\ 4 & 0 & -\frac{1}{4} & \frac{16}{3} & 0 & 1 & \frac{1}{3} & -\frac{2}{3} & 0 \\ 7 & 1 & \mathbf{\frac{5}{2}} & -56 & 0 & 0 & -2 & 6 & 1 \end{array}$$

Pēc Blenda algoritma vadošā kolonna ir pirmā, vadošais elements ir $\frac{5}{2}$, no bāzes izslēdzam septīto kolonnu.

$$T^{(q+5)} = \begin{array}{c|c|ccccccc} & \frac{1}{5} & 0 & \frac{24}{5} & 0 & 0 & -\frac{7}{5} & \frac{11}{5} & \frac{1}{5} \\ \hline 3 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 1 \\ 4 & \frac{1}{10} & 0 & -\frac{4}{15} & 0 & 1 & \mathbf{\frac{2}{15}} & -\frac{1}{15} & \frac{1}{10} \\ 1 & \frac{2}{5} & 1 & -\frac{112}{5} & 0 & 0 & -\frac{4}{5} & \frac{12}{5} & \frac{2}{5} \end{array}$$

Tagad vadošais elements ir $\frac{2}{15}$ (piektā kolonna un otrā rinda).

$$T^{(q+6)} = \begin{array}{c|c|ccccccc} & \frac{5}{4} & 0 & 2 & 0 & \frac{21}{2} & 0 & \frac{3}{2} & \frac{5}{4} \\ \hline 3 & 1 & & & & & & & \\ 5 & \frac{3}{4} & & & & & & & \\ 1 & 1 & & & & & & & \end{array}$$

Visi novērtējumi ir nenegatīvi, tātad ir atrasts optimālais plāns. Visu tabulu varam neaizpildīt, mūs interesē tikai optimālā plāna koordinātas un mērķa funkcijas vērtība. Tātad optimālais plāns ir $x^* = (1, 0, 1, 0, \frac{3}{4}, 0, 0)$ un mērķa funkcijas vērtība ir $z(x^*) = \frac{5}{4}$. ■

2.7 ATBALSTA PLĀNA ATRAŠANA

Iepriekš aprakstītā simpleksa metodes procedūra ir lietojama tikai tajā gadījumā, ja protam uzreiz noteikt atbalsta plānu x^0 un tā bāzi $\{a^{i_1}, a^{i_2}, ..., a^{i_m}\}$. Atrast sākotnējo atbalsta plānu x^0 nozīmē atrast vienādojumu sistēmas $Ax = b$, $x \geq 0$ nenegatīvu atrisinājumu. Viena no simpleksa metodes īpatnībām ir tā, ka šīs problēmas atrisinājumu var realizēt ar to pašu simpleksa metodes algoritmu.

Pieņemsim, ka vektora b visas koordinātas ir nenegatīvas: $b \geq 0$ (ja tā nav, tad atbilstošo sistēmas vienādojumu var pareizināt ar -1). Apskatīsim lineārās programmēšanas palīguzdevumu kanoniskajā formā ar $n+m$ mainīgajiem

$$
\begin{aligned}
&\max(-v_1 - v_2 - \ldots - v_m)\\
&a_{11}x_1 + a_{12}x_2 + \ldots + a_{1n}x_n + v_1 = b_1,\\
&a_{21}x_1 + a_{22}x_2 + \ldots + a_{2n}x_n + v_2 = b_2,\\
&\qquad\ldots\\
&a_{m1}x_1 + a_{m2}x_2 + \ldots + a_{mn}x_n + v_m = b_m,\\
&x_j \geq 0,\ j = 1, 2, ..., n;\ v_i \geq 0,\ i = 1, 2, ..., m.
\end{aligned}
\tag{2.7.1}
$$

Varam novērot dažas sakarības starp uzdevumu

$$
\begin{aligned}
&\max < c, x >\\
&Ax = b, \quad x \geq 0.
\end{aligned}
\tag{2.7.2}
$$

un (2.7.1). Uzdevums (2.7.1) ir pieļaujams: vektors $(0, 0, ..., b_1, b_2, ..., b_m)$ apmierina prasību sistēmu. Pie tam mērķa funkcija ir ierobežota no augšas ar 0. Tādā gadījumā uzdevumam (2.7.1) eksistē atrisinājums. Pieņemsim, ka d ir uzdevuma (2.7.1) mērķa funkcijas maksimālā vērtība. Iespējami divi gadījumi: $d = 0$ vai $d < 0$.

Teorēma 2.7.1. Ja palīguzdevumā (2.7.1) vērtība $d = 0$, tad uzdevums (2.7.2) ir pieļaujams. Ja palīguzdevumā (2.7.1) vērtība $d < 0$, tad uzdevums (2.7.2) nav pieļaujams.

Pierādījums. Pieņemsim, ka uzdevums (2.7.2) ir pieļaujams un $x^0 = (x_1^0, x_2^0, x_3^0, ..., x_n^0)$ ir šī uzdevuma plāns. Tādā gadījumā acīmredzami, ka $n+m$ dimensiju vektors $\tilde{x} = (0, 0, ..., 0, x_1^0, x_2^0, x_3^0, ..., x_n^0)$ apmierina visus uzdevuma (2.7.1) ierobežojumus, t.i., ir šī uzdevuma plāns; pie tam šī uzdevuma mērķa funkcijas vērtība ar plānu $\tilde{x}$ ir vienāda ar 0. No šejienes seko, ka plāns

$\tilde{x}$ ir optimāls palīguzdevumam (2.7.1) un $d = 0$. No otras puses, ja $d = 0$, tad jebkuram uzdevuma (2.7.1) atrisinājumam $(v_1, v_2, ..., v_m, x_1, x_2, ..., x_n)$ izpildās vienādības $v_1 = v_2 = ... = v_m = 0$, tāpēc vektors $(x_1, x_2, ..., x_n)$ ir uzdevuma (2.7.2) plāns. ■

No pierādītās teorēmas seko, ka reizē ar palīguzdevuma (2.7.1) atrisinājumu tiks atrasts uzdevuma (2.7.2) atrisinājums vai arī tiks konstatēts fakts, ka (2.7.2) ierobežojumu sistēma ir nesaderīga.

Uzdevumam (2.7.1) uzreiz var uzrādīt atbalsta plānu; par tādu der vektors $(b, 0) = (b_1, b_2, ..., b_m, 0, 0, ..., 0)$, jo tas apmierina visus šī uzdevuma ierobežojumus, bet telpas $\mathbf{R}^m$ vienības vektoru sistēma $v^1, v^2, ..., v^m$ veido uzdevuma bāzi.

Tādējādi palīguzdevums (2.7.1) ļauj atrast ne tikai atbalsta plānu uzdevumam (2.7.2), bet arī ļauj noskaidrot lineāri atkarīgos nosacījumus šajā uzdevumā. Pēc tam var lietot simpleksa metodi gadījumam, kad matricas A rangs sakrīt ar tās rindu skaitu.

Simpleksa tabula, kas tiek iegūta sākuma atbalsta plāna atrašanas pēdējā iterācijā, ļauj uzreiz noteikt sākuma simpleksa tabulu pamatuzdevumam (2.7.2) — vajag tikai atmest liekās kolonnas v^1, v^2,..., v^m un atbilstošos palīgmainīgos v_1, v_2, ..., v_m un nomainīt novērtējumu rindu, kuras elementus var pārrēķināt pēc formulām (iterāciju skaitīšanu sākam no jauna, tāpēc $q = 1$)

$$a_{0,0}(\beta^{(1)}) = b_0 = c_\beta x^\beta,$$
$$a_{0,i}(\beta^{(1)}) = c_\beta \, a^i(\beta^{(1)}) - c_i, \; i = 1, 2, ..., n,$$

kur c_β ir to mērķa funkciju koeficientu vektors, kuriem atbilstošās kolonnas ietilpst palīguzdevumā atrastā atbalsta plāna bāzē, x^{beta} ir palīguzdevumā atrastā atbalsta plāna koordinātas (tās var interpretēt arī kā jauno ierobežojumu vektoru), bet $a^i(\beta^{(1)})$ ir kolonnu vektori no palīguzdevuma pēdējās tabulās.

Piemērs 2.7.1. Apskatīsim uzdevumu kanoniskajā formā, kuram prasību sistēmas pēdējais vienādojums ir pārējo vienādojumu summa (tātad šis vienādojums ir lieks).

$$\max(x_1 + 2x_2 + 4x_3)$$
$$\begin{array}{rrrl} x_1 & +x_2 & +x_3 & = 10, \\ x_1 & +2x_2 & +3x_3 & = 15, \\ 2x_1 & +3x_2 & +4x_3 & = 25, \\ x_1, & x_2, & x_3 & \geq 0 \end{array} \tag{2.7.3}$$

Uzdevuma atrisināšanu sāksim ar sākuma atbalsta plāna atrašanu, šajā nolūkā sastādot palīguzdevumu:

$$\max(-v_1 - v_2 - v_3)$$

$$\begin{array}{rrrrl} x_1 & +x_2 & +x_3 & +v_1 & = 10, \\ x_1 & +2x_2 & +3x_3 & +v_2 & = 15, \\ 2x_1 & +3x_2 & +4x_3 & +v_3 & = 25, \end{array} \tag{2.7.4}$$

$$x_j \geq 0,\ j = 1, 2, 3;\ v_i \geq 0,\ i = 1, 2, 3.$$

Par šī uzdevuma (2.7.4) pirmo atbalsta plānu varam izvēlēties vektoru

$$(0, 0, 0, 10, 15, 25)$$

ar bāzi v^1, v^2, v^3. Pirmās simpleksa tabulas sastādīšanu veiksim bez inversās matricas palīdzības. 2.5.1.zīmējumā tika parādīts, kāda izskatās simpleksa tabula. Ievērojot šo gala secinājumu, mums atliek izrēķināt tikai novērtējumu rindu pēc formulām

$$a_{0,0}(\beta^{(1)}) = b_0 = c_\beta b,$$

$$a_{0,i}(\beta^{(1)}) = c_\beta\, a^i - c_i,\ i = 1, 2, ..., n,$$

kur c_β ir to mērķa funkciju koeficientu vektors, kuriem atbilstošās kolonnas ietilpst palīguzdevumā atrastā atbalsta plāna bāzē, tātad

$$c_\beta = (-1, -1, -1).$$

Proti,

$$\begin{aligned} & a_{0,0} = c_\beta b = (-1, -1, -1)\,(10, 15, 25) = -10 - 15 - 25 = -50, \\ & a_{0,1} = (-1, -1, -1)\,(1, 1, 2) = -1 - 1 - 2 = -4, \\ & a_{0,2} = (-1, -1, -1)\,(1, 2, 3) = -1 - 2 - 3 = -6, \\ & a_{0,1} = (-1, -1, -1)\,(1, 3, 4) = -1 - 3 - 4 = -8, \\ & a_{0,v_1} = (-1, -1, -1)\,(1, 0, 0) - (-1) = -1 + 1 = 0, \\ & a_{0,v_2} = (-1, -1, -1)\,(0, 1, 0) - (-1) = -1 + 1 = 0, \\ & a_{0,v_1} = (-1, -1, -1)\,(0, 0, 1) - (-1) = -1 + 1 = 0. \end{aligned}$$

Pirmā simpleksa tabula būs šāda (apzīmējam palīguzdevuma pirmo tabulu ar $T_p^{(1)}$)

$$T_p^{(1)} = \begin{array}{c|c|cccccc} & b & a^1 & a^2 & a^3 & v^1 & v^2 & v^3 \\ \hline & \text{-50} & \text{-4} & \text{-6} & \text{-8} & 0 & 0 & 0 \\ \hline v^1 & 10 & \mathbf{1} & 1 & 1 & 1 & 0 & 0 \\ v^2 & 15 & 1 & 2 & 3 & 0 & 1 & 0 \\ v^3 & 25 & 2 & 3 & 4 & 0 & 0 & 1 \end{array}$$

Vadošais elements (pēc Blenda algoritma) ir 1 (pirmā kolonna, pirmā rinda). Veicam aprēķinus pēc simpleksa metodes formulām: vadošo rindu dalām ar vadošo elementu (šajā konkrētajā gadījumā nekas nemainās); vadošās kolonnas elementus aizstājam ar 0, izņemot vadošo elementu, tas vienmēr nākamajā tabulā būs 1; tos elementus, kas neatrodas vadošajā rindā vai vadošajā kolonnā, pārrēķinām pēc "taisnstūra formulas": elements mīnuss projekciju uz vadsošo rindu un vadošo kolonnu elementu reizinājuma dalījums ar vadošo elementu. Piemēram, skaitļa 4 vietā tabulā $T_p^{(2)}$ būs skaitlis $4 - \frac{1\cdot 2}{1} = 2$. Otrā tabula ir šāda

$$T_p^{(2)} = \begin{array}{c|c|cccccc} & b & a^1 & a^2 & a^3 & v^1 & v^2 & v^3 \\ \hline & \text{-10} & 0 & \text{-2} & \text{-4} & 4 & 0 & 0 \\ \hline a^1 & 10 & 1 & 1 & 1 & 1 & 0 & 0 \\ v^2 & 5 & 0 & \mathbf{1} & 2 & \text{-1} & 1 & 0 \\ v^3 & 5 & 0 & 1 & 2 & \text{-2} & 0 & 1 \end{array}$$

Pēc Blenda algoritma vadošā kolonna ir otrā un no bāzes ir jāizslēdz v^2, jo tam ir mazāks indekss (kaut arī abiem vektoriem v^2 un v^3 dalījumi $\frac{5}{1} = \frac{5}{1}$ vienādi). Iegūsim trešo tabulu

$$T_p^{(3)} = \begin{array}{c|c|cccccc} & b & a^1 & a^2 & a^3 & v^1 & v^2 & v^3 \\ \hline & 0 & 0 & 0 & 0 & 2 & 2 & 0 \\ \hline a^1 & 5 & 1 & 0 & \text{-1} & 2 & \text{-1} & 0 \\ a^2 & 5 & 0 & 1 & 2 & \text{-1} & 1 & 0 \\ v^3 & 0 & 0 & 0 & 0 & \text{-1} & \text{-1} & 1 \end{array}$$

Tā kā nav negatīvu novērtējumu, tad ir atrasts optimālais plāns palīguzdevumam (2.7.4), tas ir $(5, 5, 0, 0, 0, 0)$ ar bāzi a^1, a^2 un v^3. Tā kā $d = 0$, tad pēc Teorēmas 2.7.1 dotais uzdevums (2.7.3) ir pieļaujams. Tā kā vektoram v^3 bāzē atbilst koordināta 0 un vienlaicīgi arī zem a^1, a^2 un a^3 atrodas 0, tad v^3 nevar apmainīt ar a^3 (lai īsto veco kolonnu iekļautu bāzē), tas norāda uz to, ka sistēmā trešais vienādojums ir lieks, to var atmest (matricas rangs kļūst vienāds ar rindu skaitu). Tabulā T_p^3 drīkst izmest rindu un kolonnu, kas atbilst vektoram v^3. Atbalsta plāna bāzi veido a^1 un a^2, atrastais atbalsta plāns ir (5,5,0).

