KAUNO TECHNOLOGIJOS UNIVERSITETAS

FUNDAMENTALIŲJŲ MOKSLŲ FAKULTETAS

TAIKOMOSIOS MATEMATIKOS KATEDRA

Kristina Jusevičienė

Krovinių srautų modeliavimas uždarojoje logistikos sistemoje

Magistro darbas

Darbo vadovas

Doc. dr. G. Račkauskas

Kaunas, 2006

KAUNO TECHNOLOGIJOS UNIVERSITETAS

FUNDAMENTALIŲJŲ MOKSLŲ FAKULTETAS

TAIKOMOSIOS MATEMATIKOS KATEDRA

Kristina Jusevičienė

Krovinių srautų modeliavimas uždarojoje logistikos sistemoje

Magistro darbas

Kalbos konsultantė

Lietuvių k. katedros lekt.

J. Džezulskienė

2006-05

Vadovas

Doc. dr. G. Račkauskas

2006-05

Recenzentas

doc.

2006-05

Atliko

FMMM – 4 gr. stud.

Kristina Jusevičienė

2006-05-

Kaunas, 2006

TURINYS

LENTELIŲ SĄRAŠAS

PAVEIKSLŲ SĄRAŠAS

PRATARMĖ

Transportavimas – fizinis prekių judėjimas nuo žaliavų per gamybos procesą iki vartotojo yra didžiausių kaštų reikalaujanti veikla, kuri apima transportavimo būdų ir priemonių parinkimą, taip pat maršrutų sudarymą. Kadangi krovinių pervežimai logistikos sistemoje daugiausia lėšų reikalaujanti veikla, tai ypač stengiamasi rasti algoritmus, kuriais modeliuojami optimaliausi sprendimai (minimizuojant atstumą, laiką, pervežimo kainą).

Šiame darbe sprendžiamas transporto uždavinys su fiksuotu skirtingų talpų krovininių automobilių skaičiumi. Uždavinys aprašomas matematiniais apibrėžimais. Darbo tikslas modeliuoti krovinių srautus, naudojant euristinius, matematinius programavimo metodus, rasti optimalius sprendinius, ieškant trumpiausio kelio, ar mažiausios kainos. Daromos įvairios prielaidos ir apribojimai transporto uždavinio modeliui. Transporto uždavinys sprendžiamas keletu metodų: mažiausio elemento, Fogelio aproksimacijos ir šakų ribų. Tiriamojo darbo rezultatai buvo gauti naudojantis Excel programa ir C++ programavimo kalba.

ĮVADAS

Logistiką galima apibrėžti kaip strategiškai valdomą procesą, kurio metu žaliavos, dalys ir galutinai pagaminta produkcija yra sandėliuojamos ir pervežamos iš tiekėjų gavėjams.

Logistikos sistemą sudaro transportavimas, sandėliavimas ir saugojimas, atsargų įpakavimas bei priežiūra, atsargų kontrolė, užsakymų vykdymas, klientų aptarnavimas, gamyklų ir sandėlių išdėstymas, atsargų paskirstymas, pirkimas, grąžintų prekių valdymas, atliekų surinkimas ir panaudojimas Transportavimas užima svarbią vietą logistikos sistemoje. Transportavimas padeda sujungti gamintoją su vartotoju ir kartu įtraukti laiką ir vietą į prekę.

Dažnai tenka nagrinėti ir daryti sprendimus: kaip pervežti krovinius iš sandėliavimo vietos į paskirstymo punktus, kad bendros vežimo pajamos išlaidos būtų minimalios; kokia tvarka turi judėti įmonės vidaus transporto priemonės, kad pervežimų planus jos vykdytų mažiausiomis darbo laiko ir transporto įrenginių sąnaudomis; kaip apkrauti turimas transporto priemones, kad būtų gautas didžiausias darbo našumas. Tokių problemų yra daugybė. Ir parinkimas ekonomiškiausio, tikslingiausio, arba optimalaus varianto kiekvienai tokiai problemai spręsti gali duoti labai daug naudos.

Nuolatos didėjanti konkurencija, staigiai tobulėjančios informacinės technologijos, didėjanti globalizacija, didėjantis dėmesys kokybei ir besikaičiantys įmonių tarpusavio ryšiai privertė įmones praplėsti savo požiūrį į logistinį procesą t.y. dabar į šį procesą įjungiamos visos įmonės, dalyvaujančios kuriant produktą ir pristatant jį galutiniam vartotojui reikiamu laiku ir nepriekaištingos būklės.

Transporto problemų matematiniai modeliai taikomi siekiant kokybiškai įvertinti transporto darbą. Pagrindinės matematinių metodų transporte naudojimo sritys:

- Optimalių planų rengimas ir jų vykdymo kontrolė;
- Matematinio programavimo uždavinių sprendimas, planuojant transporto darbą:
  - optimalaus pervežimų plano sudarymas,
  - transporto išlaidų minimizavimas;
- Įvairiausių techninių – ekonominių uždavinių sprendimas:

teisingiausias ūkinių objektų išdėstymas,

optimalių gamybos atsargų planavimas sandėliuose ir jų paskirstymas.

Tiriamojo darbo tikslas susipažinti su transporto uždaviniu, jo problemomis, sprendimo būdais.

Transporto uždaviniai yra komivojažieriaus (TSP) ir m-TSP uždavinių apibendrinimas, teorinėje dalyje pateikiamos TSP ir m-TSP uždavinių formuluotės, bei transporto uždavinio sąryšis su šiais uždaviniais. Formuluojamas standartinis transporto uždavinys (apibrėžimas, jo apribojimai, matematinis šio uždavinio aprašymas (tikslo funkcijos minimizavimas, kiti ribojimai), ir darbe tiriamojo transporto uždavinys su fiksuoto skaičiaus skirtingos talpos krovininiais automobiliais.

Transporto uždaviniams spręsti naudojami specialūs transporto uždavinių metodai, kurie vadinami paskirstymo metodais.

Paskirstymo metodus patariama kombinuoti su euristiniais paskirstymo metodais. Daroma taip: problema sprendžiama kurio nors euristiniu metodu, ir gaunamas tariamai optimalus sprendinys. Paskui laikant šį tariamai optimalų sprendinį pradiniu, uždavinys toliau sprendžiamas kurio nors tikslinamuoju metodu tol, kol gaunamas tikrai optimalus sprendinys. Skyriuje 1.6 aptariami transporto uždavinio sprendinio ieškojimo metodai. Darbe su pavyzdžiais nagrinėjami metodai: mažiausio elemento, Fogelio aproksimacijos. Prie šios grupės priskiriama daug daugiau metodų, tačiau ir išvardytų pakanka, sėkmingai naudojant juos transporto problemoms spręsti. Mažiausio elemento metodas yra gana lengvas būdas rasti transporto uždavinio sprendimą, dažnai šiuo metodu sudarytas sprendinys nedaug skiriasi nuo optimalaus sprendinio. Darbe nagrinėjamas Fogelio aproksimacijos metodas, kai kuriais atvejais iš karto gali duoti optimalų sprendimą, šis metodas labai artimas mažiausio elemento metodui. Visais šiais metodais sprendžiamas, aiškinamas vienas ir tas pats uždavinys.

Šakų ribų metodu nagrinėtas pavyzdys yra m-TSP tipo uždavinys, tiesiog norėta parodyti šio metodo veikimą, bei rasti to uždavinio minimaliausią atstumą.

Labai svarbi dalis logistikos sistemoje yra atsargų planavimas, darbe modeliuojamos atsargos, tiriami krovinio pervežimo partijos optimalūs dydžiai, bei pervežimų skaičius.

Tiriamojoje dalyje sprendimais iliustruojami teorinėje dalyje minėti metodai, jie lyginami tarpusavyje. Aprašomas tiriamas transporto uždavinio modelis pagal matematines formuluotes. Šis modelis sprendžiamas euristiniu metodu, daromos įvairiausios prielaidos. Darbo tikslas rasti transporto uždavinio sprendimus, rasti pradinius transporto uždavinio sprendinius.

Priede pateikiami darbe naudoti duomenys ir gauti rezultatai, bei metodų, skirtų rasti pradinius sprendinius, programos tekstas.

1 BENDROJI DALIS

1.1 KOMIVOJAŽIERIAUS UŽDAVINYS

Keliaujančio prekeivio uždavinys (Komivojažieriaus uždavinys) (*TSP Traveling Salesperson Problem*) viena iš plačiausiai nagrinėjamų užduočių, sprendžiant optimizavimo uždavinius. Šių uždavinių prasmė yra tokia. Reikia apvažiuoti (aplankyti) *n* miestų. Išvykus iš pirmojo miesto, privalu užsukti į visus kitus miestus tik po vieną kartą taip, kad visas kelias, arba kelionės kaina, arba tos kelionės trukmė būtų minimalūs.

Tokio tipo uždaviniai gali būti aktualūs ir valdymo srityje. Komivojažieriaus vaidmuo gali tekti pareigūnui, gabenančiam svarbius dokumentus, medicinos būriams, renkantiems sužeistuosius, techninių specialistų brigadoms, turinčioms užduotį nustatyti kokios nors įrangos techninę būklę ir pan.

1.2 TRANSPORTO UŽDAVINIO SĄSAJA SU TSP ir M – TSP UŽDAVINIAIS

M – keliaujančių pirklių (M – TSP) uždavinys yra lengvai transponuojamas į klasikinį keliaujančio pirklio ar komivojažieriaus uždavinį. Be to M-TSP nėra sudėtingesnė problema nei TSP.

M- TSP uždavinys nuo TSP uždavinio skiriasi, tuo kad, iš vieno taško V (bazės) yra daromi keli maršrutai, šie maršrutai prasideda ir baigiasi viename ir tame pačiame taške. Maršrutų kelia i tarpusavyje nesikerta (išlaikoma TSP uždavinio sąlyga, kad kiekvienas miestas (terminalas) aplankomas tik vieną kartą ir tik vieną agento). Aplankyti reikia n miestų (terminalų). Analogiška formuluotė gaunama, kai keičiamas pradžios taškas V į m tikslių kopijų: $V_1, V_2, ..., V_m$ kiekvienas jų susietas su kitų taškų lygiai taip pat kaip ir tikrasis taškas ir su tuo pačiu atstumu. Tai yra, jei x yra vienas iš n aplankytų taškų (terminalų), tai

$$d(V_1, x) = d(V_2, x) = ... = d(V_m, x) = d(V, x)$$

Tačiau jungtys, siejančios m taškų, yra susietos neribotu ilgiu, t.y. labai dideliu atstumu lyginant su kitais atstumais sprendžiamame uždavinyje [d(V_i, V_j) = ∞ kiekvienam i, j = 1, 2, . . . , m].

Jei spręstume šį (m+n) – taškų TSP uždavinį, tai pastebėtume, kad minimalus maršrutas niekada nejungs dviejų originalo kopijų taškų. Tada, kai visos pradinio taško V kopijos sutraukiamos į originalų vieną tašką V, "vieno pirklio" maršrutas susiskaido į m maršrutų kaip ir reikalaujamam m-TSP.

Tiriamojoje dalyje pateiksiu 2 – TSP uždavinio pavyzdį.

Transporto uždaviniai (*The Vehicle Routing Problem (VRP)*) yra komivojažieriaus ir m-TSP uždavinių apibendrinimas, kur yra iš *m* nustatytų determinuotų kelių, čia kelias yra aplankymas, kai prasideda bazėje, aplankomos įvairios klientų vietos (tačiau tik vieną kartą) ir grįžtama bazę. Reikalavimas, kad pervežamų krovinių svoris neviršytų realios transporto priemonės keliamosios galios. Transporto uždavinio tikslas kaip ir komivojažieriaus uždavinio tikslas - minimizuoti viso paskirstymo kainas, kilometražą, kitas sąnaudas.

1.3 BENDRAS TRANSPORTO UŽDAVINIO FORMULAVIMAS

Transporto uždavinys bendru atveju formuluojamas taip: tegul turime *m* siuntimo (gamybos) punktų $K_1, K_2, ..., K_m$, kuriuose yra $k_1, k_2, ..., k_m$ vienetų vienalyčio krovinio (produkto). Tiekėjais gali būti įmonės, gaminančios kitoms įmonėms produkciją, sandėliai, bazės. Krovinys turi būti pristatytas į n vartojimo (gavėjų) punktus $D_1, D_2, ..., D_n$, kuriems reikia atitinkamai $d_1, d_2, ..., d_n$ vienetų krovinio (produkto). Žinoma vieneto krovinio pervežimo kaina iš siuntimo punkto K_i į paskirties (vartojimo) punktą D_j. Ją žymėsime C_{ij}. Iš tikrųjų, c_{ij} – tikslo funkcijos vertės koeficientas; jis, priklausomai nuo pasirinkti kriterijaus, gali reikšti pervežimo išlaidas, sugaištamą laiką ir kitus transportavimo parametrus.