Tālāk atgriežamies pie pamatuzdevuma rēķināšanas. Sastādot pirmo tabulu T^1, atmetam arī palīgmainīgo pārējās kolonnas un pārrēķinām novērtējumus atbilstoši uzdevuma (2.7.3) mērķa funkcijai $\max(x_1 + 2x_2 + 4x_3)$. Tā kā atrastā atbalsta plāna bāzē ietilpst pirmā un otrā kolonna, tad $c_\beta = (1, 2)$ (pirmais un otrais mērķa funkcijas koeficients). Bāzes kolonnu novērtējumiem ir jābūt vienādiem ar 0, a^3 novērtējums ir

$$a_{0,3} = (1, 2)\,(-1, 2) = -1 + 4 - 4 = -1$$

un atbalsta plāna mērķa funkcijas vērtība ir

$$a_{0,0} = b_0 = (1, 2)\,(5, 5) = 5 + 10 = 15.$$

Vispārīgā gadījumā var iznākt arī tā, ka palīguzdevuma pēdējā tabulā a^2 ir pirms a^1, tad ieteicams rindas samainīt vietām (uzdevuma atrisinājums taču nemainās, ja samaina vienādojumu kārtību!), jo vektora c^β koordinātas parasti tiek sakārtotas indeksu pieaugšanas secībā. Iegūsim šādu simpleksa tabulu

$$T^{(1)} = \begin{array}{c|c|ccc} & b & a^1 & a^2 & a^3 \\ \hline & 15 & 0 & 0 & -1 \\ \hline a^1 & 5 & 1 & 0 & -1 \\ a^2 & 5 & 0 & 1 & \mathbf{2} \end{array}$$

Tabulā $T^{(1)}$ ir viens negatīvs novērtējums, tam atbilst viens pozitīvs koeficients, kas nozīmē, ka no bāzes tiks izslēgts a^2 un bāzē iekļauts a^3. Veicam aprēķinus pēc simpleksa metodes formulām.

$$T^{(2)} = \begin{array}{c|c|ccc} & b & a^1 & a^2 & a^3 \\ \hline & 17\frac{1}{2} & 0 & \frac{1}{2} & 0 \\ \hline a^1 & 7\frac{1}{2} & & & \\ a^2 & \frac{5}{2} & & & \end{array}$$

Tā kā novērtējumu rindā visi skaitļi ir ≥ 0, tad šajā otrajā iterācijā ir atrasts optimālais plāns. Tāpēc mūs vairs neinteresē pārējie matricas koeficienti (tabulu varam atstāt tukšu), izņemot kolonnu b. Atbildē teiksim, ka optimālais plāns ir $x^* = (7\frac{1}{2}, 0, 2\frac{1}{2})$, mērķa funkcijas maksimālā vērtība ir $17\frac{1}{2}$. Ja uzdevums risināts pareizi, tad ar simpleksa metodi sarēķinātajai mērķa funkcijas vērtībai ir jāsakrīt ar to skaitli, ko iegūsim optimālā plāna koordinātas ievietojot mērķa funkcijā: $z(x^*) = 1 \cdot 7\frac{1}{2} + 2 \cdot \frac{5}{2} = 17\frac{1}{2}$. ■

Tomēr risinot uzdevumus, varam saskarties arī ar citiem gadījumiem nekā aplūkotais Piemērs 2.7.1.

Piemērs 2.7.2. Apskatīsim pavisam vienkāršu uzdevumu, kura atrisināšanā simpleksa metode nemaz nebūtu jālieto, bet no teorijas viedokļa tāds vienkāršāks piemērs ļauj labāk saprast, kāpēc notiek tā, kā tas būs redzams šajā piemērā.

Jāatrisina uzdevums kanoniskā formā

$$\max(2x_1 + x_2)$$

$$\begin{array}{rrl} 3x_1 & +2x_2 & = 1, \\ -5x_1 & +x_2 & = 3, \\ x_1, & x_2, & \geq 0. \end{array}$$

Atbilstošais palīguzdevums ir

$$\min(-v_1 - v_2)$$

$$\begin{array}{rrrl} 3x_1 & +2x_2 & +v_1 & = 1, \\ -5x_1 & +x_2 & +v_2 & = 3, \end{array}$$

$$x_1,\ x_2,\ v_1,\ v_2 \geq 0.$$

Palīguzdevuma pirmā simpleksa tabula ir šāda

$$T_p^{(1)} = \begin{array}{c|c|ccccc} & b & a^1 & a^2 & v^1 & v^2 \\ \hline & -4 & 2 & -3 & 0 & 0 \\ \hline v^1 & 1 & 3 & \mathbf{2} & 1 & 0 \\ v^2 & 3 & \text{-}5 & 1 & 0 & 1 \end{array}$$

Jāsamaina v^1 ar a^2, iegūsim otrās iterācijas tabulu.

$$T_p^{(2)} = \begin{array}{c|c|ccccc} & b & a^1 & a^2 & v^1 & v^2 \\ \hline & -\frac{5}{2} & \frac{13}{2} & 0 & \frac{3}{2} & 0 \\ \hline a^2 & \frac{1}{2} & \frac{3}{2} & 1 & \frac{1}{2} & 0 \\ v^2 & \frac{5}{2} & -\frac{13}{2} & 0 & -\frac{1}{2} & 1 \end{array}$$

Šajā tabulā visi novērtējumi ir nenegatīvi — tātad atrasts palīguzdevuma optimālais plāns. Tā kā $d = -\frac{5}{2}$, tad pēc Teorēmas 2.7.1 sākotnējam uzdevumam neeksistē atrisinājums (uzdevums nav pieļaujams). Vērīgāks lasītājs teiks, ka tā nu gan nevar būt, jo sistēmai

$$\begin{cases} 3x_1 & +2x_2 & = 1, \\ -5x_1 & +x_2 & = 3 \end{cases}$$

noteikti eksistē atrisinājums. Taisnība, šīs sistēmas atrisinājums ir vektors $(-\frac{5}{13}, \frac{14}{13})$, taču tas neder par lineārās programmēšanas uzdevuma atrisinājumu, jo tā visas koordinātas nav nenegatīvas! ∎

Piemērs 2.7.3. Apskatīsim vēl vienu vienkāršu piemēru

$$\max(4x_1 + 3x_2)$$

$$\begin{array}{ccc} x_1 & +3x_2 & = 5, \\ x_1 & +2x_2 & = 5, \\ x_1, & x_2, & \geq 0. \end{array}$$

Atšķirībā no Piemēra 2.7.2 te ir skaidri redzams sistēmas vienīgais atrisinājums $(5, 0)$, kas acīmredzami der par optimālo plānu. Vai šo pašu atrisinājumu iegūsim ar simpleksa metodi?

Sastādām palīguzdevumu

$$\max(4x_1 + 3x_2)$$

$$\begin{array}{cccc} x_1 & +3x_2 & +v_1 & = 5, \\ x_1 & +2x_2 & +v_2 & = 5, \end{array}$$

$$x_1,\, x_2,\, v_1,\, v_2 \geq 0.$$

Palīguzdevuma pirmā simpleksa tabula ir

$$T_p^{(1)} = \begin{array}{c|c|cccc} & b & a^1 & a^2 & v^1 & v^2 \\ \hline & -10 & \text{-2} & \text{-5} & 0 & 0 \\ \hline v^1 & 5 & \mathbf{1} & 3 & 1 & 0 \\ v^2 & 5 & 1 & 2 & 0 & 1 \end{array}$$

Ievērojot Blenda algoritmu, jāapmaina ir v^1 ar a^1, rezultātā iegūsim tabulu

$$T_p^{(2)} = \begin{array}{c|c|cccc} & b & a^1 & a^2 & v^1 & v^2 \\ \hline & 0 & 0 & 1 & 2 & 0 \\ \hline a^1 & 5 & 1 & 3 & 1 & 0 \\ v^2 & 0 & 0 & \mathbf{-1} & \text{-1} & 1 \end{array}$$

Iegūtajā tabulā nav neviena negatīva novērtējuma! Tas liecina, ka atrasts palīguzdevuma optimālais plāns. Pie tam $d = 0$, kas savukārt liecina, ka sākotnējais uzdevums ir pieļaujams. Diemžēl bāzē ir palikusi palīgmainīgā kolonna. Šādā situācijā nepieciešams šo kolonnu izslēgt no bāzes, samainot to ar kolonnu, kas bija dotajā sākotnējā uzdevumā (šajā gadījumā a^2).

$$T_p^{(2')} = \begin{array}{c|c|cccc} & b & a^1 & a^2 & v^1 & v^2 \\ \hline & 0 & 0 & 0 & 1 & 1 \\ \hline a^1 & 5 & 1 & 0 & -2 & 3 \\ a^2 & 0 & 0 & 1 & 1 & -1 \end{array}$$

Varam pāriet uz dotā uzdevuma atrisināšanu. Pārrēķinot novērtējumus ar $c^\beta = (4, 3)$, iegūsim tabulu

$$T^{(1)} = \begin{array}{c|c|cc} & b & a^1 & a^2 \\ \hline & 20 & 0 & 0 \\ \hline a^1 & 5 & 1 & 0 \\ a^2 & 0 & 0 & 1 \end{array}$$

Tabula rāda, ka atrasts dotā uzdevuma optimālais plāns (5, 0) ar mērķa funkcija vērtību 20. ■

Līdz šim mēs esam apskatījuši lineārās programmēšanas uzdevumus tikai kanoniskajā formā. Ja uzdevums ir dots standartformā (jeb normālformā), tad, pārveidojot to par kanoniskās formas uzdevumu, katrai nevienādībai klāt tiek pieskaitīts palīgmainīgais gluži kā atbalsta plāna meklēšanas palīguzdevumā. Tikai šajā gadījumā tas nav nekāds palīguzdevums; tādējādi standartformas uzdevumu atrisināšana ir pat vienkāršāka par kanoniskās formas uzdevumiem, ja vien visas nevienādības ir $\leq$ un vektora ierobežojumu vektora b koordinātas ir nenegatīvas.

Piemērs 2.7.4. Apskatīsim Piemēra 1.6.1 standartformas uzdevumu bez ceturtās nevienādības

$$\begin{aligned} &\max(x_1 + x_2) \\ &4x_1 + 5x_2 \leq 20, \\ &-x_1 + x_2 \leq 3, \\ &10x_1 + 5x_2 \leq 35, \\ &x_1 \geq 0,\ x_2 \geq 0. \end{aligned}$$

Šo uzdevumu esam atrisinājuši ar ģeometriskiem paņēmieniem, tā optimālais plāns ir $x^* = (2{,}5,\ 2)$ un mērķa funkcijas vērtība ir 4,5 (ceturtā nevienādība šo atrisinājumu neietekmē).

Ja šo uzdevumu risinām ar simpleksa metodes palīdzību, tad vispirms veicam pāreju uz kanonisko formu

$$
\begin{aligned}
&\max(x_1 + x_2)\\
&4x_1 + 5x_2 + v_1 = 20,\\
&-x_1 + x_2 + v_2 = 3,\\
&10x_1 + 5x_2 + v_3 = 35,\\
&x_1,\, x_2,\, v_1,\, v_2,\, v_3 \geq 0.
\end{aligned}
$$

Šajā uzdevumā nav nepieciešams risināt atbalsta plāna atrašanas palīguzdevumu, jo atbalsta plāna bāzi veido pēdējās trīs kolonnas. Tā kā šajā gadījumā $c^\beta = (0,0,0)$, tad pirmā simpleksa tabula ir šāda

$$
T^{(1)} = \quad \begin{array}{c|c|ccccc}
 & b & a^1 & a^2 & v^1 & v^2 & v^3 \\
\hline
 & 0 & \text{-1} & \text{-1} & 0 & 0 & 0 \\
\hline
v^1 & 20 & 4 & 5 & 1 & 0 & 0 \\
v^2 & 3 & \text{-1} & 1 & 0 & 1 & 0 \\
v^3 & 35 & \mathbf{10} & 5 & 0 & 0 & 1
\end{array}
$$

Pēc Blenda algoritma bāzē ir jāiekļauj pirmā kolonna a^1. Tā kā $\frac{20}{4} = 5 > \frac{35}{10} = 3{,}5$, tad no bāzes tiks izslēgta kolonna v^3.

$$
T^{(2)} = \quad \begin{array}{c|c|ccccc}
 & b & a^1 & a^2 & v^1 & v^2 & v^3 \\
\hline
 & 3{,}5 & 0 & -0{,}5 & 0 & 0 & 0{,}1 \\
\hline
v^1 & 6 & 0 & \mathbf{3} & 1 & 0 & -0{,}4 \\
v^2 & 6{,}5 & 0 & 1{,}5 & 0 & 1 & 0{,}1 \\
a^1 & 3{,}5 & 1 & 0{,}5 & 0 & 0 & 0{,}1
\end{array}
$$

Jāapmaina v^1 ar a^2; iegūsim tabulu:

$$
T^{(3)} = \quad \begin{array}{c|c|ccccc}
 & b & a^1 & a^2 & v^1 & v^2 & v^3 \\
\hline
 & 4{,}5 & 0 & 0 & \frac{1}{6} & 0 & \frac{1}{30} \\
\hline
a^2 & 2 & 0 & 1 & \frac{1}{3} & 0 & -\frac{2}{15} \\
v^2 & 3{,}5 & 0 & 0 & -\frac{1}{2} & 1 & \frac{3}{10} \\
a^1 & 2{,}5 & 1 & 0 & -\frac{1}{6} & 0 & \frac{1}{6}
\end{array}
$$

Šajā tabulā novērtējumi ir nenegatīvi, kas nozīmē, ka atrasts optimālais plāns. No tabulas varam nolasīt optimālā plāna koordinātas $x^* = (2{,}5\,,\,2)$ un mērķa funkcijas vērtību $z(x^*) = 4{,}5$, kas sakrīt ar Piemēra 1.6.1 atrisinājumu. Vienīgi neizpratni rada tabulas rinda ar v^2. Taču atcerieties, kā tika iegūts

Piemēra 1.6.1 atrisinājums (skatīt 1.6.2.zīmējumu). Optimālo plānu sniedz divu taišņu krustpunkts. Konkrētajā gadījumā pirmās un trešā taisnes krustpunkts ir tas plānu kopas stūra punkts, kurā tiek sasniegts mērķa funkcijas maksimums. Otrā taisne neietekmē šo atrisinājumu, tas arī atspoguļojas tabulā $T^{(3)}$. ■

Noslēgumā atzīmēsim vispārīgu rezultātu.

Teorēma 2.7.2. Ja lineārās programmēšanas uzdevumam ir optimālais plāns, tad to var vienmēr atrast ar simpleksa metodi.

Dažādās mācību grāmatās tiek dažādi aprakstīta simpleksa metode, arī atbalsta plānu meklēšanai tiek piedāvāti dažādi varianti. Noteiktās situācijās droši vien katrai modifikācijai ir savas priekšrocības. Bet, ja Jūs būsiet apguvuši vienu no piedāvātajiem variantiem pietiekoši labi, tad citu autoru variācijas nesagādās lielas grūtības, jo visu dažādo aprakstu pamatā ir viena un tā pati pamatmetode.

NODAĻA 3

DUALITĀTE

3.1 DUĀLAIS UZDEVUMS

Katram lineārās programmēšanas uzdevumam pēc noteiktām likumsakarībām var uzrakstīt atbilstošu uzdevumu, kuru sauc par duālo uzdevumu. Duālā uzdevuma nozīme vislabāk izpaužas lineārās programmēšanas uzdevumam, kas dots standartformā (jeb normālformā)

$$\begin{aligned}
&\max(c_1x_1 + c_2x_2 + \ldots + c_nx_n)\\
&a_{11}x_1 + a_{12}x_2 + \ldots + a_{1n}x_n \le b_1,\\
&a_{21}x_1 + a_{22}x_2 + \ldots + a_{2n}x_n \le b_2,\\
&\qquad\qquad \ldots\\
&a_{m1}x_1 + a_{m2}x_2 + \ldots + a_{mn}x_n \le b_m,\\
&x_1 \ge 0,\ x_2 \ge 0, \ldots,\ x_n \ge 0,
\end{aligned} \tag{3.1.1}$$

tāpēc mēs pamatā apskatīsim tieši šīs formas uzdevumus. Par (3.1.1) *duālo uzdevumu* sauc standartformas uzdevumu šādā izskatā

$$\begin{aligned}
&\min(b_1y_1 + b_2y_2 + \ldots + b_my_m)\\
&a_{11}y_1 + a_{21}y_2 + \ldots + a_{m1}y_m \ge c_1,\\
&a_{12}y_1 + a_{22}y_2 + \ldots + a_{m2}y_m \ge c_2,\\
&\qquad\qquad \ldots\\
&a_{1n}y_1 + a_{2n}y_2 + \ldots + a_{nm}y_m \ge c_n,\\
&y_1 \ge 0,\ y_2 \ge 0, \ldots,\ y_m \ge 0.
\end{aligned} \tag{3.1.2}$$

Duālā uzdevuma (3.1.2) konstruēšanas likumi pēc dotā uzdevuma (3.1.1) (kuru dualitātes teorijā pieņemts saukt par *tiešo* uzdevumu) ir šādi:

1) duālā uzdevumā (3.1.2) mainīgo y_i ir tik daudz, cik rindu ir tiešā uzdevuma (3.1.1) matricā A;

2) duālā uzdevumā (3.1.2) prasību matrica ir tiešā uzdevuma (3.1.1) matricas A transponētā matrica A^T;

3) duālā uzdevumā (3.1.2) ierobežojumu vektors ir tiešā uzdevuma (3.1.1) mērķa funkcijas koeficientu vektors c;

4) duālajā uzdevumā (3.1.2) visas nevienādības ir uz otru pusi nekā tiešajā uzdevumā (3.1.1);

5) duālajā uzdevumā (3.1.2) mērķa funkcija ir lineārā forma ar tiešā uzdevuma (3.1.1) ierobežojuma vektora b koeficientiem, pie tam tiek meklēts mērķa funkcijas minimums;

6) duālā uzdevuma (3.1.2) mainīgajiem ir jābūt nenegatīviem.

Tātad matricu pierakstā

tiešais uzdevums	duālais uzdevums
$\max < c, x >$	$\max < b, y >$
$Ax \le b, \quad x \ge 0.$	$A^T y \ge c, \quad y \ge 0.$

Tā kā jebkuras formas uzdevumu var pierakstīt standartformā, tad var teikt arī tā, ka jebkuram lineārās programmēšanas uzdevumam eksistē tā duālais uzdevums. Piemēram, ja tiešais uzdevums ir šāds

$$\begin{gathered}\max(5x_1 + 12x_2 + 4x_3)\\ x_1 + 2x_2 + x_3 \le 10,\\ 2x_1 - x_2 + 3x_3 = 8,\\ x_1,\ x_2,\ x_3 \ge 0,\end{gathered}$$

tad to mēs varam noformēt standartformā šādi

$$\begin{gathered}\max(5x_1 + 12x_2 + 4x_3)\\ x_1 + 2x_2 + x_3 \le 10,\\ 2x_1 - x_2 + 3x_3 \le 8,\\ -2x_1 + x_2 - 3x_3 \le -8,\\ x_1,\ x_2,\ x_3 \ge 0.\end{gathered}$$

Šī uzdevuma duālais ir

$$
\begin{gathered}
\min(10y_1 + 8x_2 - 8y_3) \\
y_1 + 2y_2 - 2y_3 \geq 5, \\
2y_1 - y_2 + y_3 \geq 12, \\
y_1 + 3y_2 - 3y_3 \geq 4, \\
y_1,\ y_2,\ y_3 \geq 0.
\end{gathered}
$$

3.2 DUĀLITĀTES TEORĒMAS

Likumsakarības starp tiešo un duālo uzdevumu atspoguļo tā saucamās dualitātes teorēmas.

Apzīmēsim ar D tiešā uzdevuma (3.1.1) pieļaujamo plānu kopu, bet ar D^u duālā uzdevuma (3.1.2) pieļaujamo plānu kopu.

Lemma 3.2.1 (dualitātes nevienādība). Visiem $x \in D$ un visiem $y \in D^u$ izpildās nevienādība $< c, x > \leq < b, y >$.

Pierādījums. Izvēlamies patvaļīgus pieļaujamos plānus $x \in D$ un $y \in D^u$. Tiešā uzdevuma plānam x ir spēkā nevienādība $Ax \leq b$, $x \geq 0$. Savukārt duālā uzdevuma plānam y izpildās nevienādība $A^T y \geq c$, $y \geq 0$.

Pareizināsim skalāri abas nevienādības $Ax \leq b$ puses ar vektoru $y \geq 0$:

$$< Ax, y > \leq < b, y > .$$

Līdzīgi pareizināsim skalāri abas nevienādības $A^T y \geq c$ puses ar vektoru $x \geq 0$:

$$< A^T y, x > \geq < c, x > .$$

Tā kā

$$
\begin{aligned}
< Ax, y > = \quad & (a_{11}x_1 + a_{12}x_2 + ... + a_{1n}x_n)\, y_1 + \\
& +(a_{21}x_1 + a_{22}x_2 + ... + a_{2n}x_n)\, y_2 + ... \\
& ... + (a_{m1}x_1 + a_{m2}x_2 + ... + a_{mn}x_n)\, y_m = \\
& = (a_{11}y_1 + a_{21}y_2 + ... + a_{m1}y_m)\, x_1 + \\
& +(a_{12}y_1 + a_{22}y_2 + ... + a_{m2}y_m)\, x_2 + ... \\
& ... + (a_{1n}y_1 + a_{2n}y_2 + ... + a_{mn}y_m)\, x_n = \\
= < A^T y, x >, \quad &
\end{aligned}
$$

tad $< c, x > \leq < A^T y, x > = < Ax, y > \leq < b, y >$, kas bija jāpierāda. ■

Lemma 3.2.2 (dualitātes vienādība). Ja eksistē tādi $x^* \in D$ un $y^* \in D^u$, ka izpildās vienādība $< c, x^* > = < b, y^* >$, tad x^* ir tiešā uzdevuma optimālais plāns, bet y^* ir duālā uzdevuma optimālais plāns.

Pierādījums. Pieņemsim, ka $x \in D$ ir tiešā uzdevuma patvaļīgs pieļaujamais plāns. Tad pēc iepriekšējās Lemmas 3.2.1 ir spēkā

$$< c, x > \leq < b, y^* > = < c, x^* >,$$

t.i., x^* ir tiešā uzdevuma optimālais plāns (tiek sasniegta mērķa funkcijas maksimālā vērtība ar šo plānu x^*).

Līdzīgi, ja $y \in D^u$ ir duālā uzdevuma patvaļīgs pieļaujamais plāns, tad pēc Lemmas 3.2.1 ir spēkā

$$< b, y > \geq < c, x^* > = < b, y^* >,$$

t.i., y^* ir duālā uzdevuma optimālais plāns (tiek sasniegta mērķa funkcijas minimālā vērtība ar šo plānu y^*). ■

Lemma 3.2.2 dod lineārās programmēšanas uzdevuma plānu optimalitātes pietiekamo nosacījumu. Dualitātes pamatteoriju veido divas teorēmas: dualitātes teorēma un līdzsvara teorēma.

Teorēma 3.2.1 (dualitātes teorēma). Ja savstarpēji duālajie uzdevumi (3.1.1) un (3.1.2) ir pieļaujami, tad tiem abiem eksistē optimālie plāni, pie tam mērķa funkciju vērtības ir vienādas.

Pierādījums. Tātad pieņemam, ka savstarpēji duālie uzdevumi (3.1.1) un (3.1.2) ir pieļaujami. Apskatīsim lineāru nevienādību sistēmu ar nezināmajiem

$$(x, y) = (x_1, x_2, ..., x_n, y_1, y_2, ..., y_m) \in \mathbf{R}^{n+m}$$

šādā izskatā

$$\begin{array}{l} Ax \leq b, \quad A^T y \geq c, \\ < x, c > \geq < y, b >, \quad x \geq 0, \quad y \geq 0. \end{array} \tag{3.2.1}$$

Ja (x^*, y^*) ir sistēmas (3.2.1) atrisinājums, tad x^* un y^* ir atbilstoši tiešā un duālā uzdevuma optimālie plāni. Patiešām, ja (x^*, y^*) apmierina nevienādību sistēmu (3.2.1), tad x^* un y^* ir pieļaujami plāni katrs savam kanoniskajam uzdevumam. Pēc Lemmas 3.2.1 izpildās nevienādība $< c, x^* > \leq < b, y^* >$. Bet no sistēmas (3.2.1) seko, ka $< c, x^* > \geq < b, y^* >$. Kopā ar iepriekšējo nevienādību seko, ka $< c, x^* > = < b, y^* >$ — no Lemmas 3.2.2 seko, ka x^* un y^* ir atbilstoši tiešā un duālā uzdevuma optimālie plāni (ar vienādām mērķa funkcijas vērtībām).

Tādējādi atliek pierādīt, ka nevienādību sistēmai (3.2.1) eksistē atrisinājums (x^*, y^*).

Pierakstīsim sistēmu (3.2.1) matricu formā:

$$\begin{pmatrix} A & 0 \\ 0 & -A^T \\ -c & b \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} \leq \begin{pmatrix} b \\ -c \\ 0 \end{pmatrix}, \quad (x, y) \geq 0 \qquad (3.2.2)$$

vai īsāk

$$\tilde{A}z \leq \tilde{b}, \quad z \geq 0, \qquad (3.2.3)$$

kur apzīmēts

$$\tilde{A} = \begin{pmatrix} A & 0 \\ 0 & -A^T \\ -c & b \end{pmatrix}, \quad \tilde{b} = (b, -c, 0), \quad z = (x, y).$$

Pieņemsim pretējo — sistēmai (3.2.1) atrisinājums neeksistē, tas nozīmē, ka arīsistēmām (3.2.2) un (3.2.3) atrisinājumi neeksistē. Saskaņā ar nevienādību alternatīvām tādā gadījumā eksistē atrisinājums sistēmai

$$\tilde{A}^T \tilde{v} \geq 0, \quad < \tilde{v}, \tilde{b} >< 0, \quad \tilde{v} \geq 0. \qquad (3.2.4)$$

Apskatīsim šo sistēmu precīzāk:

$$\begin{gathered} \begin{pmatrix} A^T & 0 & -c \\ 0 & -A & b \end{pmatrix} \begin{pmatrix} v \\ w \\ u \end{pmatrix} \geq 0, \\ < v, b > - < w, c >< 0, \quad v \geq 0,\ w \geq 0,\ u \geq 0, \end{gathered} \qquad (3.2.5)$$

kur $\tilde{v} = (v, w, u)$. Šeit v ir $m-$-dimensiju vektors, w ir n-dimensiju vektors un u ir skaitlis.

Sistēmu (3.2.5) var pierakstīt šādā veidā:

$$\begin{aligned} &A^T v - uc \geq 0, \quad -Aw + ub \geq 0, \\ &< v, b > - < w, c >< 0, \quad v \geq 0,\ w \geq 0,\ u \geq 0, \end{aligned} \qquad (3.2.6)$$

Parādīsim, ka sistēmai (3.2.6) neeksistē atrisinājums.

Pieņemsim, ka x un y ir tiešā un duālā uzdevuma pieļaujamie plāni. Tādā gadījumā

$$\begin{aligned} &< v, b > \geq < v, Ax >==< A^T v, x > \geq\ u < c, x >, \\ &< w, c > \leq < w, A^T y >=< Aw, y > \leq\ u < b, y >, \end{aligned}$$

$$\text{t.i.,} \quad < v, b > \geq u < x, c >, \quad < w, c > \leq \; u < y, b > .$$

Atņemot no pirmās nevienādības otro, iegūsim

$$0 >< v, b > \; - \; < w, c > \geq \; u\,(< x, c > - < y, b >),$$

no šejienes seko, ka $u > 0$.

Apskatīsim vektorus $\overline{v} = \frac{v}{u}$ un $\overline{w} = \frac{w}{u}$. No nevienādībām (3.2.6), ievērojot $u > 0$, seko, ka

$$\begin{array}{l} A^T\overline{v} \geq c, \quad \overline{v} \geq 0, \\ A\overline{w} \leq b, \quad \overline{w} \geq 0, \\ < \overline{w}, c > < < \overline{,} b > . \end{array}$$

Pirmā rinda nozīmē, ka vektors $\overline{v}$ ir pieļaujams vektors duālajam uzdevumam, bet no otrās rindas seko, ka $\overline{w}$ ir pieļaujams vektors tiešajam uzdevumam Bet šādā gadījumā pēdējā nevienādība ir neiespējama, jo tā ir pretrunā ar Lemmu 3.2.2. Iegūtā pretruna noslēdz pierādījumu. ■

Teorēma 3.2.2 (līdzsvara teorēma). Pieņemsim, ka $x^* = (x_1^*, x_2^*, ..., x_n^*)$ un $y^* = (y_1^*, y_2^*, ..., y_m^*)$ ir atbilstoši *optimālie* plāni tiešajam uzdevumam (3.1.1) un duālajam uzdevumam (3.1.2). Ja $y_i^* > 0$, tad

$$a_{i1}x_1^* + a_{i2}x_2^* + ... + a_{in}x_n^* = b_i. \tag{3.2.7}$$

Un otrādāk, ja x^* un y^* ir atbilstoši *pieļaujamie* plāni tiešajam uzdevumam (3.1.1) un duālajam uzdevumam (3.1.2) un ja $y_i^* > 0$ izpildās (3.2.7), tad x^* un y^* ir optimālie plāni.

Pierādījums. $\Rightarrow$ Tā kā x^* pieder pieļaujamo plānu kopai D, tad $Ax^* \leq b$. Pareizināsim šo nevienādību skalāri ar vektoru $y^* \geq 0$, iegūsim

$$< c, x^* > \leq < A^T y^*, x^* > = < y^*, Ax^* > \leq < b, y^* > .$$

Pēc dualitātes teorēmas $< c, x^* > = < b, y^* >$, tad $< y^*, Ax^* > = < b, y^* >$. No šejienes seko, ka $< y^*, -Ax^* + b > = 0$. Vektors $r = -Ax^* + b$ ir nenegatīvs, jo $Ax^* \leq b$. Skalārais reizinājums $< y^*, r > = \sum\limits_{i=1}^{m} y_i^* r_i$ būs vienāds ar 0 tikai tad, ja vienādības $y_i^* r_i = 0$ izpildās visiem $i = 1, 2, ..., m$. Ja $y_i^* > 0$, tad

$$0 = r_i = (b - Ax^*)_i = b_i - \sum_{j=1}^{n} a_{ij} x_j^*,$$

kas arī bija jāpierāda.