Reikia sudaryti tokį pervežimo planą, kad visi kroviniai būtų išvežti, gavėjų poreikiai patenkinti ir bendros pervežimo išlaidos būtų minimalios.

Transporto uždaviniuose tenka taip pat ieškoti ir maksimumo: sudaryti tokį pervežimo planą, kad būtų gautas maksimalus pelnas, pervežant visus numatytus krovinius; kad būtų maksimaliai išnaudota turimų transporto priemonių keliamoji galia.

1.4 MATEMATINIS STANDARTINIS TRANSPORTO UŽDAVINIO MODELIS

Pažymėsime X_{ij} krovinių kiekį, kurį reikia pervežti iš punkto K_i į punktą D_j. Tada ieškomąjį planą galima išreikšti vadinamąja **pervežimų matrica**

$$X = \begin{pmatrix} X_{11} & X_{12} & \dots & X_{1n} \\ X_{21} & X_{22} & \dots & X_{2n} \\ --- & --- & --- & --- \\ X_{m1} & X_{m2} & \dots & X_{mn} \end{pmatrix} = \left(X_{ij} \right)_{m \times n}$$

.

Žinomas pervežimų kainas irgi galima išreikšti matrica.

Duotieji dydžiai C_{ij} ir ieškomi dydžiai X_{ij} užrašomi vienoje lentelėje, kuri vadinama paskirstymo lentele. Pasiskirstymo lentelės pavyzdys (Lentelė Nr.1).

Lentelė Nr.1 .1

Pasiskirstymo lentelė

Tiekėjai	Gavėjai						Atsargos
	D_1	D_2	...	D_j	...	D_n	
K_1	C_{11} X_{11}	C_{12} X_{12}	... ...	C_{1j} X_{1j}	... ...	C_{1n} X_{1n}	K_1
...	...	...	...	...	...	...	...
K_i	C_{i1} X_{i1}	C_{i2} X_{i2}	... ...	C_{ij} X_{ij}	... ...	C_{in} X_{in}	k_i
...	...	...	...	...	...	...	...

K_m	C_{m1} X_{m1}	C_{m2} X_{m2}	... ...	C_{mj} X_{mj}	... ...	C_{mn} X_{mn}	k_m
Poreikiai	d_1	d_2	...	d_j	...	d_n	

Išlaidos, susijusios su krovinio kiekio X_{ij} pervežimu bus $C_{ij} \cdot X_{ij}$ ir bendra pervežimo kaina, minimali tikslo funkcija:

$$\min F(x) = \sum_{i=1}^{m} \sum_{j=1}^{n} C_{ij} X_{ij} \quad . \tag{1.1}$$

Ieškomi dydžiai turi tenkinti tokias sąlygas:

1) visi kroviniai iš siuntimo punktų turi būti išvežti, t.y.

$$\left\{ \sum_{j=1}^{n} X_{ij} = k_i, i = 1,2,...,m \right. , \tag{1.2}$$

2) visų gavimo punktų poreikiai turi būti tenkinami, t.y.

$$\left\{ \sum_{i=1}^{m} X_{ij} = d_j, \; j = 1,2,...,n \right. , \tag{1.3}$$

3) kintamųjų neneigiamumo sąlyga $X_{ij} \geq 0$, kai $i = 1..m,$, $j = 1..n$. (1.4)

Matematiškai transporto uždavinys formuluojamas: rasti nežinomųjų X_{ij} reikšmes, tenkinančias (1.2)-(1.4) sąlygas ir minimizuojančias (1.1) tikslo funkciją.

Transporto uždavinio paskirstymo lentelės matricos rangas lygus $m+n-1$.

Taip pateiktas transporto uždavinio matematinis modelis yra standartiniame pavidale.

Tai kanoninio pavidalo teisinio programavimo uždavinys, turintis $m \times n$ nežinomųjų ir $m+n$ apribojimų lygybių. Sprendinys tenkinantis, (1.2)-(1.4) sąlygas, vadinamas **leistinuoju sprendiniu** (planu).

Teorema. Transporto uždavinys turi optimalų pervežimo planą tada ir tik tada, kai tenkinama vadinamoji balanso (modelio uždarumo) sąlyga

$$\sum_{i=1}^{m} k_i = \sum_{j=1}^{n} d_j, \text{ kai } i = 1..m,, \; j = 1..n,$$

Jei, $\sum_{i=1}^{m} k_i \neq \sum_{j=1}^{n} d_j$, tai transporto uždavinys vadinamas **nesubalansuotu** (atviruoju).

Norint išspręsti transporto uždavinius su atviru balansu, reikalinga juos suvesti į uždarą, įvedant papildomą tiekėją (eilutę) arba vartotoją (stulpelį).

1.5 MATEMATINIS TRANSPORTO UŽDAVINIO MODELIS SU SKIRTINGO TIPO KROVININIAIS AUTOMOBILIAIS

Transporto uždavinių sprendimas su skirtingo tipo automobiliais apima tokius klausimus: kurio tipo automobilį pasirinkti, kiek kokių automobilių reikia, kuriam gavėjui kokį automobilį priskirti, kokia tvarka parinkti automobiliai turi aplankyti gavėjus.

Turim grafą G=(V,E), kur V - viršūnių taškų aibė, E - viršūnes jungiančių lankų aibė. Pasižymėsim $v_0 \in V$ bazės (tiekėjo taškas) viršūnė, visos viršūnės $v \in V \setminus \{v_0\}$ atitinka gavėjus, kiekvienas lankas $e \in E$ žymi ryšius tarp viršūnių. Duota aibė $S \subseteq V$, pasižymim $E(S)$ – lankų aibė, kurių abu galai - viršūnės priklauso aibei S, o $\delta(S)$ – aibė lankų, kurių vienas galas - viršūnė priklauso aibei S, o kitas $V \setminus S$. Kai $S = \{v\}$ rašome $\delta\{v\}$. Kelionės kaina, keliaujant lanku $e \in E$ su automobiliu k yra $c_{ek} > 0$. Čia laikomasi trikampio nelygybės taisyklės: kiekvienas lankas $e \in E \setminus \delta(v_0)$ optimaliame sprendinyje pereinamas vieną kartą. Gavėjų pareikalavimai susieti su viršūnėmis $v \in V \setminus \{v_0\}$ lankais d_v. Pasižymėkim $d(S) = \sum_{v \in S} d_v$. Raide K pasižymim aibę krovininių automobilių, o k-ojo, kai $k \in K$ automobilio talpą žymėsim C_k.

Apibrėžiam kintamuosius:

$$y_{vk} = \begin{cases} 1, & \text{kai gavėjas } v \text{ yra aptarnaujamas automobiliu } k, \\ 0, & \text{kitu atveju}, \quad v \in V,\ k \in K; \end{cases}$$

$$x_{ek} = \begin{cases} 1, & \text{kai lankas e yra naudojamas automobilio k}, \\ 0, & \text{kitu atveju}, \quad e \in E, k \in K; \end{cases}$$

$$z_{vk} = \begin{cases} 1, & \text{kai automobilis k tiesiogiai grįžta iš gavėjo į bazę}, \\ 0, & \text{kitu atveju}, \quad v \in V \setminus \{v_0\}, k \in K.; \end{cases}$$

Nusistatome:

$$\min \sum_{k \in K} \sum_{e \in E} c_{ek} x_{ek} + \sum_{k \in K} \sum_{v \in V \setminus \{v_0\}} c_{(v,v_0),k} z_{vk} \text{, kur}$$

$$\sum_{k\in K} y_{vk} = 1 \qquad v \in V \setminus \{v_0\} \tag{1.5}$$

$$\sum_{e\in\delta(v_0)} x_{ek} = y_{v_0 k}, \qquad k \in K, \tag{1.6}$$

$$z_{vk} + \sum_{e\in\delta(v)} x_{ek} = 2y_{v_0 k}, \quad v \in V \setminus \{v_0\},\ k \in K, \tag{1.7}$$

$$\sum_{v\in V\setminus\{v_0\}} d_v y_{vk} \le C_k \qquad k \in K, \tag{1.8}$$

$$y_{v_0,k} = 1 \qquad k \in K, \tag{1.9}$$

$$y_{vk} \in \{0,1\} \qquad v \in V \quad k \in K, \tag{1.10}$$

$$(x, z) \in \Omega \tag{1.11}$$

Formulių paaiškinimas:
Formulė (1.5), kiekvienas klientas turi būti aptarnautas būtent vieno krovininio automobilio.

Formulės (1.6) ir (1.9) kartu reiškia, kad kiekvienai k transporto priemonei, būtent vienas x_{ek} kintamasis susietas su lanku. Formulė (1.7) yra viršūnės laipsnių apribojimas visiems gavėjams ir visoms transporto priemonėms. Automobilio talpos apribojimai nurodomi formule (1.8).

Šis metodas tinka tik tiems atvejams kai klientai aplankomi tik vieną kartą.
Aibė Ω :

$$\sum_{k\in K} x_{ek} \le 1, \qquad e \in E \tag{1.12}$$

$$\sum_{k\in K} \sum_{e\in\delta(v_0)} x_{ek} = |K|, \tag{1.13}$$

$$\sum_{k\in K} \sum_{e\in E} x_{ek} = |V| - 1, \tag{1.14}$$

$$\sum_{k\in K} \sum_{e\in E(S)} x_{ek} \le |S| - 1, \qquad S \subseteq V \setminus \{v_0\} \tag{1.15}$$

$$x_{ek} \in \{0,1\}, \qquad e \in E \tag{1.16}$$

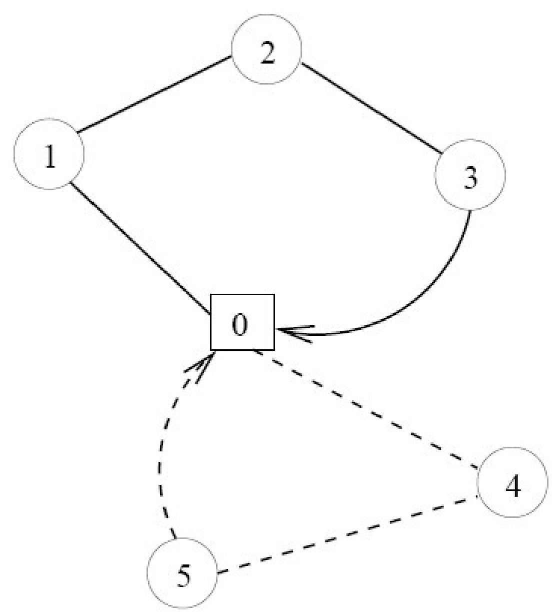
$$\sum_{v\in V\setminus\{v_0\}} z_{vk} = 1, \qquad k \in K, \tag{1.17}$$

$$\sum_{k\in K} z_{vk} \le 1, \qquad v \in V \setminus \{v_0\}, \tag{1.18}$$

$$z_{vk} \in \{0,1\}, \qquad v \in V \setminus \{v_0\}, \qquad k \in K. \tag{1.19}$$

Formulės (1.12-1.16) modeliuoja medį (spanning tree) su viršūnių laipsnių apribojimais. Formule (1.12) teigiama, kad kiekvienas lankas yra naudojamas tik vieno automobilio, (1.13) – gavėjų taškų apribojimai, (1.14)- pagrindinis, svarbiausias suvaržymas. Formule (1.15) daromas užtikrinimas, kad nėra ratų tarp gavėjų aibėje $S \subseteq V \setminus \{v_0\}$. Formule (1.17) teigiama, kad kiekviena transporto priemonė naudoja vieną grįžimo lanką (kelią). (1.18)- kiekvienas gavėjas v naudoja vieną grįžimo lanką.

Pavyzdyje duotas galimas sprendimas, kur $V = \{v_0, v_1, v_2, v_3, v_4, v_5\}$, ir $K = \{k_1, k_2\}$. Vientisa linija rodo transporto priemonės k_1 kelią, o taškuota – transporto priemonės k_2. Grįžimo lankai pažymėti linijomis su rodyklėmis.



Šiame pavyzdyje yra tokie kintamieji:

$y_{v_0,k_1}, y_{v_1,k_1}, y_{v_2,k_1}, y_{v_3,k_1}, y_{v_0,k_2}, y_{v_4,k_2}, y_{v_5,k_2}$;

$x_{(v_0,v_1),k_1}, x_{(v_1,v_2),k_1}, x_{(v_2,v_3),k_1}, x_{(v_0,v_4),k_2}, x_{(v_4,v_5),k_2}$;

z_{v_3,k_1} ir z_{v_5,k_2}. Visi kiti kintamieji yra lygūs nuliui.

Leistinos nelygybės.

Pirmiausia nagrinėsim atvejį, kai automobiliai yra homogeniniai, su talpa C, kintamieji x_e apibūdina laiko lankos skaičių naudojamą sprendime. Duota gavėjų aibė $S \subseteq V \setminus \{v_0\}$, Krovininių automobilių, kurie važiuoja šia aibe, bendra talpa turi stipriai viršyti bendrą gavėjų pareikalavimą, be to kiekviena mašina įeinanti į aibę, turi iš jos išeiti.

$$C\sum_{e\in\delta(S)} x_e \geq 2d(S), S \subseteq V \setminus \{v_0\}.$$

Šią formulę galim perkonstruoti į formulę

$$\sum_{e\in\delta(S)} x_e \geq 2\left\lceil \frac{d(S)}{C} \right\rceil, S \subseteq V \setminus \{v_0\}.$$

Panašiai konstruojame ir heterogeniniams automobiliams

$$\sum_{k\in K} C_k \left(\sum_{e\in\delta(S)} x_e + \sum_{v\in S} z_{vk} \right) \geq 2d(S), S \subseteq V \setminus \{v_0\}. \tag{1.20}$$

turim neneigiamą skaičių ϕ_S, duotai aibei $S \subseteq V \setminus \{v_0\}$, perrašom (5) formulę

$$\sum_{k\in K} \lceil \phi_S C_k \rceil \left(\sum_{e\in\delta(S)} x_e + \sum_{v\in S} z_{vk} \right) \geq 2\lceil \phi_S d(S) \rceil, S \subseteq V \setminus \{v_0\}.$$

Suprantama, kad tik dalis skaičių ϕ yra svarbūs. Šių aibę žymėsime raide Γ. Pasirenkame $\phi = \frac{1}{C}$.

1.6 SPECIALŪS TRANSPORTO UŽDAVINIŲ SPRENDIMO METODAI

Transporto uždavinio sprendimo algoritmas susideda iš dviejų etapų:

1) pradinio leistinojo bazinio plano sudarymo ;

2) nuoseklaus artėjimo prie optimalaus plano.

Panagrinėkime pirmą sprendimo etapą.

Imame transporto uždavinį, kurio paskirstymo lentelėje yra m eilučių (tiekėjų) ir n stulpelių (vartotojų). Transporto uždaviniuose pradinis planas randamas šiaurės vakarų kampo, minimalaus elemento, Fogelio aproksimacijos ir kitais metodais.

Atlikus pradinį paskirstymą pagal vieną iš minėtų taisyklių, reikia patikrinti paskirstymo teisingumą. Jeigu išrašytume apribojimų sistemos lygčių koeficientus ir nustatytume, kam lygus gautos matricos rangas, tai gautume, kad prie bet kurių m ir n reikšmių matricos rangas $r=m+n-1$. Reiškia, kad transporto uždavinių apribojimų sistemoje iš ($m+n$) lygčių nepriklausomos yra tik tai $m+n-1$. Todėl transporto uždavinyje nelygūs 0 (nuliui) gali būti ne daugiau kaip $m+n-1$ komponenčių (langelių). Vadinasi, atlikus paskirstymą užpildytų langelių skaičius $K_{pask}= m+n-1$,

kur m – tiekėjų (eilučių) skaičius,

n – vartotojų (stulpelių) skaičius.

Jeigu atlikus paskirstymą, faktinis užpildytų langelių skaičius $K_{fakt}=K_{pask}$, tai paskirstymas atliktas teisingai. Jeigu $K_{pask} \neq K_{fakt}$ - turime taip vadinamą išsigimimo atvejį. Dažnai būna, kad $K_{pask}<K_{fakt}$. Tokiu atveju reikalinga arba pradinį planą sudaryti pagal kitą metodą, arba vieną laisvą langelį užimti kroviniu lygiu 0 (nuliui) ir skaityti kaip normaliai užpildytu langeliu.

1.6.1 MAŽIAUSIO ELEMENTO METODAS

Taikant šį metodą, pirmiausia randame sprendinio kintamąjį su mažiausia gabenimo kaina (C_{ij}) (gali būti atstumas). Tada į šią celę perskirstome kiek galima daugiau prekių, kurių kiekis yra K_i ir G_i (gavėjas) minimumas. Po to, kaip ir taikant Šiaurės vakarų kampo metodą, einame per *i-tąją* eilutę ir *j-tąjį* stulpelį ir mažiname pasiūlą ir paklausą tarpre susikertančių eilučių ir stulpelių.

1.6.2 FOGELIO APROKSIMACIJOS METODAS

Šio metodo algoritmas yra toks:

1. Kiekvienoje eilutėje ir kiekviename stulpelyje randamas dviejų mažiausių tarifų skirtumas. Jis užrašomas papildomame stulpelyje ar eilutėje. Didžiausias iš jų apibraukiamas.

2. Eilutėje ar stulpelyje, kur tas didžiausias skirtumas, užpildomas langelis su mažiausiu tarifu. Eilutė ar stulpelis, kuriame nebeliek a išteklių ar poreikių, toliau nebeimamas.

3. Spręsti pradedama nuo pirmojo punkto su likusia lentele. Gautas planas yra artimas optimaliam.

1.6.3 ŠAKŲ IR RIBŲ METODAS

Metodo esmė ta, kad vienas iš galimų kiekvieno uždavinio sprendimo metodų yra nuoseklus visų galimų sprendimų peržiūrėjimas ir jų palyginimas pagal pasirinktą efektyvumo kriterijų, siekiant atrinkti optimalų.

Sprendžiant prekybos agento uždavinį šiuo metodu, sudaroma visų galimų sprendinių aibė, t.y. visi galimi maršrutai nuosekliai dalijami į mažėjančius poaibius, kiekvieną kartą nustatant trumpiausią maršrutą. Taip dalijant gaunamas poaibis, turintis tik vieną maršrutą neilgesnį už bet kurį kitą maršrutą. Savaime suprantama, kad poaibiai dalijami taip, kad toliau nagrinėti būtų paliekami perspektyvūs sprendiniai, o neperspektyvūs - atmetami.

Šakų ir ribų metodu skaičiuojama etapais:

- sudaroma atstumų matrica;

- formuojamas medis;
- įvertinamos viršūnės;
- uždraudžiami neperspektyvūs keliai;
- atrenkamas optimalus maršrutas.

- **Atstumų matricos sudarymas**

Atstumų matricą $C = \left|\left(C_{ij}\right)\right|, i, j = \overline{1,N}$ sudaro elementai C_{ij} nusakantys atstumus tarp atskirų objektų.

	1	2	...	*j*	...	*N*
1	∞	C_{12}	...	C_{1j}	...	C_{1N}
2	C_{21}	∞	...	C_{2j}	...	C_{2N}
...	...	...	∞	...	...	...
i	C_{i1}	C_{i2}	...	∞	...	C_{iN}
...	...	...	...	....	∞	...
N	C_{N1}	C_{N2}	...	C_{Nj}	...	∞

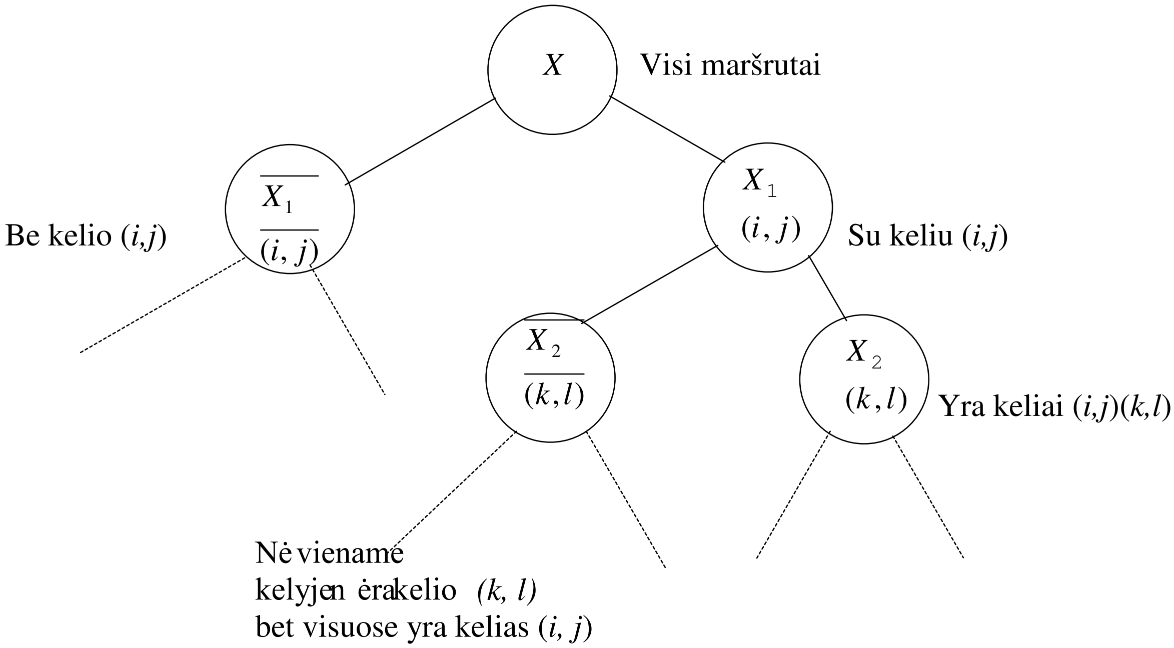
Ši matrica pasižymi šiomis savybėmis:

1. Degonaliniai elementai $C_{11} = C_{22} = ... = C_{NN} = \infty, i, j = \overline{1,N}$. Tuo draudžiama grįžti į tą patį objektą.

2. Bendru atveju $C_{ij} \neq C_{ji}$, nes gali egzistuoti įvairūs galimi keliai.

- **Medžio formavimas**

Jeigu X pažymėsime visų galimų maršrutų aibę, tai šios aibės skaidymą į nepersikertančius poaibius X_1 ir $\overline{X_1}$ galima vaizduoti kaip medžio šakojimąsi. Medžio viršūnė (*i, j*) atitinka maršrutų poaibį X_1, kurioje yra kelias iš objekto i į objektą j, o $\overline{X_1}$ - maršruto poaibis, kuriame nėra maršruto iš i -ojo objekto į j -ąjį objektą. Toliau šakojantis X_1, gaunami du nauji poaibiai X_2 ir $\overline{X_2}$, o kartu ir dvi naujos medžio viršūnės. Taip skaidant poaibius, gaunamas medis, kurio kabančios viršūnės atitinka tik vieną maršrutą. Medžio formavimo schema atrodo taip:



1.1 Pav. Medžio formavimo schema

Toliau aiškinama, kaip gautus maršrutus skirstyti į perspektyvius ir neperspektyvius.

- **Viršūnių įvertinimas**

Šakojimosi metu skaičiuojamas mažiausias kiekvienos maršrutų aibės ilgis, t.y. bet kuris kitas tos aibės maršrutas yra ne trumpesnis už apskaičiuotą trumpiausią maršrutą. Viršūnių įvertinimas remiasi teiginiu, kad išsprendus prekybos agento uždavinį ir nustačius objektų apvažiavimo seką ši seka nepasikeis, jei iš atstumų matricos bet kurios eilutės ar stulpelio elementų atimsime bet kurį teigiamą skaičių, o šiuo dydžiu pasikeis tik maršruto ilgis. Remdamiesi šiuo teiginiu perskaičiuojame atstumų matricą: iš kiekvienos eilutės elemento atimamas jos minimalus elementas h_i , ir po to iš kiekvieno stulpelio elemento atimamas jos minimalus elementas h_j . Taip perskaičiuota atstumų matrica turi mažiausiai po vieną nulį kiekvienoje eilutėje ir kiekviename stulpelyje. Perskaičiuotos matricos maršruto ilgis L_1 skiriasi nuo pradinio maršruto ilgio L dydžiu H lygiu perskaičiavimo koeficientų h_i ir h_j sumai:

$$L = L_1 + H,$$

$$H=\sum_{i=1}^{N} h_i + \sum_{j=1}^{N} h_j$$

$$H=\sum_{i=1}^{N} h_i + \sum_{j=1}^{N} h_j$$

Kadangi perskaičiuotos matricos elementai yra teigiami skaičiai, tai $L_1 \geq 0$ ir dydis H vadinamas mažiausios viršūnės reikšme.

- **Neperspektyvių kelių draudimas** (Draudimas šakotis)

Įtraukus į formuojamą maršrutą kelią (i, j), galima nenagrinėti visų kitų kelių, išeinančių iš i-ojo objekto, ir visų kitų kelių, ateinančių į j-ąjį objektą. Tada galima sumažinti atstumų matricą, išbraukiant i-ąją eilutę ir j-ąjį stulpelį. Mažinant atstumų matricą, reikia uždrausti ir tokius kelius, kurie sudarytų ciklus, tai reikštų grįžimą į tą patį objektą, nenuvažiavus į kitus. Todėl perrašant atstumų matricą, elementą C_{ji} reikia prilyginti ∞, t.y.