$\Leftarrow$ No nosacījuma (3.2.7) seko vienādība $< y^*, Ax^* >=< y^*, b >$. Tā kā $< y^*, Ax^* >=< A^T y^*, x^* >\geq< c, x^* >$, tad iegūstam nevienādību $< c, x^* >\leq< b, y^* >$, kas saskaņā ar Lemmu 3.2.1 nozīmē, ka izpildās vienādība, un ja tā, tad saskaņā ar Lemmu 3.2.2 x^* un y^* ir optimālie plāni. ∎

Vispārīgā gadījumā uzdevumiem (3.1.1) un (3.1.2) var būt vairāki atrisinājumi. Jāievēro, ka līdzsvara teorēmā tiek runāts par jebkuru optimālo atrisinājumu pāri x^* un y^*.

Atzīmēsim dažas sekas no dualitātes un līdzsvara teorēmām.

Teorēma 3.2.3. Ja vienam no savstarpēji duālajiem uzdevumiem eksistē optimālais plāns, tad eksistē optimālais plāns arī otram.

Pierādījums. Pieņemsim, ka tiešajam uzdevumam (3.1.1) eksistē atrisinājums x^*. Iespējami divi gadījumi.

1) Duālais uzdevums (3.1.2) ir pieļaujams. Tādā gadījumā kopa D^u nav tukša, mērķa funkcija $< b, y >$ ir ierobežota no apakšas ar skaitli $< c, x^* >$ (pēc Lemmas 3.2.1), tad tā sasniedz savu minimālo vērtību, t.i., duālajam uzdevumam eksistē optimālais plāns.

2) Duālais uzdevums (3.1.2) nav pieļaujams (sistēmai (3.1.2) neeksistē nenegatīvs atrisinājums). Pēc nevienādību alternatīvas tas nozīmē, ka nevienādību sistēmai $Ax \leq 0$, $< c, x >> 0$ eksistē atrisinājums $\overline{x}$. Apskatīsim vektoru $\tilde{x} = x^* + \overline{x}$. Skaidrs, ka $A\tilde{x} \leq b$, t.i., $\tilde{x} \in D$. Pie tam

$$< c, \tilde{x} >=< c, x^* > + < c, \overline{x} >>< c, x^* >,$$

kas ir pretrunā ar x^* optimalitāti (maksimums!). Tādējādi šis otrs gadījums nav iespējams. ∎

Teorēma 3.2.3 parāda, ka savstarpēji duālajiem uzdevumiem ir iespējami šādi gadījumi:

a) abi uzdevumi ir pieļaujami (t.i., abiem eksistē atrisinājumi);

b) abi uzdevumi nav pieļaujami;

c) viens uzdevums ir pieļaujams, bet otrs ne.

Vienkārši piemēri parāda, ka visas trīs situācijas ir iespējamas.

a) Piemēram šādam uzdevumu pārim

tiešais uzdevums	duālais uzdevums
$\max(3x_1 + 2x_2)$	$\min(6y_1 + 4y_2)$
$3x_1 + x_2 \leq 6,$	$3y_1 + y_2 \geq 3,$
$x_1 + 2x_2 \leq 4,$	$y_1 + 2y_2 \geq 2,$
$x_1,\, x_2 \geq 0.$	$y_1,\, y_2 \geq 0.$

eksistē atrisinājumi. Tiešā uzdevuma optimālais plāns ir $x^* = (\frac{8}{5}, \frac{6}{5})$, duālā uzdevuma optimālais plāns ir $y^* = (\frac{4}{5}, \frac{3}{5})$, mērķa funkciju vērtības ir vienādas (un tā tam pēc teorijas ir jābūt!) — $\frac{36}{5}$.

b) Abi nav pieļaujami, piemēram, šādā gadījumā (lai par to pārliecinātos, var risinājumu meklēt grafiski)

tiešais uzdevums	duālais uzdevums
$\max(x_1 + 3x_2)$	$\min(3y_1 - 4y_2)$
$x_1 - x_2 \leq 3,$	$y_1 - y_2 \geq 1,$
$-x_1 + x_2 \leq -4,$	$-y_1 + y_2 \geq 3,$
$x_1,\ x_2 \geq 0.$	$y_1,\ y_2 \geq 0.$

c) Vienkāršs piemērs gadījumam, kad tiešais uzdevums ir pieļaujams, bet tam neeksistē optimālais plāns mērķa funkcijas neierobežotības dēļ, ir šāds

tiešais uzdevums	duālais uzdevums
$\max x_2$	$\min y_1$
$x_1 - x_2 \leq 1,$	$y_1 \geq 0,$
$x_1,\ x_2 \geq 0.$	$-y_1 \geq 1.$

Duālais uzdevums šajā gadījumā nav pieļaujams.

3.3 DUĀLĀ UZDEVUMA ATRISINĀŠANA UN EKONOMISKĀ INTERPRETĀCIJA

Duālā uzdevuma atrisinājumu praktiski atrod reizē ar tiešā uzdevuma atrisinājumu, ja tas tiek meklēts ar simpleksa metodi. Bez tam duālajam uzdevumam var piemeklēt glītu ekonomisko interpretāciju. Šajā nolūkā apskatīsim vienu no iespējām.

Piemērs 3.3.1. Pieņemsim, ka dots lineārās programmēšanas uzdevums standartformā.

$$\begin{aligned}
&\max(2x_1 + 3x_2)\\
&4x_1 + 5x_2 \leq 20,\\
&3x_1 + 6x_2 \leq 18,\\
&x_1 + 4x_2 \leq 10,\\
&x_1 \geq 0,\ x_2 \geq 0.
\end{aligned} \tag{3.3.1}$$

Ekonomiski šo uzdevumu varam interpretēt šādi (Kantoroviča modelis): uzņēmums ražo divu veidu produkcijas no trim dažādām izejvielām, kuras ir ierobežotā daudzumā, proti, 20, 18 un 10 vienības no katra resursa ir uzņēmuma riecībā apskatāmajā laika periodā. Pirmās produkcijas ražošanai vajag 4 vienības no pirmā resursa, 3 vienības no otrā resursa un 1 vienību no trešā resursa, attiecīgi otrajai produkcijai šie skaitļi ir 5, 6, 4. Cik daudz produkcijas var saražot, lai gūtu maksimālo ienākumu, ja pirmās produkcijas cena ir 2 naudas vienības par vienu produkcijas vienību, bet otrās produkcijas cena ir 3 naudas vienības par vienu produkcijas vienību?

Tātad matrica A sastāv no tā saucamajiem tehnoloģijas koeficientiem, vektora b koordinātas norāda resursu ierobežojumus, vektora c koordinātas ir produkcijas cenas, bet $x = (x_1, x_2)$ šeit ir saražojamās produkcijas daudzums (plāns).

Uzdevuma (3.3.1) duālo uzdevumu varam pierakstīt šādi:

$$\begin{aligned}
&\min(20y_1 + 18y_2 + 10y_3)\\
&4y_1 + 3y_2 + y_1 \geq 2,\\
&5y_1 + 6y_2 + 4y_3 \geq 3,\\
&y_1,\ y_2 \geq 0.
\end{aligned} \tag{3.3.2}$$

Vektors y šeit ir interpretējams kā ražošanas resursu cenas. Ar uzdevumu (3.3.2) tiek noteiktas tādas resursu cenas, lai varētu sekmīgi realizēt uzņēmuma uzdevumu (3.3.1). Ja uzņēmums par resursiem maksās vairāk, tad tas neiegūs plānotos maksimālos ienākumus, savukār, ja resursus var iepirkt par zemākām cenām, tad uzņēmuma ienākumi būs lielāki. Tāpēc y dažkārt sauc par resursu "ēnu" cenām.

Abu uzdevumu risināšanu var apvienot vienā simpleksa tabulā. Praktiski nekas nemainās salīdzinājumā ar iepriekš risinātajiem uzdevumiem, tikai tabulu noformējumā papildus tiek klāt pierakstīti duālo mainīgo un papildmainīgo kolonnu apzīmējumi. Risināts tiek tiešais uzdevums ar iepriekš aprakstīto simpleksa metodes algoritmu, līdzi atbilstoši nomainot duālā uzdevuma kolonnas.

Tā kā tiešais uzdevums (3.3.1) sākotnēji bija dots standartformā, tad tā risināšana tiek veikta līdzīgi kā Piemērā 2.7.4.

Ja tiešo uzdevumu (3.3.1) risinām ar simpleksa metodes palīdzību, tad vispirms veicam pāreju uz kanonisko formu

$$\begin{aligned}
&\max(2x_1 + 3x_2)\\
&4x_1 + 5x_2 + v_1 = 20,\\
&3x_1 + 6x_2 + v_2 = 18,\\
&x_1 + 4x_2 + v_3 = 10,\\
&x_1,\, x_2,\, v_1,\, v_2,\, v_3 \geq 0.
\end{aligned}$$

Duālā uzdevuma (3.3.2) kanoniskā forma ir

$$\begin{aligned}
&\min(20y_1 + 18y_2 + 10y_3)\\
&4y_1 + 3y_2 + y_1 - u_1 = 2,\\
&5y_1 + 6y_2 + 4y_3 - u_2 = 3,\\
&y_1,\, y_2,\, u_1,\, u_2 \geq 0.
\end{aligned}$$

Tā kā šajā gadījumā $c^\beta = (0, 0, 0)$, tad pirmā simpleksa tabula tiek noformēta šādi

$T^{(1)} =$

	b	a^1	a^2	v^1	v^2	v^3	
	0	-2	-3	0	0	0	
v^1	20	**4**	5	1	0	0	y^1
v^2	18	3	6	0	1	0	y^2
v^3	10	1	4	0	0	1	y^3
		u^1	u^2	y^1	y^2	y^3	

Pēc Blenda algoritma bāzē ir jāiekļauj pirmā kolonna a^1. Tā kā $\frac{20}{4} = 5 < \frac{18}{3} = 6 < \frac{10}{1}$, tad no bāzes tiks izslēgta kolonna v^1. Atbilstoši tiek nomainīts duālais mainīgais y^1 ar u^1.

$T^{(2)} =$

	b	a^1	a^2	v^1	v^2	v^3	
	10	0	$-\frac{1}{2}$	$\frac{1}{2}$	0	0	
a^1	5	1	$\frac{5}{4}$	$\frac{1}{4}$	0	0	u^1
v^2	3	0	$\mathbf{\frac{9}{4}}$	$-\frac{3}{4}$	1	0	y^2
v^3	5	0	$\frac{11}{4}$	$-\frac{1}{4}$	0	1	y^3
		u^1	u^2	y^1	y^2	y^3	

Bāzē ir jāiekļauj otrā kolonna a^2. Tā kā $5 : \frac{5}{4} = 4 > 3 : \frac{9}{4} = \frac{4}{3} < 5 : \frac{11}{4} = \frac{20}{11}$, tad no bāzes tiks izslēgta kolonna v^2. Atbilstoši tiek nomainīts duālais mainīgais y^2 ar u^2.

$$T^{(3)} = \begin{array}{c|c|cccccc|c}
 & b & a^1 & a^2 & v^1 & v^2 & v^3 & \\
\hline
 & \frac{32}{3} & 0 & 0 & \frac{1}{3} & \frac{2}{9} & 0 & \\
\hline
a^1 & \frac{10}{3} & 1 & 0 & \frac{2}{3} & -\frac{5}{9} & 0 & u^1 \\
a^2 & \frac{4}{3} & 0 & 1 & -\frac{1}{3} & \frac{4}{9} & 0 & u^2 \\
v^3 & \frac{4}{3} & 0 & 0 & \frac{2}{3} & -\frac{11}{9} & 1 & y^3 \\
\hline
 & & u^1 & u^2 & y^1 & y^2 & y^3 &
\end{array}$$

Esam ieguvuši tiešā uzdevuma atrisinājumu $x^* = (\frac{10}{3}, \frac{4}{3}$ ar mērķa funkcijas vērtību $\frac{32}{3} = 10\frac{2}{3}$. No tabulas $T^{(3)}$ var nolasīt arī papildmainīgā v koordinātas $v = (0, 0, \frac{4}{3})$, tās rāda ražošanas vektoru pārpalikumus (šajā gadījumā pie optimālā plāna pāri paliks trešais resurss). Duālā uzdevuma atrisinājums ir atrodams novērtējumu rindā un tas ir $y = (\frac{1}{3}, \frac{2}{9}, 0)$. ■

3.4 DUĀLAIS SIMPLEKSA ALGORITMS

Standartformas tiešā un duālā uzdevuma savstarpējā dualitāte ļauj modificēt simpleksa algoritmu tā, lai varētu atrisināt tādus lineārās programmēšanas uzdevumus, kurus tiešā veidā nevar risināt ar simpleksa algoritmu, ja neizpildās nenegativitātes nosacījumi. Kā to var izdarīt, mēģināsim nodemonstrēt ar piemēra palīdzību.

Piemērs 3.4.1. Pieņemsim, ka jāatrisina lineārās programmēšanas uzdevums

$$\begin{aligned}
&\min(3x_1 + 2x_2)\\
&2x_1 + x_2 \geq 2,\\
&x_1 + 3x_2 \geq 3,\\
&5x_1 + 4x_2 \leq 20,\\
&x_1,\, x_2 \geq 0.
\end{aligned}$$

Pēc iepriekšējās nodaļās aprakstītā šajā gadījumā ir jāpāriet uz standartformu

$$\begin{aligned}
&\max(-3x_1 - 2x_2)\\
&-2x_1 - x_2 \leq -2,\\
&-x_1 - 3x_2 \leq -3,\\
&5x_1 + 4x_2 \leq 20,\\
&x_1,\, x_2 \geq 0,
\end{aligned}$$

kuru savukārt jāpārraksta kanoniskajā formā

$$\begin{aligned}
&\max(-3x_1 - 2x_2)\\
&-2x_1 - x_2 + v_1 = -2,\\
&-x_1 - 3x_2 + v_2 = -3,\\
&5x_1 + 4x_2 + v_3 = 20,\\
&x_1,\ x_2,\ v_1,\ v_2,\ v_3 \geq 0.
\end{aligned}$$

Pēc iepriekšējiem priekšrakstiem varam sastādīt pirmo simpleksa tabulu

$$T^{(1)} = \begin{array}{c|c|cccccc}
 & b & a^1 & a^2 & v^1 & v^2 & v^3 \\
\hline
 & 0 & 3 & 2 & 0 & 0 & 0 \\
\hline
v^1 & \text{-2} & \textbf{-2} & \text{-1} & 1 & 0 & 0 \\
v^2 & \text{-3} & \text{-1} & \text{-3} & 0 & 1 & 0 \\
v^3 & 20 & 5 & 4 & 0 & 0 & 1
\end{array}$$

Šajā tabulā parādās divas pretrunīgas lietas: novērtējumu rindā nav negatīvu skaitļu — tātad it kā atrisinājums būtu optimāls, bet vektors $(0, 0, -2, -3, 20)$ neder par optimālo plānu, jo tas nav pieļaujams (divas koordinātas negatīvas). Bet, ja sākotnējo uzdevumu risinātu ar ģeometriskiem paņēmieniem, kas šajā reizē ir iespējams, tad mēs redzētu, ka pieļaujamo plānu kopa nav tukša un uzdevumam ir viens konkrēts atrisinājums (lasītājam ieteicams pašam uzzīmēt un atrast optimālo plānu). Kā uzdevumu atrisināt ar duālo simpleksa metodi?

$\star$ Dotajā tabulā $T^{(1)}$ novērtējumu rindas vietā jāapskata kolonna b. No bāzes izslēdzamais mainīgais ir tas, kura novērtējums ir negatīvs un lielākais pēc absolūtās vērtības vai pēc Blenda algoritma — ar mazāko indeksu. Pēc šī kritērija jāizslēdz v^1 (mēs centīsimies pieturēties Blenda algoritma tradīcijām). Ja b kolonnā visi skaitļi ir nenegatīvi, tad ir atrasts optimālais plāns.

$\star$ Bāzē iekļaujamais mainīgais tiek izvēlēts starp nebāzes mainīgajiem. Pieņemsim, ka k norāda to rindu, kuras mainīgais pēc augstākā apraksta ir jāizslēdz no bāzes. Ja visi $a_{ki} \geq 0$, $i = 1, 2, ..., n$, tad uzdevumam neeksistē optimālais atrisinājums. Ja ir tādi nebāzes mainīgo koeficienti $a_{ki} < 0$, tad izskaitļo attiecības $\lambda_i = \left|\frac{a_{0,i}}{a_{ki}}\right|$ un izvēlas to i, kuram λ_i ir mazākais (nenoteiktības gadījumā izvēlamies ar mazāko indeksu). Konkrētajā uzdevumā $\lambda_1 = \frac{3}{2} < \lambda_2 = 2$, tātad bāzē iekļauj a^1.

$\star$ Pāreja uz nākamo tabulu notiek pēc simpleksa algoritmā aprakstītajiem soļiem.

$\star$ Mērķa funkcijas vērtība tabulās būs ar nepareizu zīmi!
Konkrētajā piemērā nākamā tabula ir

$$T^{(2)} = \begin{array}{c|c|cccccc} & b & a^1 & a^2 & v^1 & v^2 & v^3 \\ \hline & -3 & 0 & \frac{1}{2} & \frac{3}{2} & 0 & 0 \\ \hline a^1 & 1 & 1 & \frac{1}{2} & -\frac{1}{2} & 0 & 0 \\ v^2 & -2 & 0 & -\mathbf{\frac{5}{2}} & -\frac{1}{2} & 1 & 0 \\ v^3 & 15 & 0 & \frac{3}{2} & \frac{5}{2} & 0 & 1 \end{array}$$

Tālāk jāsamaina v^2 ar a^2.

$$T^{(3)} = \begin{array}{c|c|cccccc} & b & a^1 & a^2 & v^1 & v^2 & v^3 \\ \hline & -\frac{17}{5} & 0 & 0 & \frac{7}{5} & \frac{1}{5} & 0 \\ \hline a^1 & \frac{3}{5} & & & & & \\ a^2 & \frac{4}{5} & & & & & \\ v^3 & \frac{69}{5} & & & & & \end{array}$$

Šajā tabulā nav aizpildīta koeficientu matrica, jo esam ieguvuši optimālo atrisinājumu — visi novērtējumi b kolonnā ir nenegatīvi. Optimālais plāns ir $x^* = (\frac{3}{5}, \frac{4}{5})$, sākotnējā uzdevuma mērķa funkcijas vērtība ir $\frac{17}{5}$. Duālā uzdevuma atrisinājums ir $y^* = (\frac{7}{5}, \frac{1}{5}, 0)$. $\blacksquare$

NODAĻA 4

LINEĀRĀ PROGRAMMĒŠANA VESELOS SKAITĻOS

4.1 LINEĀRĀS PROGRAMMĒŠANAS UZDEVUMU PIEMĒRI VESELOS SKAITĻOS

Piemērs 4.1.1 (mugursomas pakošanas uzdevums). Ceļotājs gatavojas pārgājienam. Viņam nepieciešams sakravāt mugursomā n dažāda nosaukuma lietas, pie tam var būt nepieciešams somā ielikt vairākas viena nosaukuma lietas. Mogursoma ir ar noteiktiem izmēriem, tilpumu, ceļotājs var panest noteiktu svaru; kopumā ir m dažādi ierobežojumi. Apzīmēsim ar a_{ij} j-tā priekšmeta i-to raksturojumu, $i = 1, 2, ..., m$, $j = 1, 2, ..., n$. Apzīmēsim ar b_i, $i = 1, 2, ...m$, dažādo ierobežojumu (svara, izmēra, tilpuma, utt.) skaitliskos rādītājus. Apzīmēsim ar x_j j-tā priekšmeta skaitu, ko ceļotājs plāno ielikt mugursomā, $j = 1, 2, ..., n$. Uzskatīsim, ka ceļotājam ir zināms j-tā priekšmeta vienas vienības derīgums c_j. Tādā gadījumā mugursomas pakošanas uzdevums matemātiski ir formulējams šādi:

$$\max \sum_{j=1}^{n} c_j x_j$$

$$\sum_{j=1}^{n} a_{ij} x_j \leq b_i, \quad i = 1, 2, ..., m,$$

$$x_j \geq 0, \quad x_j - \text{ vesels skaitlis }, \; j = 1, 2, ..., n.$$

Redzams, ka dotais uzdevums nav lineārās programmēšanas uzdevums, jo prasību par plāna koordinātām veselos skaitļos nevar aprakstīt ar lineārām nevienādībām. Var izveidot arī citus šādus līdzīga rakstura uzdevumus ar praktisku lietojumu. ■

Piemērs 4.1.2 (komivojažiera uzdevums). (Komivojažieris — ceļojošs tirdzniecības firmas pārstāvis, kas piedāvā pircējiem preces pēc paraugiem.) Pieņemsim, ka ir n dažādas pilsētas, kas sanumurētas ar skaitļiem $0, 1, 2, ..., n$. Komivojažieris izbrauc no 0-tās pilsētas, viņa pienākums ir apmeklēt visas pārējās pilsētas, pie tam katrā pilsētā jābūt tikai vienreiz; beigās jāatgriežas 0-tajā pilsētā. Ir zināmi attālumi c_{ij} starp pilsētām i, j (i, $j = 0, 1, 2, ..., n$). Nepieciešams noskaidrot īsāko maršrutu.

Apzīmēsim

$$x_{ij} = \begin{cases} 1, & \text{ja komivojažieris ceļo no pilsētas } i \text{ uz pilsētu } j, \; i \neq j, \\ 0, & \text{pretējā gadījumā,} \end{cases}$$

$$i, \, j = 0, 1, 2, ..., n.$$

Uzdevuma matemātiskajam aprakstam būs nepieciešami papildus mainīgie u_i, $i = 1, 2, ..., n$. Uzdevumu var pierakstīt šādi

$$\min \sum_{i=0}^{n} \sum_{j=0}^{n} c_{ij} x_{ij}$$

$$\begin{array}{l} \sum\limits_{i=0}^{n} x_{ij} = 1, \quad j = 1, 2, ..., n, \\ \sum\limits_{j=0}^{n} x_{ij} = 1, \quad i = 1, 2, ..., n, \\ u_i - u_j + n x_{ij} \leq n - 1, \quad i, j = 1, 2, ..., n, \\ x_{ij} \text{ ir 0 vai 1.} \end{array} \tag{4.1.1}$$

Mainīgie u_i, $i = 1, 2, ..., n$, atrodas tikai uzdevumu prasību sistēmā, bet neietilpst mērķa funkcijā. Šiem papildus mainīgajiem ir palīgloma, bet tanī pašā laikā bez tiem nevar iztikt.

Pārliecināsimies, ka formālais modelis (4.1.1) ir ekvivalents ar komivojažiera uzdevumu. Pirmā vienādība nozīmē, ka komivojažieris uz pilsētu $j \neq 0$ atbrauc tikai vienu reizi. Otrā vienādība nozīmē, ka komivojažieris no pilsētas $j \neq 0$ izbrauc tikai vienu reizi. Taču abi šie nosacījumi kopā negarantē, ka izvēlētas maršruts ir saistīts. Izrādās, ka saistību nodrošina nevienādības ar papildus mainīgajiem. Šie nosacījumi aizliedz jebkādu ciklu, kas neiet caur 0-to pilsētu.

Varam pierādīt, ka jebkuram komivojažiera maršrutam atbilst uzdevuma (4.1.1) pieļaujams plāns.

Izvēlēsimies patvaļīgu ciklu, kas sākas 0-tajā pilsētā

$$l_0\, l_1\, l_2 ... l_r ... l_n\, l_0,$$

kur $l_0 = 0$, bet visi l_r ir atšķirīgi skaitļi starp 1 un n.

Apskatīsim sekojošu uzdevuma plānu: skaitļus x_{ij} izvēlamies kā iepriekš tas definēts, bet skaitļus u_i definējam pēc formulas: ja $i = l_r$, tad $u_i = r$, $i = 1, 2, ..., n$. Citiem vārdiem sakot, skaitlis u_i ir vienāds ar i-tās pilsētas apmeklēšanas kārtas numuru paredzētajā maršrutā. Parādīsim, ka tā izvēlēti skaitļiu x_{ij}, $i, j = 0, 1, 2, ..., n$, u_i, $i = 1, 2, ..., n$, ir uzdevuma (4.1.1) plāns.

Ievērosim, tā kā $1 \leq u_i \leq n$, tad $u_i - u_j \leq n - 1$ visiem $i, j = 1, 2, ..., n$. No šejienes seko, ka izpildās nevienādība $u_i - u_j + nx_{ij} \leq n - 1, \quad i, j = 1, 2, ..., n$, ja $x_{ij} = 0$. Ja $x_{ij} = 1$, $i, j \neq 0$, tad pēc dotā plāna konstrukcijas eksistē tāds r, $1 \leq r \leq n - 1$, ka $i = l_r$, $j = l_{r+1}$. Tādā gadījumā $u_i = r$, $u_j = r + 1$ un $u_i - u_j = 1$. Seko, ka $u_i - u_j + nx_{ij} = n - 1$. ∎

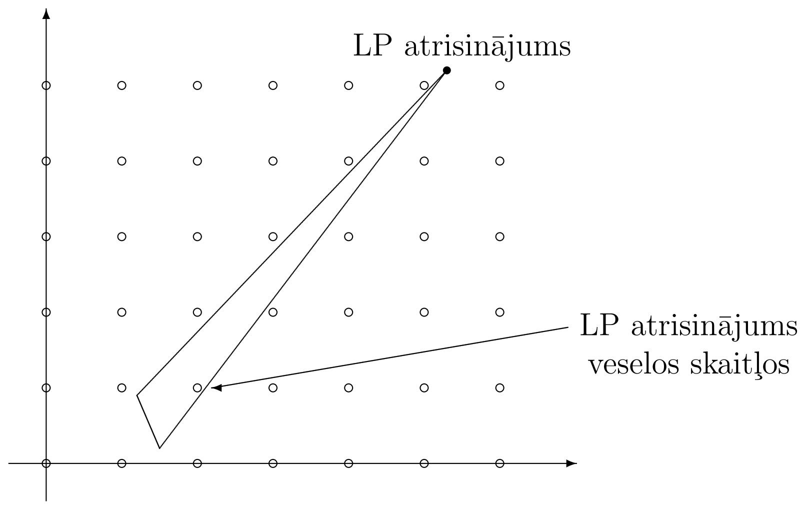
Dotie divi piemēri ir īpaši lineārās programmēšanas uzdevumi, jo optimālo plānu koordinātām jābūt veseliem skaitļiem. Atrisinot lineārās programmēšanas uzdevumus ar simpleksa metodi netiek garantēts, ka optimālā plāna koordinātas būs veseli skaitļi. Dažkārt šādos gadījumois ir pieļaujama rezultātu noapaļošana. Piemēram, ja optimālajā plānā paredzēts, ka jāsažo $567, 4$ maizes kukuļi, tad ekonomiski pilnīgi attaisnojama ir rezultāta noapaļošana. Tomēr praksē ir sastopami tādi gadījumi, kur šāda noapaļošana nav ieteicama, jo var radīt lielas kļūdas. Piemēram, ja optimālajā plānā paredzēts, ka valstī jāuzbūvē $1, 76$ elektrostacijas, tad formāla noapaļošana uz 1 vai 2 nav ekonomiski pieļaujama.

Praktiska un teorētiska nozīme ir lineārās programmēšanas uzdevumu atrisināšanas metodēm, ar kuru palīdzību var atrast optimālo plānu, kura koordinātas ir veseli skaitļi. Lineārās programmēšanas uzdevumus, kuru plāniem jābūt ar veselu skaitļu koordinātām, sauc par *lineārās programmēšanas uzdevumiem veselos skaitļos.*

Lineārās programmēšanas uzdevumiem veselos skaitļos var būt vairākas pieraksta formas. Lineārās programmēšanas uzdevumiem veselos skaitļos ir uzdots kanoniskā formā, ja tas formulēts šādi:

$$\max < c, x >$$
$$Ax = b,$$
$$x \geq 0,$$
$$x_i,\ i = 1, 2, ..., n, \text{ — veseli skaitļi.}$$

Atšķirība no parasta lineārās programmēšanas uzdevuma redzama tikai vienā papildus nosacījumā — atrisinājumam jābūt veselos skaitļos.



4.1.1. zīm.

4.1.1.zīmējumā redzama situācija, kad lineārās programmēšanas (LP) uzdevuma atrisinājums (maksimums, divu mainīgo gadījums) būtiski atšķiras no lineārās programmēšanas uzdevuma veselos skaitļos atrisinājuma (tas nav iegūstams kā koordinātu noapaļošanas sekas). Ierobežotais apgabals reprezentē lineārās programmēšanas uzdevuma pieļaujamo plānu kopu. Varam redzēt, ka šajā kopā ir tikai viens punkts ar veselām koordinātām (punkti ar tukšajiem aplīšiem reprezentē veselās koordinātas saturošos punktus). Skatoties uz šo zīmējumu, kļūst skaidrs, ka var rasties pat tādas situācijas, kad lineārās programmēšanas uzdevumam pieļaujamo plānu kopa nav tukša, ir ierobežota, eksistē optimālais plāns, bet atbilstošajam lineārās programmēšanas uzdevumam veselos skaitļos nav atrisinājuma (nav neviena plāna ar veselām koordinātām, kas piederētu atbilstošā lineārās programmēšanas uzdevuma pieļaujamo plānu kopai).

4.2 GOMORI METODE

Lineārās programmēšanas uzdevumu veselos skaitļos atrisināšanas metodes ir iedalāmas divās kategorijās (kas var arī daļēji šķelties): atšķeļošās plaknes algoritmi, kas izmanto simpleksa algoritmu, un pārlases algoritmi, kas veic saprātīgu visu iespējamo atrisinājumu pārlasi. Šajā nodaļā mēs apskatīsim pirmās kategorijas algoritmus.