$C_{ji} = \infty$.

Išbraukus eilutę ir stulpelį bei uždraudus negalimus maršrutus, gaunamas naujas prekybos agento uždavinys su vienetu mažesniu objektų skaičiumi. Šis mažinimo procesas tęsiamas tol, kol atstumų matrica bus 2x2, tai yra gaunama viršūnė su vieninteliu maršrutu. Tačiau lieka neaišku, kurią viršūnę, (i,j) pasirinkti medžio šakojimuisi. Reikia pasirinkti tokį maršrutą, kuris turėtų didžiausią "baudą" už jo nepanaudojimą.

Tam tikslui, kiekvienam matricos elementui $C_{rs} = 0$ apskaičiuojama bauda

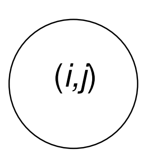
$$h_{rs} = \min_{j \neq s}\{C_{rs}\} + \min_{i \neq r}\{C_{i,s}\}$$

Dydis h_{rs} rodo, kiek mažiausiai pralošime jei maršrute nebus kelio C_{rs}. Apskaičiuojama maksimali bauda:

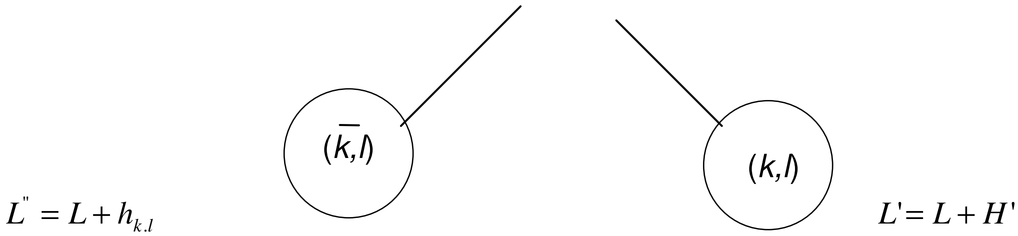
$$h_{kl}=max\{h_{rs}\}.$$

Indeksai (k, l) kaip tik ir parodo medžio šakojimosi viršūnę, o kartu ir atstumų matricos redukavimo elementą.

Sumažinus matricą, t.y. išbraukus k-ąją eilutę ir l-ąjį stulpelį, apskaičiuojama H'. žinant H' ir h_{kl} galima nurodyti kaip skaičiuojamos viršūnės. Medžio viršūnių skaičiavimo schema (2 pav.)



$L=H$



1.2. Pav. Medžio viršūnių skaičiavimo schema

- **Optimalaus maršruto atrinkimas**

Sudarius vieną maršrutą, t.y. egzistuojant keliui iš medžio šaknies į kabančią viršūnę, sužinomas jo ilgis M_1. Iškyla klausimas, ar šis sprendinys yra optimalus?

Kadangi toliau dalijant maršrutų aibę gali tik padidėti mažiausia reikšmė, tai šakojimą tikslinga tęsti tik iki tos viršūnės, kurios mažiausia reikšmė yra mažesnė už M_1. Jei tokios viršūnės nėra, tai maršrutas M_1 yra optimalus. Priešingu atveju, jei tokia viršūnė egzistuoja, tai iš jos šakojant, randamas maršrutas M_2. Jei šis maršrutas yra trumpesnis už M_1, tai maršrutas M_2 tampa tolesnių skaičiavimų pamatu - etalonu. Jei visų sekančių medžio viršūnių mažiausios reikšmės yra ne mažesnės už nustatyto etaloninio maršruto ilgį, tai skaičiavimai baigti ir etaloninis maršrutas laikomas optimaliu.

Ieškant naujo maršruto M_2 reikia atkurti matricą. Tam reikia nustatyti, iš kurios viršūnės bus šakojamas medis. Todėl sprendimo medyje randama viršūnė, kurios mažiausia reikšmė yra mažesnė už apskaičiuoto maršruto ilgį. Jei yra kelios tokios viršūnės, imama mažiausia.

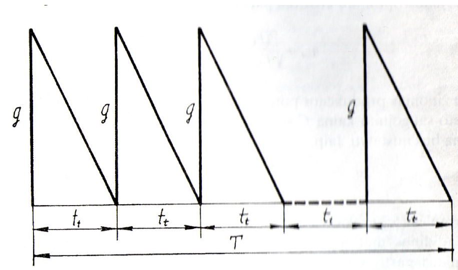
1.7 ATSARGŲ VALDYMAS VEŽIMO PARTIJŲ OPTIMIZAVIMAS

Nuostolingos yra per daug didelės ir per daug nepakankamos atsargos. Didelės atsargos „užšaldo“ lėšas, panaudotas nevartojamai produkcijai įsigyti ir saugoti.

Atsargos pildomos periodiškai. Kiekvieną papildymą lydi atitinkamos išlaidos. Sandėliai turi išlaidas susijusias su produkcijos saugojimu. Jeigu pareikalavimas krovinių vežimui greitai neišpildomas, tai vartotojas turi nuostolių. Atsargų valdymo uždavinys – nustatyti atsargų papildymo apimtį ir momentą, su minimaliomis saugojimo, fondų ir tiekimo išlaidomis.

Atsargų valdymo uždaviniai skirstomi į statinius (vienetinis atsargų sudarymo aktas) ir dinaminius (atsargų naudojimas ir periodinis jų pildymas, procesui besivystant).

Paprasčiausias atsargų valdymo modelis – sistema su fiksuotu užsakymo dydžiu. Kai pakartotinas užsakymas pateikiamas sumažėjus atsargoms iki tam tikro kritinio dydžio.



1.3. Pav. Paprasčiausias atsargų valdymo modelis

Galimos šios pagrindinės atsargų valdymo strategijos.

- Pastovaus dydžio užsakymo strategija. Ji numato nepertraukiamą atsargų būklės kontrolę. Užsakymo momentai susidaro nelygiais laiko tarpais.
- Pastovaus periodiškumo užsakymų strategija. Jai charakteringas užsakymų pasikartojimas lygiais laiko tarpais, o užsakymas gaunamas kintamo dydžio.
- Atsargų papildymo iki pastovaus lygio nustatytu periodiškumu strategija. Ji numato kintamo dydžio užsakymo vykdymą vienodai laiko tarpais.

Tokiu būdu vežamų partijų dydžiai q ir intervalai tarp eilinių siuntų gali būti pastovūs ar kintami, priklausomai nuo priimtos atsargų valdymo strategijos.

Naudojantis faktiškais duomenimis apie realizavimo apimtis ir užsakymo pristatymo trukmę, galima procesą modeliuoti ir nustatyti deficito atsiradimo tikimybę bei vidutinį atsargų lygį naudojant atitinkamą krovinių pristatymo sistemą per ilgą laiko tarpą. Paprasčiausia atsargų dydį nustatyti statistikos uždavinyje, kada atsitiktinis dydis yra produkcijos pareikalavimas periodu tarp dviejų eilinių siuntų.

Tikimybė, kad atsargų dydis *R* bus nepakankamas padengti skirtumų tarp eilinės partijos siuntos dydžio q ir produkcijos pareikalavimo tarp dviejų eilinių siuntų G bus lygi

$$P = P\{G - q > R\}$$

Norint nustatyti rezervą *R,* būtina išaiškinti pareikalavimo G (atsitiktinio dydžio) pasiskirstymo charakterį tarp dviejų tarpinių siuntu. Jeigu G pasiskirsčiusi pagal normalųjį dėsnį, tai žinant jos vidutinį kvadratini nukrypimą G ir tikimybės *P* reikšmę, galima surasti optimalų atsargų dydį. Be deficitinio tiekimo tikimybė turi būti lygi *I-P. o* atsargos dydis

$$R = t_{1-n}\sigma$$

čia t_{1-p} - normaliojo pasiskirstymo integralinės funkcijos standartizuoto nukrypimo kaitmeninė reikšmė, atitinkanti leidžiamą tikimybę $\beta_1 = 1 - p$.

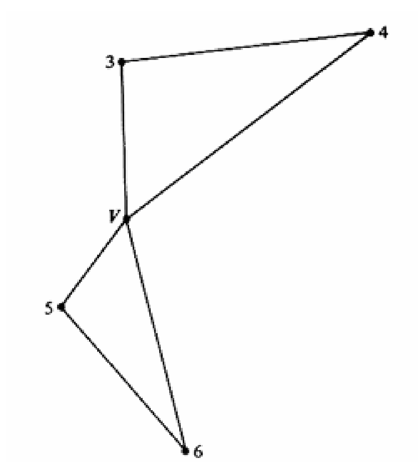
2 TIRIAMOJI DALIS

2.1 2-TSP UŽDAVINIO SPRENDIMAS ŠAKŲ-RIBŲ METODU

Pateiksiu dviejų keliaujančių pirklių 2 – TSP uždavinio pavyzdį.
Duota simetrinė atstumų matrica, pateikta lentelės pavidalu

	V	3	4	5	6
V	/	28	57	20	45
3	28	/	47	46	53
4	57	47	/	76	85
5	20	46	76	/	40
6	45	53	85	40	/

ir geometriniu matricos vaizdiniu.



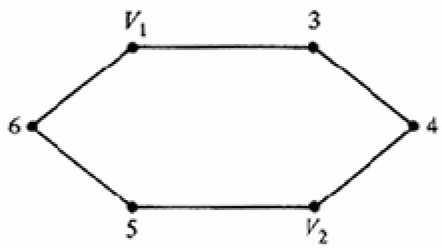
Keturi taškai yra aplankomi iš pradžios taško V dviejų “pirklių”. Taigi, mums reikia dviejų pradžios taško V kopijų, kurias pažymėsime V_1 ir V_2. Pateikiama nauja atstumų matricos lentelė:

Lentelė Nr. 2.1

Atstumų lentelė, 2-TSP uždaviniui

	V_1	V_2	3	4	5	6
V_1	∞	∞	28	57	20	45
V_2	∞	∞	28	57	20	45
3	28	28	∞	47	46	53
4	57	57	47	∞	76	85
5	20	20	46	76	∞	40
6	45	45	53	85	40	∞

Ir geometrinis vaizdinys:



Pastebėkim, kad V_1 ir V_2 eilutės (ir stulpeliai) yra identiški ir turime $d(V_1 V_2) = d(V_2, V_1) = \infty$,

Šakų – ribų metodu ieškosime trumpiausio kelio.

Skaičiavimai pradedami matricoje M_0. Randami h_i, tai mažiausi skaičiai iš kiekvienos eilutės.

	V1	V2	3	4	5	6	h_i
V1	∞	∞	28	57	20	45	20
V2	∞	∞	28	57	20	45	20
3	28	28	∞	47	46	53	28
4	57	57	47	∞	76	85	47
5	20	20	46	76	∞	40	20
6	45	45	53	85	40	∞	40

$$\sum_{i=1}^{6} h_i = 20 + 20 + 28 + 47 + 20 + 40 = 175$$

Perskaičiuojama matrica M_0, t.y. iš kiekvieno eilutės elemento atimamas h_i, ir gauname matricą, jos kiekviename stulpelyje randame mažiausią elementą.

M_1	V1	V2	3	4	5	6
V1	∞	∞	8	37	0	25
V2	∞	∞	8	37	0	25
3	0	0	∞	19	46	53
4	10	10	0	∞	29	85
5	0	0	26	56	∞	20
6	5	5	13	45	0	∞
h_j	0	0	0	19	0	20

$$\sum_{i=1}^{6} h_j = 19 + 20 = 39$$

Perskaičiuojama matrica M_1, t.y. iš kiekvieno stulpelio elemento atimamas h_j, ir gauname

M_2	V1	V2	3	4	5	6
V1	∞	∞	8	18	0	5
V2	∞	∞	8	18	0	5
3	0	0	∞	0	46	33
4	10	10	0	∞	29	65
5	0	0	26	37	∞	0
6	5	5	13	26	0	∞

Skaičiuojame dydį H vadinamą mažiausia viršūnės reikšme. $H = \Sigma h_i + \Sigma h_j = 175+39 = 214$

Kiekvienam matricos M_2 elementui $C_{rs}=0$ apskaičiuojama bauda:

$h_{15} = \min\{C_{1j}\} + \min\{C_{i5}\} = \min\{\infty, \infty, 8, 18, 5\} + \min\{0, 46, 29, \infty, 0\} = 5 + 0 = 5$

$h_{25} = 5,\ h_{31} = 0,\ h_{32} = 0,\ h_{34} = 18,\ h_{43} = 18,\ h_{51} = 0,\ h_{52} = 0,\ h_{56} = 5,\ h_{65} = 5,$

Randama maksimali bauda

$h_{kl} = \max\{h_{15}, h_{25}, h_{31}, h_{32}, h_{34}, h_{42}, h_{51}, h_{52}, h_{56}, h_{65}\} = \max\{5, 5, 0, 0, 18, 18, 0, 0, 5, 5\} = 18$, $h_{kl} = h_{43}$.

Indeksas $(k,l) = (4,3)$ kaip tik ir rodo medžio šakojimosi viršūnę, o kartu ir matricos redukavimo elementą. Sumažiname matricą, t.y. išbraukiame k-ąją eilutę ir l-ąjį stulpelį. Be to, pervežant matricą, reikia įrašyti $C_{34} = \infty$ (tuo uždraudžiame tokius kelius, kurie sudarytų ciklus, t.y. grįžtume į tą patį miestą, nenuvažiavę į kitus). Išbraukus eilutę ir stulpelį bei uždraudus negalimus maršrutus, gaunamas naujas prekybos agento uždavinys su vienetu mažesniu miestų skaičiumi. Skaičiavimo procesas kartojamas iki tol, kol gaunama viršūnė su vieninteliu maršrutu.

M_3	V1	V2	4	5	6	h_i
V1	∞	∞	18	0	5	0
V2	∞	∞	18	0	5	0
3	0	0	∞	46	33	0
5	0	0	37	∞	0	0
6	5	5	26	0	∞	0

$$\sum_{i=1}^{5} h_i = 0$$

M_4	V1	V2	4	5	6
V1	∞	∞	18	0	5
V2	∞	∞	18	0	5
3	0	0	∞	46	33
5	0	0	37	∞	0
6	5	5	26	0	∞
h_j	0	0	18	0	0

. $H = \Sigma h_i + \Sigma h_j = 0 + 18 = 18$

Vėliškiekvienostulpelioatimamejomažiausi ąelement ą. Matricinėjelentel ėje M_5 minusuoti stulpelių minimalūselementai, irprienulini ųelement ųsurašytosj ųbaudos,kuriodidžiausiabaud ą, tąišbraukiame

M_5	V1	V2	4	5	6
V1	∞	∞	B=0 0	B=0 0	5
V2	∞	∞	B=0 0	B=0 0	5
3	B=0 0	B=0 0	∞	46	33
5	B=0 0	B=0 0	19	∞	B=5 0
6	5	5	8	B=5 0	∞

Elementodidžiausiabauda – 5. Išbraukiame to elementoeilut ęirstul pelį. Lieka

M_6	V1	V2	4	5
V1	∞	∞	0	0
V2	∞	∞	0	0
3	0	0	∞	46

6	5	5	8	∞

Matricos lentelėje M_6 kiekvienoje eilutėje ir kiekviename stulpelyje yra nulinių elementų. O nulinių elementų didžiausia bauda yra 5. Tuomet gauname, kad trumpiausias atstumas yra 214+18+5 =237. Optimaliausias maršrutas gaunamas einant viršūnėmis: V_1-3-4-V_2-5-6- V_1. Sujungiame V_1 ir V_2 į vieną pradinį V tašką, gaunamas tas pats, gautas dviejų –TSP sprendimas su atstumo ilgiu 237 vienetų.

2.2 TIRIAMOJO MODELIO „POPIERIAUS CENTRAS“ APRAŠYMAS, PRIELAIDOS

Turim grafą G=(V, E), kur V – viršūnių taškų aibė, E – viršūnes jungiančių lankų aibė (kelias nuo sandėlių iki parduotuvių, ir tarp parduotuvių). Pasižymėsim $\{v_0\} \in V$ Kauno bazės (tiekėjo taškas) viršūnės, visos viršūnės $v \in V \setminus \{v_0\}$ atitinka gavėjus, kiekvienas lankas $e \in E$ žymi ryšius tarp viršūnių. Raide K pasižymim aibę krovininių automobilių, o k -ojo , kai $k \in K$ automobilio talpą žymėsim C_k. Kelionės kaina, keliaujant lanku $e \in E$ automobiliu k yra $c_{ek} > 0$.

Kainos $c_{ek} > 0$ skaičiavimas:

Pervežimo kaina priklauso nuo nuvažiuotų kilometrų ir išsikrovimo taškų skaičiaus. Didžiausiai kroviniai mašinai (24t arba $C_1 = 80$ m^3 talpos) skaičiuojam 2.20 Lt/1 km, o už išsikrovimo tašką 30 Lt./1 Vnt., atitinkamai ir kitoms mašinoms: 10 t arba $C_2 = 35$ m^3 kr. automobiliui - 1,60 Lt/1 km ir 20 Lt/ Vnt.; 5t arba $C_3 = 15$ m^3 kr. automobiliui – 1,20 Lt/1km ir 15 Lt/ 1 vnt.

Modelio krovininių automobilių parkas ir jo struktūra:

Didelio įkrovumo automobiliai ekonomiškesni už mažatonius. Jais pasiekiamas didesnis automobilių išdirbis ir mažėja vežimo savikaina, to paties reikia mažiau vairuotojų. Tačiau ne visada efektyvu naudoti tik didelio įkrovumo automobilius. Krovininių automobilių racionalų panaudojimą lemia šie faktoriai: krovinio rūšis, jo fizinės bei mechaninės savybės ir vežimų organizavimo būdai.

Mūsų atveju yra trijų tipų k_1, k_2, k_3 automobiliai 24t, 10t ir 5t, skaičiavimai bus atliekami ir kubiniais metrais, 1t =3.33 m^3). 24 t talpos automobilių modelyje turim 30 vnt., 10t – 20 vnt., o 5 t – 10 vnt.

Turim 5 „Popieriaus centras“ sandėlius Kaune. Kauno penki sandėliai yra netoli vienas kito, tarp jų atstumų neskaičiuojame, galime laikyti kaip vieną tašką. Čia kaip ir M-TSP uždavinyje, bazės taškas gali būti skaidomas į kopijas.

Tarkim krovininiai automobiliai krovinius pakraunami šiuose sandėliuose. Krovininiame automobilyje pakrautas krovinys gali būti vežamas

- į vieną vietą,
- į kelias vietas;

Krovinys nesukonkretinamas, vienarūšis, vežama paletėmis ar padėklais. Tarkim vežamas popierius, kuriam nereikia specialių krovininių automobilių su šaldytuvais, taip pat užsitęsus pervežimui krovinys nesugenda.

- iš kiekvieno sandėlio gališ vesti nustatytą krovinių kiekį,t.

Įvedama priklausomybė nuo laiko t.y. kad momentas t=0 yra pradinė pasiūla $a_i(0)$ ir pasiūla $b_i(0)$ ir kiekvieną t-tą dieną i-tas vartotojas pareiškia naują paklausą $a_i(t)$.

$$\sum_{i=1}^{m} a_i(t) = \sum_{ji=1}^{n} b_j(t)\,.,\ t \in [0, z]$$

- Dalis krovinių privalo būti pristatyti kitą dieną,
- Tam tikras krovinių kiekis į tam tikrus terminalus yra pristatomi su tam tikru periodu.

Parduotuvių tinklą sudaro 10 parduotuvių – saugyklų. Klientai iš anksto yra davę duomenis, kiek joms reikės prekių (Lentelė priede Nr.1). Turim planą vežamų prekių visiems keturiems mėnesiams į priekį.

Lentelėje pateiktas krovinių x_{ij} kiekis tonomis (arba kub. metrais) pagal mėnesius $(i = 1,2,\ldots 12)$, gaunamų trijų dienų periodu j per mėnesį $(j = 1,2,\ldots 10)$. Kadangi bendrų apimčių pagal mėnesius, dėl jų skirtingo dienų skaičiaus, suvienodinti negalima, tai sąlyginai imama, kad kiekvienas mėnuo turi 30 dienų: iš mėnesių turinčių 31 dieną viena diena atimama.

Čia periodą atitinka trys dienos, vadinsime tridieniais, vadinasi ir užsakymai duoti vienam periodui, atitinka užsakymą trims dienoms.

Vežamos prekės ir krovininiai nedetalizuojami, vežamas nustatytas kiekis kub. metrais.

Vežimai vyksta kalendorinėmis dienomis, savaitgalių ir švenčių neišskiriam.

Automobilių talpos išnaudojimas pervežant krovinius yra 70-80% visos pajėgumo.

Parduotuvių tinklo užsakymai buvo sugeneruoti pagal normalųjį skirtinį su skirtingu vidurkiu kiekvienam mėnesiui ir $\sigma = 150$ taip:

Lapkričio mėn. vidurkis – 766 t.(apkrovumas 79%), gruodžio mėn. – 806t. (83%), sausio mėn. – 680t (s 70%), vasario mėn. – 690 (71 %). Lapkričio ir gruodžio mėnesiais prekių pareikalavimas didesnis, nes tokios tendencijos pastebimos kiekvienais metais transporto firmų užsakymuose.

2.3 VEŽIMŲ IR PARTIJŲ DYDŽIO SKAIČIAUS NUSTATYMAS

Turėdami visą mėnesio ar mėnesių duomenis ir tam tikrus gavėjų nurodymus (kas kiek dienų privalu pristatyti krovinį, į kiek dienų bent kartą atvežti siuntą) nustatysim vežimų ir partijų dydžius.

Turim gavėjų reikalavimą, kad krovinys turi būti pristatytas bent 1 kartą į K_i dienų, vadinasi per K_i dienų reikia vežti $X_i \cdot \frac{K_i}{N}$ (akumuliuotas atvežtas kiekis), čia X_i - i-ojo gavėjo krovinio kiekis per mėnesį, K_i - i-ojo gavėjo reikalavimas, kas kiek dienų bent kartą atvežti krovinį, N – visas periodas (mėnuo), N=10.

i-ojo gavėjo krovinio vežimų skaičius $VS_i = \frac{N}{K_i}$.

Krovinio pervežimo partijos dydis i-ojo gavėjo $q_i = X_i \cdot \frac{K_i}{N}$. Vadinasi po 1-ojo vežimo bus nuvežta q_i, po antrojo $2 \cdot q_i$, t,y $A(VS_i) = X_i \cdot \frac{V_i K_i}{N}$, $V_i = 1,2,...,VS_i$, po paskutinio pervežimo turi būti pervežtas visas krovinys.

Galimas nukrypimas nuo akumuliuoto atvežto kiekio $X_i \cdot \frac{E_i}{N}$, čia E_i - i-ojo gavėjo leistinas vežimo netolygumas, skaičiuojamas dienomis., taip pat duotas ir iš anksto žinomas.

Kaip pavyzdį pateiksiu dviejų įmonių „G“ ir „H“ pavyzdžiu. Pavyzdžiui naudosime lentelės Nr. 3, esančios priede gruodžio mėnesio duomenimis.

Duomenys\Įmonė	„G“	„H“
Galimų vežimų skaičius (N) , vnt	10	10
K(bent 1 kartą atvežti krovinį į K) , dienos	5	2
E, dienos	3	2

Jų vežimų skaičius gaunamas naudojantis formule $VS_i = \frac{N}{K_i}$, vadinasi

$VS_G = \frac{10}{5} = 2$, vadinasi įmonei „G“ per 10 dienų periodą privalu krovinį įvežti bent du kartus, po antrojo pervežimo visas užsakymas turi būti įvykdytas, o $VS_H = \frac{10}{2} = 5$, tai į įmonę „H“ krovininius automobilius reikės siųsti bent penkis kartus ir po penkto atvežimo krovinys turi būti pervežtas.

Akumuliuotas pervežtas kiekis: vežimo partijos dydis $q_i = X_i \cdot \frac{K_i}{N}$

	„G“	„H“
q_i	$1500 \cdot \frac{5}{10} = 750$	$850 \cdot \frac{2}{10} = 170$
V_i	$V_G = 1,2$	$V_H = 1,2,...,5$
$q_i = A(VS_i)_1 = X_i \cdot \frac{1*K_i}{N}$	$1500 \cdot \frac{1 \cdot 5}{10} = 750$	$850 \cdot \frac{1 \cdot 2}{10} = 170$
$A(VS_i)_2 = X_i \cdot \frac{2*K_i}{N}$	$1500 \cdot \frac{2 \cdot 5}{10} = 1450$	$850 \cdot \frac{2 \cdot 2}{10} = 340$
$A(VS_i)_3 = X_i \cdot \frac{3*K_i}{N}$	–	$850 \cdot \frac{3 \cdot 2}{10} = 509$
$A(VS_i)_4 = X_i \cdot \frac{4*K_i}{N}$	–	$850 \cdot \frac{4 \cdot 2}{10} = 680$
$A(VS_i)_5 = X_i \cdot \frac{5*K_i}{N}$	–	$850 \cdot \frac{5 \cdot 2}{10} = 850$

Transporto uždavinys priklauso sveikaskaičių uždavinių tipui. Poreikiai ir ištekliai yra sveikieji skaičiai, kaina – nesvarbu, sudarant pradinį planą ir jį gerinant gaunamas sveikaskaitis atsakymas.