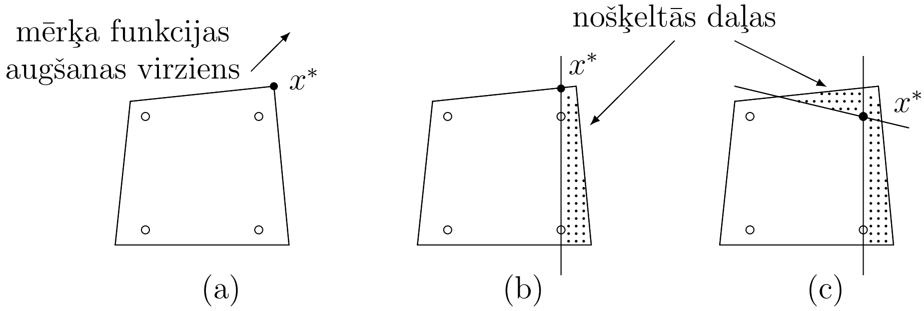
Pieņemsim, ka mums ir dots lineārās programmēšanas uzdevums veselos skaitļos

$$\begin{gathered} \max < c, x > \\ Ax = b, \\ x \geq 0, \\ x_i,\ i = 1, 2, ..., n, \text{ — veseli skaitļi.} \end{gathered} \tag{4.2.1}$$

Reizē ar (4.2.1) apskatīsim atbilstošo lineārās programmēšanas uzdevumu, kura atrisinājumiem nav uzlikts nosacījums par veseliem skaitļiem,

$$\begin{gathered} \max < c, x > \\ Ax = b, \\ x \geq 0. \end{gathered} \tag{4.2.2}$$

Atzīmēsim divus vienkāršus, bet noderīgus faktus. Pirmais fakts — ja (4.2.2) uzdevuma atrisinājums izrādās ir atrisinājums veselos skaitļos, tad tas ir arī (4.2.1) atrisinājums. Otrais fakts — (4.2.2) uzdevuma optimālā plāna mērķa funkcijas vērtība ir augšējā robeža (4.2.1) uzdevuma mērķa funkcijai. Šīs vienkāršās idejas tiek izmantotas atšķeļošās plaknes algoritmos. Ja (4.2.1) uzdevumam pievienotu ierobežojumus, kuri no pieļaujamo plānu kopas neizslēdz plānus ar veselo skaitļu koordinātām, tad uzdevuma atrisinājums neizmainītos. Tāpēc tālākā mūsu stratēģija būs tāda, ka mēs pievienosim linārās programmēšanas uzdevumam veselos skaitļos tādus lineārus nosacījumus, līdz atbilstošajam lineārās programmēšanas uzdevumam (bez prasības par atrisinājumu veselos skaitļos) būs atrisinājums veselos skaitļos. Tā kā mēs neizslēdzam pieļaujamos plānus ar veselām koordinātām, tad šis gala rezultātā iegūtais atrisinājums būs sākotnējā uzdevuma atrisinājums. Aprakstītais process ir ilustrēts 4.2.1.zīmējumā. (a) zīmējumā parādīts sākotnējais lineārās programmēšanas uzdevuma veselos skaitļos un redzams atbilstošā lineārā uzdevuma atrisinājums x^* (punkts, kurā mērķa funkcija sasniedz maksimālo vērtību), (b) zīmējumā pievienots lineārs ierobežojums, ko sauc par atšķeļošās plaknes ierobežojumu un kas no pieļaujamās kopas neizslēdz nevienu punktu ar veselām koordinātām, un parādīts atbilstošā lineārā uzdevuma atrisinājums x^*, (c) zīmējumā pievienots vēl viens atšķeļošās plaknes ierobežojums, kā rezultātā atbilstošā linērās programmēšanas uzdevuma optimālais plāns ir ar veselām koordinātām un tas ir arī sākotnējā lineārās programmēšanas uzdevuma veselos skaitļos atrisinājums.



4.2.1. zīm.

Aprakstīsim algebrisko metodi atšķēlumu ģenerēšanai – Gomori metodi, kuras apraksts atrodams *R.E.Gomory* 1958.gadā publicētajā rakstā "Outline of an Algorithm for Integer Solution to Linear Programs", Bulletin Amer. Math. Soc., V.64. Pieņemsim, ka dots lineārās programmēšanas uzdevums veselos skaitļos (4.2.1), un apskatīsim tam atbilstošo lineārās programmēšanas uzdevumu (4.2.2) bez prasības par atrisinājumu veselos skaitļos. Atrisināsim (4.2.2) ar simpleksa algoritmu, tādējādi atradīsim (4.2.2) optimālo plānu x, kuram atbilst bāze $\beta^{(q)}$. Noslēguma simpleksa tabulas tipisks vienādojums ir izskatā

$$b_i = x_{\beta^{(q)}} + \sum_i \alpha_{ij} v_j, \tag{4.2.3}$$

$0 \le i \le m$. Izmantosim sekojošus apzīmējumus.

Definīcija 4.2.1. Par reāla skaitļa a *veselo daļu* $\lfloor a \rfloor$ sauc vislielāko veselo skaitli, kas nepārsniedz skaitli a. Par reāla skaitļa a *daļveida daļu* $\{a\}$ sauc starpību $\{a\} = a - \lfloor a \rfloor$.

Piemēram, $\lfloor 16,4 \rfloor = 16$, $\lfloor 0,63 \rfloor = 0$, $\lfloor 0 \rfloor = 0$, $\lfloor 4 \rfloor = 4$, bet, ievērojiet, ka $\lfloor -2,7 \rfloor = -3$. Savukārt $\{6,45\} = 0,45$, $\{23\} = 0$, bet $\{-6,45\} = -6,45 - (-7) = 0,55$.

Mainīgajam x formulā (4.2.3) ir jābūt nenegatīvam, tāpēc

$$\sum_i \lfloor \alpha_{ij} \rfloor v_j \le \sum_i \alpha_{ij} v_j. \tag{4.2.4}$$

Tādā gadījumā no (4.2.3) seko, ka

$$x_{\beta(q)} + \sum_i \lfloor \alpha_{ij} \rfloor v_j \le b_i. \tag{4.2.5}$$

Uzdevumā (4.2.1) plānam x ir jābūt ar veselām koordinātām, tāpēc (4.2.5) kreisā puse ir vesels skaitlis, tāpēc, neizmainot nevienādību, var (4.2.5) labo pusi aizstāt ar tās veselo daļu, tas noved pie nevienādības

$$x_{\beta(q)} + \sum_i \lfloor \alpha_{ij} \rfloor v_j \le \lfloor b_i \rfloor. \tag{4.2.6}$$

Atņemot (4.2.6) no (4.2.3), iegūsim

$$\sum_i (\alpha_{ij} - \lfloor \alpha_{ij} \rfloor) v_j \le b_i - \lfloor b_i \rfloor \text{ jeb} \tag{4.2.7}$$

$$\sum_i \{\alpha_{ij}\} v_j \le \{b_i\}. \tag{4.2.8}$$

Šo pēdējo (4.2.8) ierobežojošo nevienādību sauc par *Gamora šķēlumu*, kas atbilst rindai i.

Uzdevuma (4.2.2) simpleksa noslēguma tabulai pievienosim Gomora šķēlumu (4.2.8). Lai saglabātu bāzes atrisinājumu, pareizināsim (4.2.8) ar -1 un pievienosim vēl vienu mainīgo s, rezultātā iegūsim

$$-\sum_i \{\alpha_{ij}\} v_j + s = -\{b_i\}. \tag{4.2.9}$$

Nākamā lemma apraksta rezultātu, kas tiek iegūts, ja (4.2.2) simpleksa noslēguma tabulai pievieno Gomora šķēlumu izskatā (4.2.9).

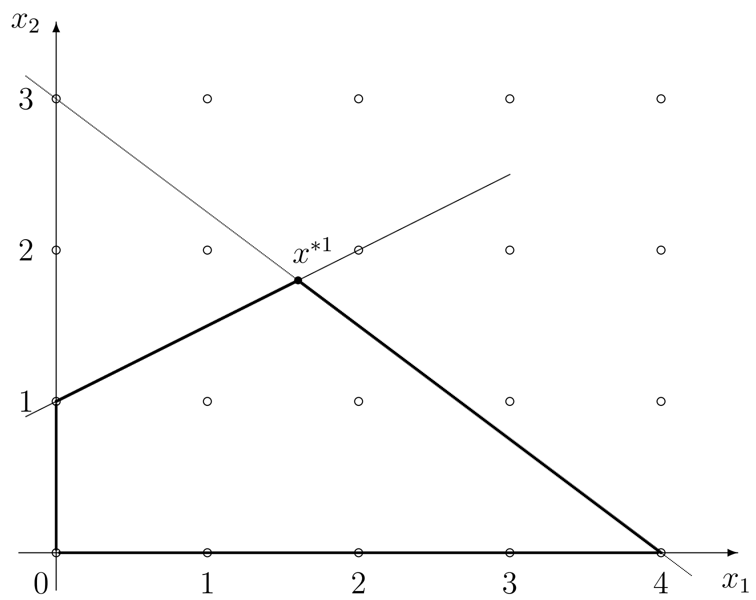
Lemma 4.2.1. Ja Gomora šķēlums (4.2.9) tiek pievienots uzdevuma (4.2.2) simpleksa noslēguma tabulai, tad
1) no pieļaujamo plānu kopas netiek izslēgti nekādi plāni ar veselajām koordinātām;
2) jaunā simpleksa tabula ir bāzes tabula, kas nav pieļaujama tiešajam uzdevumam, ja b_i nav vesels skaitlis, bet tā ir pieļaujama duālajam uzdevumam.

Pierādījums. Fakts, ka, pievienojot Gomora šķēlumu (4.2.9) uzdevuma (4.2.2) simpleksa noslēguma tabulai, no pieļaujamo plānu kopas netiek izslēgti nekādi plāni ar veselajām koordinātām, seko no secinājuma, ka šis šķēlums ir dotā uzdevuma (4.2.1) noapaļojums līdz veselajiem skaitļiem. Jaunais mainīgais s ir jauns bāzes mainīgais un kopā ar optimālo bāzi $\beta^{(q)}$ veido jauno bāzi. Ja b_i nav vesels skaitlis, tad $\{b_i\} > 0$, tāpēc bāzes atrisinājumā $s = -\{b_i\}$, no šejienes seko, ka šis mainīgais ir nepieļaujams tiešajam uzdevumam, bet tas ir pieļaujams duālajam uzdevumam, jo novērtējumu rinda nav izmainījusies. ■

No Lemmas 4.2.1 seko, kā rīkoties tālāk — jāizmanto duālā simpleksa algoritms; tā rezultātā pēc vienas vai vairākām bāzes maiņām nonāksim pie jauna atrisinājuma vai arī nāksies secināt, ka tiešais uzdevums nav pieļaujams. Šī pēdējā iespēja norāda uz to, ka sākuma lineārās programmēšanas uzdevumam veselos skaitļos nav neviena pieļaujama punkta ar veselām koordinātām.

Piemērs 4.2.1. Apskatīsim šādu lineārās programmēšanas uzdevumu veselos skaitļos

$$\max(x_1 + 2x_2)$$
$$3x_1 + 4x_2 \le 12,$$
$$-x_1 + 2x_2 \le 2,$$
$$x_1,\ x_2 \ge 0,$$
$$x_1,\ x_2 \text{ — veseli skaitļi}$$



4.2.2. zīm. Pirmais optimālais plāns $x^{*1} \notin \mathbf{Z}^2$.

Tā kā dotajā uzdevumā ir tikai divi nezināmie, tad vienlaicīgi ar algebrisko risinājumu ir iespējams sekot līdzi uzdevuma ģeometriskajai interpretācijai. Protams, realitātē konkrēto uzdevumu risināt ar Gamora metodi ir tikpat kā šaut uz zvirbuļiem ar lielgabalu, bet lasītāja izpratnes veicināšanai der tieši salīdzinoši vienkārši piemēri. Tātad sākotnējās situācijas ģeometriskā ilustrācija ir redzama 4.2.2.zīmējumā.

Atrisināsim dotā uzdevuma atbilstošo lineārās programmēšanas uzdevumu bez nosacījuma par atrisinājumu veselos skaitļos ar simpleksa metodi! Vispirms veiksim pāreju uz kanonisko formu

$$\begin{aligned}
&\max(x_1 + 2x_2)\\
&3x_1 + 4x_2 + v_1 = 12,\\
&-x_1 + 2x_2 + v_2 = 2,\\
&x_1,\ x_2,\ v_1,\ v_2 \geq 0.
\end{aligned}$$

Sastādīsim simpleksa pirmo tabulu un veiksim tālākas iterācijas ar simpleksa algoritmu, kamēr atradīsim atrisinājumu.

$$T^{(1)} = \begin{array}{c|c|cccc}
 & b & a^1 & a^2 & v^1 & v^2 \\
\hline
 & 0 & -1 & -2 & 0 & 0 \\
\hline
v^1 & 12 & \mathbf{3} & 4 & 1 & 0 \\
v^2 & 2 & -1 & 2 & 0 & 1
\end{array}$$

$$T^{(2)} = \begin{array}{c|c|cccc}
 & b & a^1 & a^2 & v^1 & v^2 \\
\hline
 & 4 & 0 & -\frac{2}{3} & \frac{1}{3} & 0 \\
\hline
a^1 & 4 & 1 & \frac{4}{3} & \frac{1}{3} & 0 \\
v^2 & 6 & 0 & \mathbf{\frac{10}{3}} & \frac{1}{3} & 1
\end{array}$$

$$T^{(3)} = \begin{array}{c|c|cccc}
 & b & a^1 & a^2 & v^1 & v^2 \\
\hline
 & \frac{26}{5} & 0 & 0 & \frac{2}{5} & \frac{1}{5} \\
\hline
a^1 & \frac{8}{5} & 1 & 0 & \frac{1}{5} & -\frac{2}{5} \\
a^2 & \frac{9}{5} & 0 & 1 & \frac{1}{10} & \frac{3}{10}
\end{array}$$

Iegūtā atrisinājuma $x^{*1} = (\frac{8}{5}, \frac{9}{5}$ abas koordinātas nav veseli skaitļi. Noskaidrosim, kāds izskatās Gomora šķēluma nosacījums, kurš jāpievieno simpleksa tabulai $T^{(3)}$. Principā te ir vairākas iespējas; mēs veiksim izmaiņas attiecībā pret pirmo koordinātu (var arī pret otro koordinātu, kā arī pret mērķa funkcijas vērtību).

$$\frac{8}{5} = x_1 + \frac{1}{5}v_1 - \frac{2}{5}v_2,$$

$$\left\{\frac{8}{5}\right\} \leq \{1\}\, x_1 + \left\{\frac{1}{5}\right\} v_1 + \left\{-\frac{2}{5}\right\} v_2,$$

$$\frac{3}{5} \leq \frac{1}{5}v_1 + \frac{3}{5}v_2, \tag{4.2.10}$$

izmantojot formulu (4.2.9), secinām, ka jāpievieno simpleksa tabulai šāda rinda

$$-\frac{3}{5} = -\frac{1}{5}v_1 - \frac{3}{5}v_2 + s_1.$$

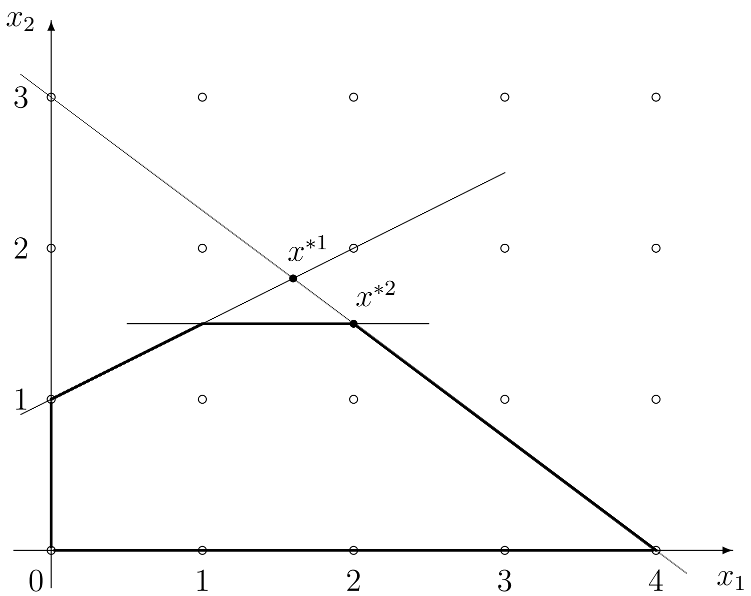
Jaunā tabula ir šāda

$$T^{(3)'} = \begin{array}{c|c|cccccc} & b & a^1 & a^2 & v^1 & v^2 & s^1 \\ \hline & \frac{26}{5} & 0 & 0 & \frac{2}{5} & \frac{1}{5} & 0 \\ \hline a^1 & \frac{8}{5} & 1 & 0 & \frac{1}{5} & -\frac{2}{5} & 0 \\ a^2 & \frac{9}{5} & 0 & 1 & \frac{1}{10} & \frac{3}{10} & 0 \\ s^1 & -\frac{3}{5} & 0 & 0 & -\frac{1}{5} & -\mathbf{\frac{3}{5}} & 1 \end{array}$$

No nevienādības (4.2.10) var izsecināt, kādu ierobežojumu esam pielikuši klāt sākotnējai sistēmai (citiem vārdiem sakot, ko esam nošķēluši nost no sākotnējās pieļaujamo plānu kopas):

$$\begin{aligned} &\tfrac{3}{5} \leq \tfrac{1}{5}v_1 + \tfrac{3}{5}v_2 = \tfrac{1}{5}(12 - 3x_1 - 4x_2) + \tfrac{3}{5}(2 + x_1 - 2x_2) = \\ &= \tfrac{12}{5} - \tfrac{3}{5}x_1 - \tfrac{4}{5}x_2 + \tfrac{6}{5} + \tfrac{3}{5}x_1 - \tfrac{6}{5}x_2 = \tfrac{18}{5} - 2x_2 \Rightarrow \\ &\Rightarrow x_2 \leq \tfrac{3}{2}. \end{aligned}$$

Zīmējumā tas izskatīsies šādi:



4.2.3. zīm.