2.4 ATSARGŲ VALDYMO MODELIAVIMAS

Šiam modeliavimui reikia apibrėžti keletą specifinių sąlygų:

- Pradinės atsargos sandėlyje yra 1,5q čia q – vežimų partijos dydis,
- Atsargų lygis tikrinamas kiekvienos dienos pradžioje, kai jis tampa mažesnis kaip kritinis atsargų kiekis (pasirinkau 1.2q), formuojamas naujas užsakymas q,
- Užsakymas įvykdomas per dieną, jei vieną dieną užsakoma, kitą dieną jau atvežama,
- Nesuplanuoti užsakymai įvykdomi per dvi dienas.

Įvertinant sumodeliuotą paklausą reikia sumodeliuoti atsargų kitimą ir įvertinti vidutinį atsargų lygį bei užsakymų skaičių.

Vežamos partijos dydį pasirinksime pagal skyrelio 2.2 skaičiavimus.

Tirsim įmonės „D“ atsargas lapkričio, gruodžio, sausio ir vasario mėnesiais. Šiais mėnesiais vežamų partijų dydis buvo gautas toks 11 mėn. - 353t, 12 mėn. – 380t, 01 – 376t ir 02 – 526t. Vadinasi atitinkamais mėnesiais kritinis atsargų kiekis bus toks: 424t, 453t, 451t ir 631t. Pradinės atsargos 530t.

Modeliavimo rezultatai pateikiami lentelėje (Lentelė Nr.2.2):

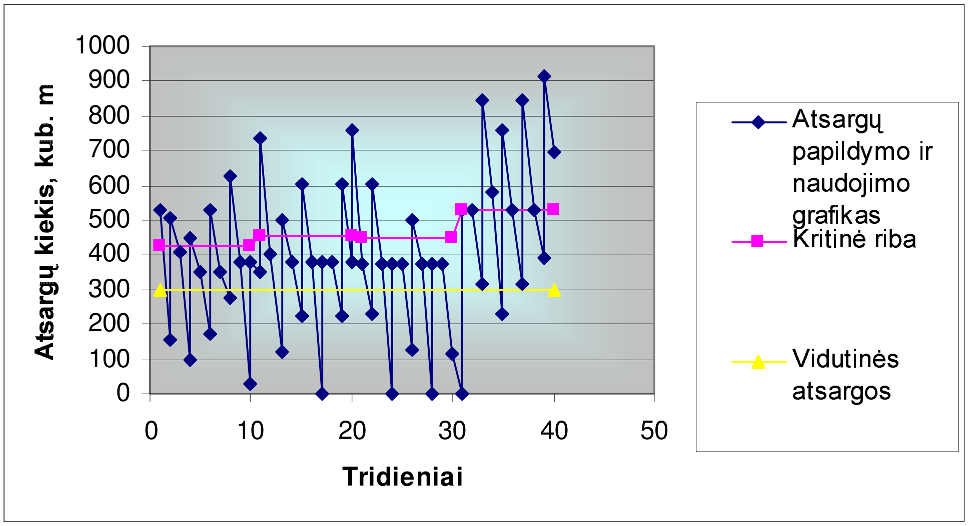
Lentelė Nr. 2.2

Atsargų valdymo modeliavimas su išankstiniais užsakymais

Mėnuo	Diena	Dienos pradžios atsargos	Paklausa	Užsakymo formavimas	Užsakymo įvykdymas	Dienos pabaigos atsargos	Deficitas
	1	530	375			155	0
	2	155	100	užsakymas	353	408	0
11	3	408	312	užsakymas		96	0
	4	96	350	užsakymas	353	353	254
	5	353	180	užsakymas		173	0
	6	173	280	užsakymas	353	353	107
	7	353	79	užsakymas		274	0
	8	274	250	užsakymas	353	377	0
	9	377	350	užsakymas		27	0
	10	27	162	užsakymas	353	353	135
	11	353	330	užsakymas	380	403	0
	12	403	281	užsakymas		122	0
12	13	122	250	užsakymas	380	380	128
	14	380	156	užsakymas		224	0
	15	224	343	užsakymas	380	380	119
	16	380	450	užsakymas		0	70
Mėnuo	Diena	Dienos pradžios atsargos	Paklausa	Užsakymo formavimas	Užsakymo įvykdymas	Dienos pabaigos atsargos	Deficitas
	17	0	10	užsakymas	380	380	10
	18	380	156	užsakymas		224	0
	19	224	280	užsakymas	380	380	56
	20	380	450	užsakymas	376	376	70
	21	376	148	užsakymas		228	0
	22	228	300	užsakymas	376	376	72
01	23	376	442	užsakymas		0	66
	24	0	420	užsakymas	376	376	420
	25	376	250	užsakymas		126	0
	26	126	250	užsakymas	376	376	124
	27	376	401	užsakymas		0	25
	28	0	50	užsakymas	376	376	50
	29	376	263	užsakymas		113	0
	30	113	450	užsakymas		0	337

02	31	0	180	užsakymas	526	526	180
	32	526	208	užsakymas		318	0
	33	318	263	užsakymas	526	581	0
	34	581	349	užsakymas		232	0
	35	232	470	užsakymas	526	526	238
	36	526	208	užsakymas		318	0
	37	318	318	užsakymas	526	526	0
	38	526	136	užsakymas		390	0
	39	390	220	užsakymas	526	696	0
	40	696	120			576	0

301 Vidutinės atsargos	10590 Paklausa		8175 Numatyta pervežti	Suma 2461



2.1 Pav. Atsargų modeliavimas su išankstiniais užsakymais

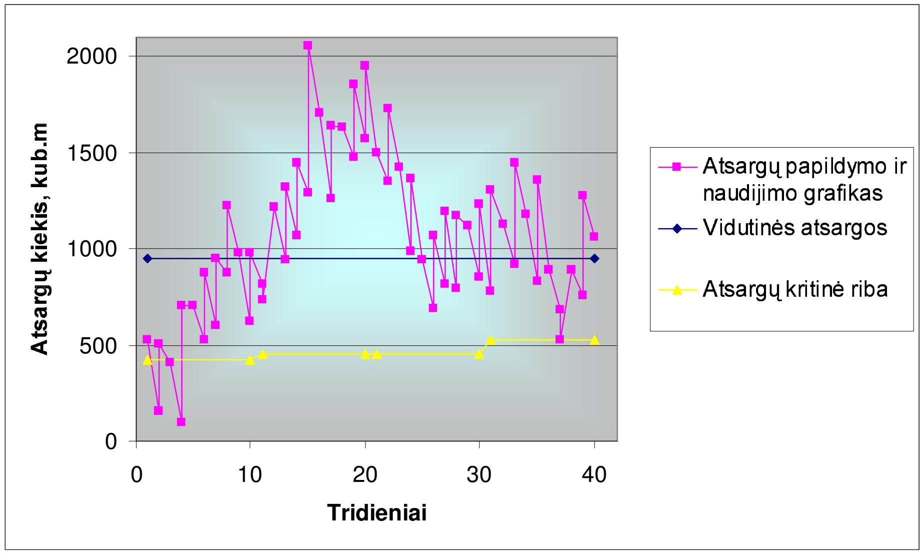
Lentelė Nr. 2.3

Atsargų valdymo modeliavimas su išankstiniais ir ekstra užsakymais

Mėnuo	Diena	Dienos pradžios atsargos	Paklausa	Ekstra užsakymo formavimas	Užsakymo įvykdymas	Dienos pabaigos atsargos	Deficitas
11	1	530	375			155	0
	2	155	100	353	353	408	0
	3	408	312			96	0
	4	96	350		706	706	254
	5	706	180	353		526	0
	6	526	280		353	599	0
	7	599	79		353	873	0
	8	873	250		353	976	0
	9	976	350	353		626	0
	10	626	162		353	817	0

12	11	817	330		733	1220	0
	12	1220	281	380		939	0
	13	939	250	380	380	1069	0
	14	1069	156		380	1293	0
	15	1293	343		760	1710	0
	16	1710	450			1260	0
	17	1260	10		380	1630	0
	18	1630	156			1474	0
	19	1474	280		380	1574	0
	20	1574	450		376	1500	0
1	21	1500	148			1352	0
	22	1352	300		376	1428	0
	23	1428	442			986	0
	24	986	420		376	942	0
	25	942	250	376		692	0
	26	692	250		376	818	0
	27	818	401		376	793	0
	28	793	50	376	376	1119	0
	29	1119	263			856	0
	30	856	450		376	782	0
2	31	782	180		526	1128	0
	32	1128	208			920	0
	33	920	263		526	1183	0
	34	1183	349			834	0
	35	834	470		526	890	0
	36	890	208			682	0
	37	682	318		526	890	0
	38	890	136			754	0
	39	754	220		526	1060	0
	40	1060	120			940	0

952 Vidutinės atsargos	10746 Pervežtas kiekis	254 Deficitas



2.2 Pav. Atsargų modeliavimas su išankstiniais užsakymais ir ekstra užsakymais

2.5 EURISTINIO MODELIO APRAŠYMAS IR JO SPRENDIMO EIGA

Ieškant metodo, kad būtų galima gauti optimaliausią rezultatą: minimalią pervežimo kainą, mažiausią kilometražą, sugaišti kiek įmanoma mažiau laiko ar gauti didžiausią pelną, tenka įtraukti tokius metodus kaip matematinis programavimas. Tačiau kai atsiranda daugiau apribojimų: kroviniai automobiliai skirtingų talpų, automobilių kiekis yra ribotas ir kiti, dažnai tenka naudotis euristiniais modeliais ar juos derinti su matematiniu programavimu. Jais sprendžiant, iš pradžių priimamos tam tikros prielaidos apribojimai.

Mano sprendimo algoritmas ir eiga.

Turim pradinius duomenis. Keturis mėnesius (lapkritis, gruodis, sausis ir vasaris), jie suskirstyti tridieniais, vadinasi kiekvienas mėnuo turės po 10 tridienių. Žinom, kiek kiekvieną tridienį reikia atvežti tam tikrai parduotuvei (gavėjui). Laikome, kad uždavinys subalansuotas, t.y. sandėliuose yra tiek pat prekių, kiek jų reikalauja gavėjai. Automobilių kiekis yra ribotas, jų talpa yra trijų skirtingų dydžių. Kaina priklauso nuo iššikrovimo taškų ir nuo krovinį vežusios mašinos tonažo.

Pirmiausia peržvelgią vieno tridienio užsakytus vežimus, kur reikalavimai dideli (didesni už 80 m^3), siunčiam didžiuosius automobilius. Taip peržiūrim visų įmonių reikalavimus. Kai lieka mažesni kiekiai siunčiame mažesnio tonažo automobilius. Jei jau išnaudoti visi didieji automobiliai tuomet siunčiam mažesnius automobilius, kurie dar yra laisvi.

Visą laiką yra stebimi pervežami kiekiai tarp artimiausių taškų-gavėjų: tarp Vilniaus abiejų įmonių, tarp Tauragės, Šilalės ir Klaipėdos, tarp Kupiškio ir Panevėžio, bei tarp Vabalninko ir Biržų. Jei šių artimiausių gavėjų likęs pervežti krovinys telpa į vieną kokį nors automobilį, tai krovinys vežamas į šiuos taškus vienu reisu, vadinasi automobilis aplankys ne vieną o kelis taškus ir tik tada grįš į bazę.

Jei išnaudoti visi automobiliai, o ne visų gavėjų poreikiai patenkinti, tuomet likęs nepervežtas krovinys, vežamas jau kitą tridienį.

Šiuos aiškinimus pateiksiu spręsdama uždavinį. Darbo ataskaitoje parodau dviejų tridienių ir vieno mėnesio analizę, tačiau skaičiavimus esu atlikus visiems keturiems mėnesiams ir kiekvienam jų tridieniui.

2.5.1 LAPKRIČIO MĖNESIO II-OJO TRIDIENIO PERVEŽIMŲ ANALIZĖ

Turim lapkričio mėnesio II-ojo tridienio duomenis.

Eil. Nr.	Gavėjas	Užsakymas, m^3
1	“A” Vilnius	113
2	“B” Vilnius	77
3	“C”Tauragė	703
4	“D” Šilalė	260
5	“E” Kupiškis	186
6	“F” Panevėžys	210
7	“G” Vabalninkas	216
8	“H” Biržai	110
9	“J” Klaipėda	280
10	“K” Kaunas	52
Krovinių suma, m^3		2207

I-ąjį tridienį visi užsakymai buvo įvykdyti, vadinasi II-ąjį tridienį, pervežamas krovinys atitinka iš anksto žinomą užsakymą.