No 4.2.3.zīmējuma redzams, ka pieļaujamo plānu kopas augšējā labās puses virsotne nav ar veselām koordinātām, tātad šis nebūs vienīgais Gomori šķēlums.

Simpleksa tabula $T^{(3)'}$ nav pieļaujama tiešajam uzdevumam, bet tā ir pieļaujama duālajam uzdevumam. Tad jāapmaina s^1 ar v^1 vai v^2:

$$\left|\frac{2\cdot 5}{5\cdot(-1)}\right| = 2 > \left|\frac{1\cdot 5}{5\cdot(-3)}\right| = \frac{1}{3},$$

tātad s^1 samainīsim ar v^2. Iegūsim tabulu

$$T^{(4)} = \begin{array}{c|c|cccccc} & b & a^1 & a^2 & v^1 & v^2 & s^1 \\ \hline & 5 & 0 & 0 & \frac{1}{3} & 0 & \frac{1}{3} \\ \hline a^1 & 2 & 1 & 0 & \frac{1}{3} & 0 & -\frac{2}{3} \\ a^2 & \frac{3}{2} & 0 & 1 & 0 & 0 & \frac{1}{2} \\ v^2 & 1 & 0 & 0 & \frac{1}{3} & 1 & -\mathbf{\frac{5}{3}} \end{array}$$

Iegūtā atrisinājuma $x^{*2} = (2, \frac{3}{2}$ otrā koordināta nav vesels skaitlis. Noskaidrosim, kāds izskatās Gomora šķēluma nosacījums, kurš jāpievieno simpleksa tabulai $T^{(4)}$.

$$\frac{3}{2} = x_2 + \frac{1}{2}s_1,$$

$$\left\{\frac{3}{2}\right\} \leq \left\{\frac{1}{2}\right\} s_1,$$

$$\frac{1}{2} \leq \frac{1}{2}s_1$$

izmantojot formulu (4.2.9), secinām, ka jāpievieno simpleksa tabulai šāda rinda

$$-\frac{1}{2} = -\frac{1}{2}s_1 + s_2.$$

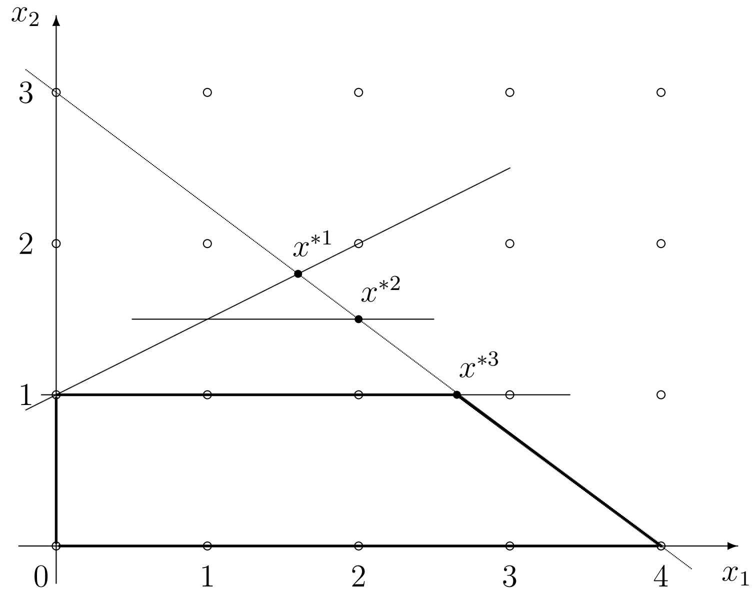
Jaunā tabula ir šāda

$$T^{(4)'} = \begin{array}{c|c|ccccccc} & b & a^1 & a^2 & v^1 & v^2 & s^1 & s^2 \\ \hline & 5 & 0 & 0 & \frac{1}{3} & 0 & \frac{1}{3} & 0 \\ \hline a^1 & 2 & 1 & 0 & \frac{1}{3} & 0 & -\frac{2}{3} & 0 \\ a^2 & \frac{3}{2} & 0 & 1 & 0 & 0 & \frac{1}{2} & 0 \\ v^2 & 1 & 0 & 0 & \frac{1}{3} & 1 & -\frac{5}{3} & 0 \\ s^2 & -\frac{1}{2} & 0 & 0 & 0 & 0 & -\mathbf{\frac{1}{2}} & 1 \end{array}$$

Ko šajā gadījumā mēs nogriezīsim nost no pieļaujamās atrisinājumu kopas?

$$\tfrac{1}{2} \leq \tfrac{1}{2}s_1 = \tfrac{1}{2}(-\tfrac{3}{5} + \tfrac{1}{5}v_1 + \tfrac{3}{5}v_2) =$$
$$= -\tfrac{3}{10} + \tfrac{1}{10}(12 - 3x_1 - 4x_2) + \tfrac{3}{10}(2 + x_1 - 2x_2) = \tfrac{3}{2} - x_2 \quad \Rightarrow x_2 \leq 1.$$

Zīmējumā tas izskatīsies šādi:



4.2.3. zīm.

Ar duālā simpleksa algoritmu tabulā $T^{(4)'}$ samainīsim s^2 ar s^1:

$T^{(5)} =$

	b	a^1	a^2	v^1	v^2	s^1	s^2
	$\frac{14}{3}$	0	0	$\frac{1}{3}$	0	0	$\frac{2}{3}$
a^1	$\frac{8}{3}$	1	0	$\frac{1}{3}$	0	0	$-\frac{4}{3}$
a^2	1	0	1	0	0	0	1
v^2	$\frac{8}{3}$	0	0	$\frac{1}{3}$	1	0	$-\frac{10}{3}$
s^1	1	0	0	0	0	1	-2

Iegūtajam atrisinājumam $x^{*3} = (\frac{8}{3}, 1$ pirmā koordināta nav vesels skaitlis. Atkal jānoskaidro, kāds izskatās Gomora šķēluma nosacījums, kurš jāpievieno simpleksa tabulai $T^{(5)}$.

$$\frac{8}{3} = x_1 + \frac{1}{3}v_1 - \frac{4}{3}s_2,$$
$$\left\{\frac{8}{3}\right\} \leq \left\{\frac{1}{3}\right\} v_1 + \left\{-\frac{4}{3}\right\} s_2,$$

$$\frac{2}{3} \leq \frac{1}{3}v_1 + \frac{2}{3}s_2$$

izmantojot formulu (4.2.9), secinām, ka jāpievieno simpleksa tabulai šāda rinda

$$-\frac{2}{3} = -\frac{1}{3}v_1 - \frac{2}{3}s_2 + s_3.$$

Jaunā tabula ir šāda

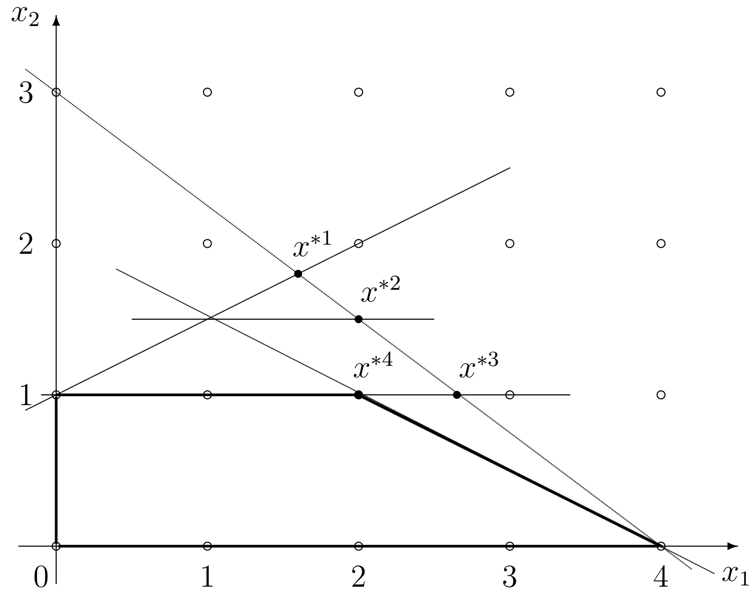
$T^{(5)'} =$

	b	a^1	a^2	v^1	v^2	s^1	s^2	s^3
	$\frac{14}{3}$	0	0	$\frac{1}{3}$	0	0	$\frac{2}{3}$	0
a^1	$\frac{8}{3}$	1	0	$\frac{1}{3}$	0	0	$-\frac{4}{3}$	0
a^2	1	0	1	0	0	0	1	0
v^2	$\frac{8}{3}$	0	0	$\frac{1}{3}$	1	0	$-\frac{10}{3}$	0
s^1	1	0	0	0	0	1	-2	0
s^3	$-\frac{2}{3}$	0	0	$-\mathbf{\frac{1}{3}}$	0	0	$-\frac{2}{3}$	1

Mēģināsim noskaidrot, ko šajā gadījumā nozīmē jaunais ierobežojums:

$$\begin{aligned} &\tfrac{2}{3} \leq \tfrac{1}{3}v_1 + \tfrac{2}{3}s_2 = \tfrac{1}{3}(12 - 3x_1 - 4x_2) + \tfrac{2}{3}(-\tfrac{1}{2} + \tfrac{1}{2}s_1) = \\ &= 4 - x_1 - \tfrac{4}{3}x_2 - \tfrac{1}{3} + \tfrac{1}{3}(3 - 2x_2) = \tfrac{14}{3} - x_1 - 2x_2 \quad \Rightarrow x_1 + 2x_2 \leq 4 \end{aligned}$$

Zīmējumā tas izskatīsies šādi:



4.2.4. zīm.

Tabulā $T^{(5)'}$ jāsamaina s^3 ar v^1, noslēguma tabula ir

$$T^{(6)} = \begin{array}{c|c|ccccccc} & b & a^1 & a^2 & v^1 & v^2 & s^1 & s^2 & s^3 \\ \hline & 4 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ \hline a^1 & 2 & & & & & & & \\ a^2 & 1 & & & & & & & \\ v^2 & 2 & & & & & & & \\ s^1 & 1 & & & & & & & \\ v^1 & 2 & & & & & & & \end{array}$$

Šajā tabulā neesam centušies aizpildīt matricas rindiņas, jo novērtējumu rindā un kolonnā visi skaitļi ir pozitīvi, kā arī atrastais atrisinājums $x^* = x^{*4} = (2,1)$ ir ar veselām koordinātām, mērķa funkcijas vērtība ir 4.

Labi apskatot 4.2.4.zīmējumu varam pamanīt, ka vērtību 4 mērķa funkcija sasniedz arī ar plānu $(4,0)$. Veicot pāreju no $T^{(5)'}$ uz $T^{(6)}$ mums bija iespēja izvēlēties, varējām mainīt arī s^3 ar s^1, pēc Blenda algoritma mainījām s^3 ar v^1. Lasītājs var pārbaudīt pats, vai, veicot citu maiņu, tiktu iegūts otrs atrisinājums. ■

NODAĻA 5

ALGORITMU SAREŽĢĪTĪBA

5.1 IZRĒĶINĀMĪBA UN LAIKA NOVĒRTĒJUMI

Šajā lekciju konspekta nodaļā tiks saīsināti atreferēti rezultāti un secinājumi no *C.H.Papadimitriou* grāmatas "Combinatorial optimization: Algorithms and Complexity" 5.nodaļas (izdevniecība Prentice-Hall, Inc., 1982, vai krievu valodā Maskava, izdevniecība Mir, 1985).

Simpleksa algoritma plašs lietojums ar simtiem un tūkstošiem mainīgo kļuva iespējams pateicoties skaitļojamajām mašīnām. Vai eksistē skaitļojamo mašīnu iespēju robeža? Acīmredzami, ka dators nevar izpildīt nekorekti formulētus nematemātiskus uzdevumus, piemēram, atrisināt enerģētikas problēmu vai pārspēt cilvēkus asprātībā. Skaitļojamā mašīna var izpildīt tikai *algoritmus*, t.i., precīzu un viennozīmīgi saprotamu komandu virkni, ar kuras palīdzību tiek atrisināti patvaļīgi stingri noteikti izskaitļojami uzdevumi. Tipisks algoritms, piemēram, ir metode, kā izpildīt aritmētiskās darbības ar veseliem skaitļiem decimālajā sistēmā. Tās ir precīzas metodes, kuras var lietot jebkuriem veseliem skaitļiem pēc patikas lieliem, un tās ir korektas metodes tādā nozīmē, ka garantē pareizas atbildes iegūšanu ar galīgu soļu skaitu. Šie soļi ir tik burtiski aprakstīti, ka tos var uzticēt mašīnai. Manipulācijas ar simboliem 1936.gadā aprakstīja angļu matemātiķis *A.M.Turing* un matemātiskais objekts tika nosaukts par *Tjūringa mašīnu*. Bet vai eksistē tādi matemātiski korekti uzdevumi, kuriem nav algoritma atrisinājumam? Tjūrings ir pierādījis, ka tādi eksistē. Tipisks piemērs ir apstāšanās problēma: noskaidrot konkrētai programmai un dotiem ieejas datiem, vai programma beigs darbu. Tjūrings ir pierādījis, ka neeksistē tāds algoritms, kurš spētu korekti atrisināt visus speciālgadījumus šādam uzdevumam.

Lineārās programmēšanas uzdevumi ir izrēķināmi; citiem vārdiem, eksistē tāds algoritms, kas korekti atrisina jebkuru šī uzdevuma speciālgadījumu. Tomēr ar to ne vienmēr pietiek, lai algoritmu uzskatītu par labu; var izrādīties, ka algoritma realizācijai ir nepieciešams tik daudz laika, ka praktiski algoritms ir nederīgs.

Piemērs 5.1.1. Komivojažiera uzdevums acīmredzami ir atrisināms, jo katru individuālu komivojažiera uzdevumu var atrisināt, atrodot labāko ceļu galīga skaita ceļu kopā. Tādējādi skaitļojamā mašīna var atrisināt jebkuru individuālu komivojažiera uzdevumu, sistemātiski apskatot visus ceļus, izskaitļojot visus to garumus un izvēloties starp tiem īsāko.

n pilsētu ceļu skaits ir $\frac{(n-1)!}{2}$, tāpēc aprakstītais algoritms skaitļojamajai mašīnai prasīs apmēram $n!$ soļus (elementārās komandas). Vidēja izmēra komivojažiera uzdevums (piemēram, ar 50 pilsētām) ar visu ceļu caurskatīšanas algoritmu prasītu ļoti daudz laika pat pie visoptimistiskākajām skaitļojamo mašīnu ātrumu prognozēm (50! ir skaitlis ar apmēram 65 zīmēm). ■

Viens no populārākajiem algoritmu darba kvalitātes mēriem ir laika intervāls, kāds ir nepieciešams līdz gala atbildes iegūšanai. Analizējot algoritmus, šajā nodaļā algoritmu darba laiks būs izteikts ar elementāro soļu (aritmētiskās operācijas, salīdzināšana, sazarošanās komanda, utt.) skaitu, kas nepieciešams algoritma izpildei uz jebkādas patvaļīgas skaitļojamās mašīnas; citiem vārdiem sakot, pieņemsim, ka elementārs solis prasa vienu laika vienību.

Soļu skaits, kas jāveic algoritmam, nav vienāds pat vienam uzdevuma tipam. Piemēram, simpleksa algoritma soļu skaits, lai atrisinātu lineārās programmēšanas uzdevumu

$$\max < c, x >$$
$$Ax \leq b, \quad x \geq 0,$$

var būtiski mainīties atkarībā no parametriem A, b un c pat ja to izmēri ir nemainīgi. Ekstremālā gadījumā, ja $c \leq 0$, sākotnējais atbalsta plāns ir optimāls un tāpēc nav jāveic nekādas bāzes vektoru maiņas, bet pie citām parametru vērtībām, lai sasniegtu optimālo plānu, var būt nepieciešams izpildīt nozīmīgu skaitu iterāciju.

Lai nogludinātu tādus izteiktus kontrastus algoritma uzvedībā, apskatīsim visus algoritma uzvedības iespējamos gadījumus kopā pie dotiem datiem, kuru izmērs ir n, un definēsim *algoritma sarežģītību* šim izmēram kā algoritma soļu skaitu pašā sliktākajā gadījumā. Tādā situācijā algoritma sarežģītība būs funkcija no datu izmēra n un būs, piemēram, šādā izskatā: $10n^3$, 2^n vai $n \log_2 n$.

Pētot algoritmu sarežģītību, bieži vien interesējas tikai par algoritma uzvedību, kad soļu skaits ir ļoti liels, jo tieši šie skaitļi nosaka robežas algoritma iespējām. Atšķirību starp algoritmu sarežģītību $10n^3$ un $9n^3$ var padarīt nebūtisku ar tehnoloģijas sasniegumu palīdzību, palielinot skaitļojamās mašīnas ātrumu desmitkārtīgi. No otras puses, lēni augoši saskaitāmie (tādi kā $5n$ novērtējumā $n \log_2 n + 5n$) tiks pilnībā dzēsti ar ātrāk augošiem saskaitāmajiem, ja n būs pietiekoši liels ($n \geq 1000$). Tādējādi mūs interesē algoritma sarežģītības augšanas ātrums.

Definīcija 5.1.1. Pieņemsim, ka $f(n)$ un $g(n)$ ir funkcijas, kas definētas naturālo skaitļu kopā un kuru vērtības ir pozitīvi reāli skaitļi.

a) Rakstīsim $f(n) = O(g(n))$, ja eksistē tāda konstante $c > 0$, ka $f(n) < cg(n)$ pietiekoši lieliem n.

b) Rakstīsim $f(n) = \Omega(g(n))$, ja eksistē tāda konstante $c > 0$, ka $f(n) \geq cg(n)$ pietiekoši lieliem n.

c) Rakstīsim $f(n) = \Theta(g(n))$, ja eksistē tādas konstantes c, $c' > 0$, ka $cg(n) \leq f(n) \leq c'g(n)$ pietiekoši lieliem n.

Attiecība Θ ir ekvivalences tipa attiecība. Ekvivalences klasi attiecībā pret šo attiecību, kas satur $f(n)$ (t.i., visu tādu funkciju $g(n)$ kopu, kuras $f(n) = \Theta(g(n))$), sauc par $f(n)$ *augšanas ātrumu*.

Izmantojot tikko izveidoto jēdzienu, algoritma sarežģītības augšanas ātrumu var novērtēt no augšas, izmantojot tādu izteicienu, kā, piemēram, "prasa laiku $O(n^3)$ ".

Mēs vēlamies izmērīt algoritma sarežģītību, ja dota ieejas datu funkcija. Varam pieņemt, ka algoritma sākumā ievadā ir simbolu virkne, kas nosaka ieejas izmēru. Definēsim ieejas izmēru kā šīs virknes garumu, t.i., simbolu skaitu šajā virknē.

Piemērs 5.1.2. Kāds ir lineārās programmēšanas uzdevuma izmērs?

Uzskatīsim, ka matricas A elementi un vektoru b un c koordinātas ir veseli skaitļi. Tāpēc par lineārās programmēšanas uzdevuma izmēru var uzskatīt simbolu skaitu, kas nepieciešams A, b un c pierakstīšanai. Tā kā to var izdarīt, pierakstot matricas elementus binārajā (vai decimālajā) sistēmā, izmantojot atbilstošus atdalītājus tabulas horizontālajām un vertikālajām līnijām, tad lineārās programmēšanas ar matricu $m \times n$ izmērs ir

$$\Theta(mn + \lceil \log_2 |P| \rceil),$$

kur ar $\lceil x \rceil$ tiek apzīmēts mazākais veselais skaitlis, kas lielāks vai vienāds ar reālo skaitli x, bet P ir visu nenulles koeficientu reizinājums.

Simpleksa algoritms ietver sevī sākuma soli un pēc tam vairākas iterācijas. Ja matricas A izmēri ir $m \times n$, tad sākuma solis prasa $O(mn)$ aritmētiskās operācijas. Analoģiski katru iterāciju var apskatīt kā matricas reizinājumu ar vektoru, t.i., to var izpildīt, veicot $O(mn)$ aritmētiskās operācijas. Ja iecikošanās ir novērsta, tad simpleksa algoritms pašā sliktākajā gadījumā var pārstaigāt visus atbalsta plānus, kuru skaits nepārsniedz C_m^{m+n}. Lai cik tas dīvaini arī neizklausītos, realitātē ar tik sliktiem uzdevumiem neiznāk saskarties, tomēr, veicot algoritmu sarežģītības novērtējumus, mēs rēķināmies ar teorētiski iespējamo pašu sliktāko gadījumu. ∎

5.2 POLINOMIĀLIE ALGORITMI

Skaitļojamo zinātņu speciālistu vidū šobrīd valda uzskats, ka algoritms ir praktiski derīgs (izskaitļojams) tikai tanī gadījumā, ja tā sarežģītība aug polinomiāli attiecībā pret ieejas izmēru. Piemēram, derīgi algoritmi ir ar sarežģītību $O(n)$ vai $O(n^3)$ (ievērojiet, ka polinomiālo ātruma pieaugumu pilnībā nosaka polinoma pakāpe!). Tāpat pieļaujami tādi algoritmi, kuru sarežģītība nav izsakāma ar polinomu, bet ir ierobežota ar polinomu, piemēram, $n^{2{,}5}$ vai $n \log_2 n$.

funkcija	aptuvenās vērtības		
n	10	100	1000
$n \log_2 n$	33	664	9966
n^3	1000	1 000 000	10^9
$10^6 n^8$	10^{14}	10^{22}	10^{30}
2^n	1024	1.27×10^{30}	1.05×10^{301}
$n^{\log_2 n}$	2099	1.93×10^{13}	7.89×10^{29}
$n!$	3 628 800	10^{158}	4×10^{2567}

5.2.1.tabula. Polinomiālo un eksponensiālo funkciju augšana.

Lai labāk saprastu polinomiāli ierobežoto algoritmu klases priekšrocības, apskatīsim tādus algoritmus, kuru sarežģītība pietiekoši lieliem n pārsniedz jebkuru polinomiālo novērtējumu. Parasti šādus algoritmus sauc par *eksponensiāliem*, jo 2^n ir viens no nepolinomiālo algoritmu ātruma mēra paraugiem. Citi piemēri eksponensiālā ātruma novērtējumam ir k^n (pie jebkura fiksēta $k > 1$), $n!$, 2^{n^2}, n^n un $n^{\log_2 n}$. Acīmredzami, ka reizē ar ieejas izmēra palielināšanos jebkurš polinomiālais algoritms kļūst efektīvāks par jebkuru eksponensiālo algoritmu, tas labi redzams 5.2.1.tabulā.

Cita pozitīva īpašība polinomiālajiem algoritmiem slēpjas tanī apstāklī , ka šie algoritmi kaut kādā nozīmē labāk izmanto tehnoloģiju iespējas. Piemēram, katru reizi, kad tehnoloģiskie sasniegumi ļauj palielināt skaitļojamo mašīnu ātrumu par 10 reizēm, lielākais uzdevuma izmērs, ko var izrēķināt ar polinomiālo algoritmu teiksim stundas laikā, tiek pareizināts ar konstanti starp 1 un 10, bet eksponensiālo algoritmu gadījumā uzdevumu izmēri pieaug tikai par aditīvu konstanti, kuru šis algoritms var izrēķināt fiksētā laika momentā (5.2.2.tabula).

funkcija	izmērs, ko var atrisināt vienā dienā	izmērs, ja ātrums palielinās 10 reizes
n	10^{12}	10^{13}
$n \log_2 n$	0.948×10^{11}	0.87×10^{12}
n^2	10^6	3.16×10^6
n^3	10^4	2.15×10^4
$10^8 n^4$	10	18
2^n	40	43
10^n	12	13
$n^{\log_2 n}$	79	95
$n!$	14	15

5.2.2.tabula. Polinomiālais algoritms labāk izmanto tehnoloģiju sasniegumus.

Un vēl viena polinomiālo algoritmu svarīga īpašība ir tā, ka tie ir "slēgti" algoritmi: drīkst kombinēt kopā polinomiālos algoritmus vienu ar otru, lai atrisinātu viena uzdevuma dažādus speciālgadījumus; viens polinomiālais algoritms var izsaukt citu kā apakšprogrammu; rezultātā kopumā viss algoritms paliks polinomiāls.

Lai noteiktu, kuri algoritmi ir polinomiāli un kuri eksponensiāli, īpaši uzmanīgiem jābūt tajā gadījumā, kad laika novērtējums ietver dotā uzdevuma skaitliskos ievaddatus.

Bet vai algoritms ar novērtējumu n^{80} būs praktiski risināms uzdevums? Ticami, ka nē, jo jau pie uzdevuma ar izmēru $n = 3$ laiks, kas nepieciešams uzdevuma atrisināšanai ar šādu polinomiālo algoritmu, ir ļoti liels un var izrādīties, ka kaut kāds eksponenciālais algoritms strādā daudz labāk pie visiem saprātīgiem ieejas izmēriem. Tāpēc teikt, ka polinomiālie algoritmi ir visās situācijās labāki par eksponenciālajiem, nav pareizi. Taču šim apgalvojumam pretī stājas pieredze. Ja uzdevumam tiek atrasts kaut kāds polinomiālais algoritms, tā uzreiz rodas iespējas samazināt polinoma pakāpi, jo dažādi pētnieki uzlabo algoritma ideju. Parasti noslēgumā tiek iegūts ātruma novērtējums $O(n^3)$ vai pat labāks. Eksponensiālie algoritmi pilnīgi pretēji praksē prasa tikpat daudz laika kā teorijā un no tiem cenšas atbrīvoties, tikko tiek atrasts uzdevumam polinomiālais algoritms.

Tomēr šie empīriskie noteikumi nav universiāli un dažos gadījumos var izrādīties pilnīgi aplami. Tālāk mēs apskatīsim simpleksa algoritmu un ne pārāk sen atklāto polinomiālo algoritmu lineārās programmēšanas uzdevumam. Rezultātā būs jāsecina, ka NEVAR teikt — polinomiāls algoritms un praktisks algoritms ir sinonīmi.

Apgalvojums 5.2.1. Simpleksa algoritms nav polinomiāls, tas ir eksponensiāls algoritms $O(2^d)$ (d — nezināmo skaits).

5.3 ELIPSOĪDA ALGORITMS

1979.gada pavasarī padomju matemātiķis L.G.Hačijans publicēja pierādījumu, ka ir tāds lineārās programmēšanas uzdevuma algoritms, kurš ir polinomiāls. Šis jautājums ilgu laiku bija atklāts. Hačijana rezultāts ir balstīts uz citu padomju matemātiķu darbiem par nelineāro programmēšanu un ir pilnīgi atšķirīgs no iepriekšējām lineārās programmēšanas uzdevumu analīzēm.

Atzīmēsim dažus svarīgākos secinājumus.

Definēsim uzdevumu par lineārām nevienādībām šādi: dotai $m \times n$ matricai A un m-dimensiju vektoram b jānoskaidro, vai eksistē tāds n-dimensiju vektors x, ka $Ax \leq b$.

Teorēma 5.3.1. Lineārās programmēšanas uzdevumam kanoniskajā formā eksistē polinomiāls algoritms tad un tikai tad, ja eksistē polinomiāls algoritms uzdevumam par lineārām nevienādībām.

Apskatīsim vēl vienu uzdevumu — uzdevumu par stingrajām lineārajām nevienādībām: dotai $m \times n$ matricai A un m-dimensiju vektoram b jānoskaidro, vai eksistē tāds n-dimensiju vektors x, ka $Ax < b$.

Sekas 5.3.1. Ja eksistē polinomiālais algoritms uzdevumam par stingrajām lineārajām nevienādībām, tad eksistē polinomiāls algoritms uzdevumam par lineārām nevienādībām.

Pamatideja elipsoīda algoritmam ir samērā vienkārša. Algoritms sastāv no iterācijām. Pieļaujamo plānu kopa tiek izveidota kā elipsoīds, kurš satur kaut kādu uzdevuma par stingrajām lineārajām nevienādībām atrisinājumu (ja tāds vispār eksistē). Iterācija sastāv no tekošā elipsoīda aizmainīšanas ar mazāku elipsoīdu, kurš satur uzdevuma par stingrajām lineārajām nevienādībām atrisinājumu. Pēc noteikta daudzuma iterācijām tiek atrasts atrisinājums vai tiek konstatēts, ka atrisinājums neeksistē.

Elipsoīda algoritms ir polinomiāls, tāpēc var apgalvot

Teorēma 5.3.2. Lineārās programmēšanas uzdevumam eksistē polinomiāls algoritms.

Diemžēl praktiski elipsoīda algoritms ir ļoti jūtīgs pret noapaļošanas kļūdām un tāpēc praksē netiek lietots. Tanī pašā laikā simpleksa algoritms strādā nevainojami praktiski visās situācijās. Un kā tas tika atzīmēts jau iepriekš, tad praksē ar pašiem sliktākajiem gadījumiem neiznāk saskarties, tāpēc paši sliktākie šī eksponensiālā algoritma novērtējumi netiek konstatēti.