Į visus 10 gavėjus siunčiam automobilius, kurių talpa 80 m^3. Jau po pirmo automobilių išsiuntimo, kai kurių (šiuo atveju įmonių „B“ ir „K“) poreikiai jau įvykdyti. Peržiūrim kiek kokiai įmonei trūksta ir siunčiam atitinkamus automobilius. Įmonei „A“, kurios užsakymas buvo 113 m^3 jau vienas automobilis atvežė 80 m^3, vadinasi liko atvežti 33 m^3, todėl į šią įmonę siunčiame 35 m^3 talpos automobilį. Jau ir šios įmonės poreikiai įvykdyti.

- Vieno tridienio dviejų įmonių pervežimo sprendimas

Peržvelkim įmonių „E“ ir „F“ pervežamus krovinius. Pirmu ir antru automobilių išsiuntimu iš bazės, šioms įmonėms atvežtas kiekis yra po 160 m^3, t.y. vieną ir kitą įmonę aplankė po dvi krovinines priemones su 80 m^3 talpa, vadinasi įmonei „E“ liko atvežti 26 m^3, o įmonei „F“ – 50 m^3. Kadangi šias įmones skiria nedidelis atstumas 51 km, tai būtų protinga vienu 80 m^3 talpos automobiliu pervežti krovinį šioms dviems įmonėms kartu. Taip sukombinavus pervežimą šiuo reisu būtų nuvažiuoti 361 km (Maršrutas: Kaunas – Kupiškis – Panevėžys – Kaunas), atlikti du išsikrovimai po 30 Lt už vieną išsikrovimą, vežant krovinį 80 m ^{3} talpos automobiliu vieno kilometro kaina 2,20 Lt. Vadinasi pervežimo kaina būtų $361 \cdot 2,20 + 2 \cdot 30 = 854,20 Lt$

Jei darytume du pervežimus, tai pervežant į įmonę „E“ būtų nuvažiuota 330 km automobiliu, kurio talpa 35 m^3, kaina už kilometrą pervežant šiuo automobiliu būtų 1,60 Lt, už išsikrovimo tašką skaičiuojama 20 Lt. Vadinasi šio pervežimo kaina būtų $330 \cdot 1,60 + 20 = 548 Lt$. Vežant įmonei „F“ būtų nuvažiuota 290 km automobiliu, kurio talpa 80 m^3 , tai pervežimo kaina $290 \cdot 2,20 + 30 = 668 Lt$. Abiejų reisų kaina $548 + 668 = 1216 Lt$ gaunama žymiai didesnė kaina, nei vežant kombinuotu reisu. Vykdant du reisus, būtų siunčiamos dvi transporto priemonės, skaičiuojamas dviejų vairuotojų darbas, dviejų automobilių nusidėvėjimas, be to automobilio, kuris vežė 50 m ^{3} krovinį, būtų labai neefektyviai išnaudotas automobilio talpa. Šiame pavyzdyje puikiai matome, kad reikia naudoti kombinuotą reisą, net jei ir transporto priemonei tektų palaukti, kokią valandą ar dvi (dėl prastovų arne spėjant iškrauti), toks reisas vis tiek apsimokėtų.

Pervežimo kaina būna nustatyta iš anksto, kokiu būdu bus skaičiuojama. Ji gali būti:

fiksuota (tam tikra suma už tam tikrą reisą, su tam tikros talpos automobiliu),

skaičiuojami kilometrai ir dauginami iš numatytos vieno kilometro už pervežimą kainos,

fiksuota kaina kaip dalinio krovinio (dažniausia naudojama, kai vežami kelių įmonių kroviniai).

Šioms įmonėms pirmi du krovinių pervežimai kainavo

„E“ įmonei $330 \cdot 2,20 + 30 = 756$ Lt, kadangi buvo du tokie reisai , tai bendrai 1512 Lt.,

„F“ įmonei $290 \cdot 2,20 + 30 = 668$ Lt, o tokie reisai buvo du, vadinasi abu reisai įmonei kainuos 1336 Lt.

Kadangi trečiasis reisas buvo kombinuotas, nebuvo sių stos dvi mašinos, tai vežėjai tokiais atvejais dažniausia derinasi su užsakovais: kartais įmonėms padalina kilometrus ir abiem įmonėms tenka mokėti mažesnes sumas už pervežimą (nauda ir gavėjams, ir vežėjui).

Panaši situacija ir su įmonių „D“ ir „J“ krovinių pervežimu. Pirmais trimis pervežimais, abiem įmonėm buvo pervežta po 240 m ^{3} tūrio krovinių, automobiliais, kurių talpa 80 m ^{3}, vadinasi įmonei „D“ liko atvežti 20 m ^{3}, o įmonei „J“ – 40 m^3. Kadangi tarp šių įmonių atstumas yra 81 km, tai darom kombinuotą pervežimą. Abiem įmonėms krovinį vešime vienu krovininių automobiliu, kurio talpa yra didžiausia, iš turimų automobilių. Maršruto Kaunas – Šilalė – Klaipėda – Kaunas kilometražas bus 456 km. Atliekami du išsikrovimai, vadinasi pervežimo kaina būtų

$456 \cdot 2{,}20 + 2 \cdot 30 = 1063{,}20 Lt$. Atliekant pervežimus su dviem automobiliais tektų važiuoti vienam automobiliui 310 km (į Šilalę), kitam 440 km (į Klaipėdą).

- Viso tridienio pervežimų sprendimų analizė

Visoms įmonėms pervežti planuojamo krovinio tūris yra 2207 m^3, per tridienį turimais automobiliais galima pervežti 3250 m^3 ($30 \cdot 80 + 20 \cdot 35 + 10 \cdot 15 = 3250$), vadinasi tikrai įmanoma pervežti numatytą krovinį. Pagal mano skaičiavimo metodą, šio tridienio kroviniams pervežti buvo naudota 30 automobilių (28 automobiliai 80 m^3 talpos, ir 2 – 35 m^3 talpos). Vežant krovinius šiais automobiliais, gauta, kad jų talpa išnaudota 95,33%. Pervežimo kaina, jei tai viskas būtų vežta vienai ir tai pačiai įmonei, būtų 20865,40 Lt (čia sumuojami pervežimų kilometrai, jie dauginami iš atitinkamų kainų ir pridedama sis iškrovimo taškų skaičius padaugintas iš atitinkamo įkainio). Per šį tridienį krovininiai automobiliai pravažiavo 9667 km.

2.5.2 LAPKRIČIO MĖNESIO VI-OJO TRIDIENIO PERVEŽIMŲ ANALIZĖ

Lapkričio mėn. VI-ojo tridienio duomenys:

Eil. Nr.	Gavėjas	Užsakymas, m^3
1	"A" Vilnius	303
2	"B" Vilnius	150
3	"C"Tauragė	565
4	"D" Šilalė	100
5	"E" Kupiškis	259
6	"F" Panevėžys	476
7	"G" Vabalninkas	253
8	"H" Biržai	186
9	"J" Klaipėda	766
10	"K" Kaunas	193
Krovinių suma, m^3		3251

Prieš tai buvusiame skyrelyje rašiau, kad su turimomis mašinomis, maksimaliai pervežti galima 3250 m^3, vadinasi negalėsim įvykdyti visų užsakovų prašymų.

Šį kartą visi reisai vykdomi tik į vieną įmonę, nesusidaro kombinuoti reisai. Jau penktajame automobilių išsiuntimo žingsnyje gauname, kad panaudoti visi 80 m^3 talpos automobiliai, o dar liekę keletas nemažos talpos krovinių (didesnių nei 80 m^3). Tenka įjungti automobilius su mažesne talpa. Suprantama tai labai neekonomiška, nes ilgus maršrutus kelis kartus tenka važiuoti 35 ar 15 m^3 talpos automobiliais, kai tuo tarpu, jei būtų laisvų 80 m^3 talpos automobilių kai kuriais atvejais užtektų ir vieno reiso.

Tirkim „J" įmonės pervežimą. Jos užsakymas 766 m^3, iš pradžių vykdomas 80 m^3 talpos automobiliais. Penkios C_1 talpos mašinos perveža 400 m^3 ($5 \cdot 80 = 400$). Lieka pervežti 366 m^3

$(766 - 400 = 366\,\mathrm{m}^3)$. Turim, kad pervežti liko pakankamai daug, tačiau jau išnaudotos visos C_1 talpos mašinos, toliau į šią įmonę siųsime $C_2 = 35\,\mathrm{m}^3$ talpos automobilius, o kai nebeliks ir jų laivų, siųsim $C_3 = 15$ talpos automobilius. Vadinasi šios įmonės pervežimui naudota tiek krovininių automobilių:

Automobilio talpa, m^3	Automobilių skaičius, Vnt.	Pervežtas tūris, m^3
80	5	400
35	5	175
15	6	90

Gavom, kad per šį tridienį pervežta 665 m^3, liko nepristatyta gavėjui 101 m^3. Mano uždavinyje, tokiu atveju, likęs krovinys vežamas kitą tridienį.
Už tai, kad nepristatytas visas krovinys, gali nukentėti vežėjas, jam gali būti išstatyta bauda ar sumažinta pervežimo kaina.

Taip tiriami visi keturių mėnesių tridieniai su išankstiniais užsakymais. Darbe taip pat skaičiuojami pervežimai ir su ekstra pervežimais. Pervežimuose su ekstra užsakymais susidaro žymiai daugiau neišvežtų krovinių. Čia jau vežėjo sprendimas, ką pasirinkti ar rinktis išankstinius užsakymus, kuriems priemones galima rezervuoti iš anksto, ar pasiimti daugiau ekstra užsakymų, už kuriuos yra daugiau mokama. Vežėjai turi labai apsiskaičiuoti kokia schema jie turi dirbti, kaip pasirinkti maršrutus, kokią dalį rezervuoti ekstra užsakymams, kaip vežti krovinius, kad susidarytų kuo mažesnė bauda, jei kroviniai laiku nepristatomi į vietą.

2.5.3 VISO LAPKRIČIO MĖNESIO PERVEŽIMŲ ANALIZĖ

Per lapkričio mėnesį buvo tir tai 10 tridienių. Kiekvieną tridienį galima analizuoti kaip aukščiau esančiame skyrelyje nagrinėjau III ir VI-ąjį tridienius.

VI, IX ir X tridieniais ne visų klientų užsakymais įvykdomi. Todėl VIII ir X tridieniai ir gruodžio mėnesio I tridienis pasipildo, prieš tai buvusių tridienių neišvežtu kroviniu.

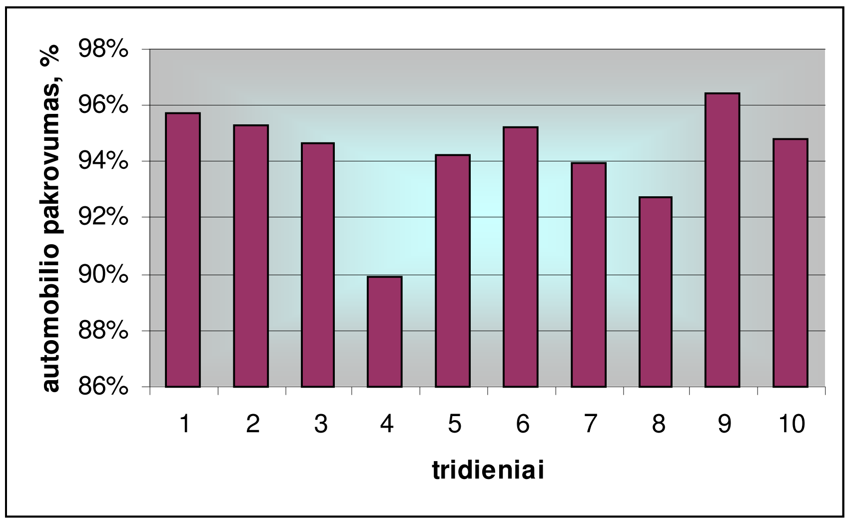
Lentelėje (Lentelė Nr.) pateikiu lapkričio mėnesio pervežimų tam tikrus skaičius:

Lentelė Nr. 2.4

Lapkričio mėnesio pervežimų gauti skaičiai

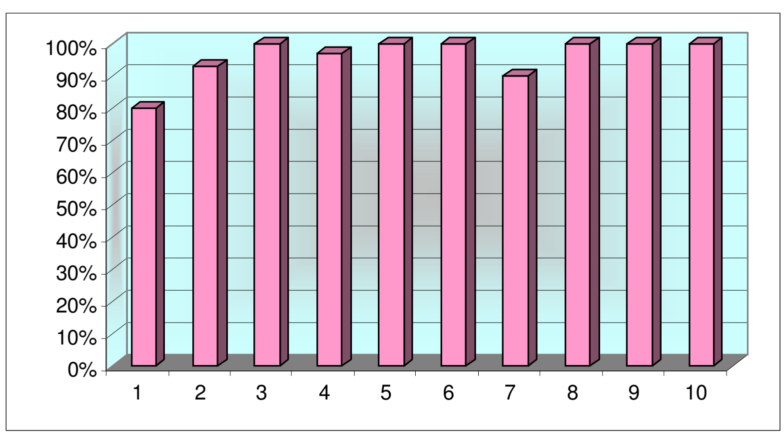
Tridienis		1	2	3	4	5	6	7	8	9	10
Numatytas vežti kiekis, m^3		2067	2207	2924	2386	2610	3251	2182	2551	3328	3315
Pervežtas kiekis, m^3		2067	2207	2924	2386	2610	3150	2182	2551	3158	3179
Pakrovumas		95.70%	95.33%	94.66%	89.94%	94.22%	95.20%	93.94%	92.72%	96.42%	94.77%
Automo-bilio tipas	15 m^3	0	0	4	3	4	10	1	2	10	10
	35 m^3	6	2	16	4	7	20	3	7	20	20
	80 m^3	24	28	30	29	30	30	27	30	30	30

Lapkričio mėnesį pakankamai aukštas krovinių automobilių užpildymas kroviniais lygis, vadinasi pakankamai neblogai išnaudota automobilių talpa, parodyta paveiksle (2.3 Pav.)



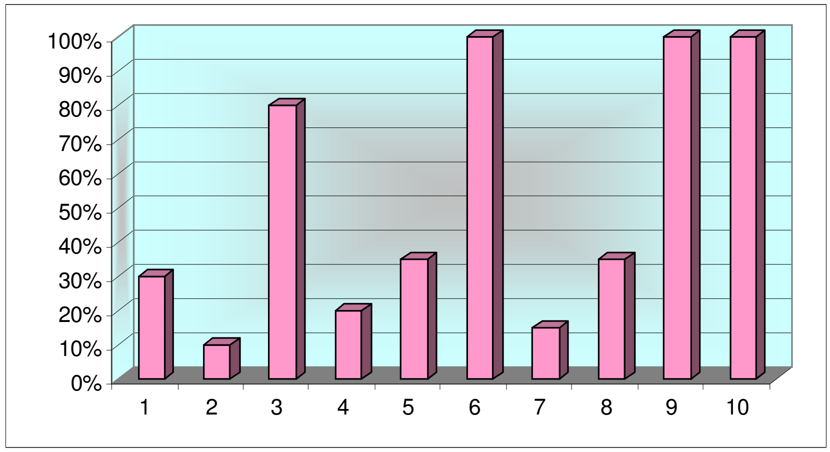
2.3 Pav. Lapkričio mėnesio automobilių užpildymas kroviniais

Daugiausia naudota didžiausios talpos automobilių. Jų naudojimo procentas parodytas paveiksle (Pav. Lapkričio mėnesio 80 m ^{3} talpos automobilių naudojimas)

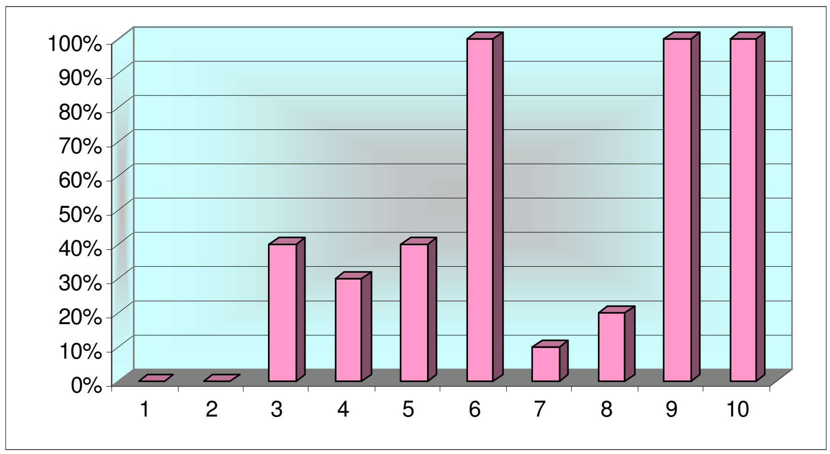


2.4. Pav. Lapkričio mėnesio 80 m ^{3} talpos automobilių naudojimas

Atitinkamai sekančiuose paveiksluose pateikiamas 35 m^3 ir 15 m^3 talpos krovininių automobilių naudojimas kiekvieną tridienį.



2.5. Pav. Lapkričio mėnesio 35 m^3 talpos automobilių naudojimas



2.6. Pav. Lapkričio mėnesio 15 m^3 talpos automobilių naudojimas

2.5.4 ĮMONĖS „A“ ATSARGŲ VALDYMO MODELIAVIMAS LAPKRIČIO MĖN.

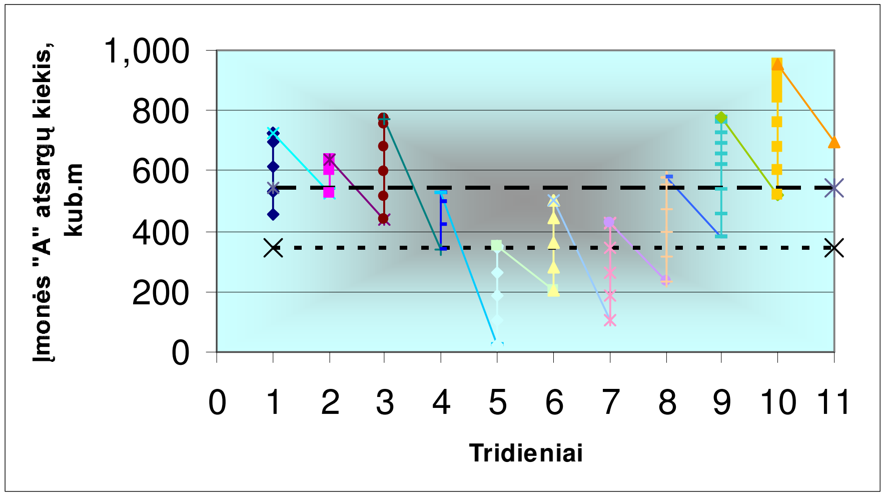
Šiame tyrime naudojame prieš tai skyriuose gautus duomenis ir priimtas taisykles (prielaidas).

- Pradinės atsargos, kurios yra įmonės „A“ sandėlyje yra 1,5 q čia q – vežimų partijos dydis, kuris šiai įmonei gautas pagal skyriuje 2.2 aprašytą nustatymo metodai, vežimo partijos dydis paimtas iš lentelės, esančios prieduose (Lentelė Nr. 1 Pervežimų partijos dydžiai) gautas 303 m^3, vadinasi pradinės atsargos $1.5 \cdot 303 = 455$ m^3
- Pervežimai vykdomi kiekvieną dieną;
- Paklausa – sugeneruoti atsitiktiniai dydžiai;

- Kritinis atsargų kiekis (pasirinkta 1.2q), vadinasi kritinis atsargų kiekis $1{,}2 \cdot 303 = 364$ m^3 .

Analizuojant atsargų kitimą, įvertiname vidutinį atsargų lygį bei užsakymų skaičių.

Įmonės „A“ atsargų kiekio kitimo grafikas pavaizduotas paveiksle (Pav. Atsargų kiekio kitimas). Šio grafiko laužtinė rodo atsargų kiekio kitimą, stambesnė brūkšninė linija vaizduoja kritinę ribą, smulki brūkšninė linija vaizduoja vidutines atsargas.



2.7. Pav. Atsargų kiekio kitimas

Iš šio grafiko matome, kiekvieną tridienį atvežamas krovinys, kiekvieną tridienį buvo tam tikra paklausa. Nagrinėjant modeliavimo rezultatus, matyti, kad penktąjį tridienį dienos pražios atsargų lygis yra 0, kaip ir ketvirtąjį tridienį užfiksuotas deficitas (paklausa buvo didesnė, nei turėta atsargų sandėlyje), todėl įmonė turėjo patirti nuostolius: negaunama pajamų, kyla rizika, kad pirkėjai gali nebeateiti. Kritinė riba pasiekta beveik kiekvieną tridienį. Gavėjai turi didinti atsargas sandėlyje, t.y. didinti atvežimo partijas ar pridėti papildomų užsakymų pervežimams.

2.6 PRADINIO UŽDAVINIO SPRENDINIO RADIMAS

Transporto uždaviniams spręsti naudojami specialūs transporto uždavinių metodai, kurie vadinami paskirstymo metodais.

Paskirstymo metodus patariama kombinuoti su euristiniais paskirstymo metodais, nes tai padeda išvengti ilgų ir varginančių skaičiavimų. Daroma taip: problema sprendžiama kuriuo nors euristiniu metodu, ir gaunamas tariamai optimalus sprendinys. Paskui laikant šį tariamai optimalų sprendinį pradiniu, uždavinys toliau sprendžiamas kuriuo nors tikslinamuoju metodu tol, kol gaunamas tikrai optimalus sprendinys. Šiame skyriuje nagrinėsime tokius pradinių sprendinių ieškojimo metodus: mažiausiojo elemento ir Fogelio aproksimacijos.

Pradinio sprendinio radimui naudosime duomenis, jie pateikti priede (Lentelė Nr. 7). Turim subalansuotą transporto uždavinį. Atsargos, esančios sandėliuose, ir gavėjų poreikiai sutampa. Turime 10 parduotuvių pervežti numatytą krovinį. Darom prielaidą, kad sandėlis – bazė yra išskaidoma pagal turimų automobilių talpą, jų kiekis ir pervežamas krovinių tūris tikrinamas su užsakymais. Konkrečiu atveju Lentelėje Nr. 7 yra numatyti kroviniui pervežti 33 krovininiai automobiliai. Iš jų 24 automobiliai - 80 m^3 talpos automobiliai, 7 – 35 m^3 talpos automobiliai ir po vieną 27 ir 15 m^3 talpos automobilį. Pervežimo kainos šioje matricos lentelėje skaičiuojamos taip: atstumas iki miesto ir atgal dauginamas iš atitinkamo tarifo, kuris priklauso nuo krovinį pervežančio krovininio automobilio talpos, ir pridedamas į kainis užšsikrovimo tašką, kuris taip pat priklauso nuo krovinį pervežančio krovininio automobilio talpos. Pagal šiuos pradinius duomenis padaryti krovinių paskirstymai mažiausio ir Fogelio aproksimacijos metodais. Kai gauti krovinių paskirstymai, pradinis sprendinys nėra gaunamas skaičiuojant tikslo funkciją, nes ją skaičiuojant mano uždavinyje gautume nekorektišką sprendimą. Pavyzdžiui, jei žiūrėtume mažiausio elemento metodu gautą paskirstymą automobilio Nr. 10. Priedo Lentelė 8. 10 –ojo krovininio automobilio krovinys yra išskaidomas taip: 8 m^3 talpos krovinys yra skirtas Šilalės įmonei, ir 72 m^3 talpos krovinys – Kupiškio įmonei, jei neatsižvelgtume į krovinio talpą, iš pradinių duomenų gautume, kad reiso kaina lygi tų dviejų maršrutų kainai, kaip iš Kauno miesto: 682+726=1418 Lt. Todėl, kur krovininio automobilio krovinys išskaidomas į kelis miestus, reiso kainą perskaičiuojame, atsižvelgdami į kilometrus tarp tų miestų, krovinio tūrį.

2.6.1 PRADINIO SPRENDINIO RADIMAS MAŽIAUSIOJO ELEMENTO METODU

Sudaroma vertės koeficientų lentelė, įrašant į ją turimus resursus stulpelyje „Atsargos“, pagal kiekvieną eilutę, taip pat kiekvieno vartotojo poreikių sumas eilutėje „Poreikiai“.

Mūsų tikslo funkcijos ieškomas pobūdis yra minimumas.

Peržiūrint iš eilės kiekvieną vertės koeficientų lentelės elementą, surandamas minimalus koeficientas. Šiuo atveju būtų 36 (pervežimo kaina 36 Lt, maršrutu Kaunas bazė, Kaunas, automobilių, kurio pervežimo talpa 15 m^3).

Rastoji kaina, rodo, langelį, į kurį resursų paskirstymo lentelėje bus įrašomi duomenys. Palyginami to stulpelio, į kurį reiks įrašyti duomenis, poreikiai su tos eilutės, į kurią bus įrašyti duomenys, atsargomis ir mažesnysis iš tų dviejų skaičių pasirenkamas ir įrašomas į lentelę. Atsargos 33 sandėlio yra 15, o poreikis įmonės „K“ Kaunas yra 52, vadinasi į langelį (33,10) įrašome 33 sandėlio atsargas 15.

Iš eilutės, į kurią įrašytas skaičius, atsargų sumos atimamas į lentelę įrašytas skaičius, ir likutis įrašomas į stulpelio „Atsargos“ atitinkamos eilutės langelį, prieš tai užbraukus tame langelyje buvusią atsargų sumą.

Tiekimo punktai	....	Gavėjai H Biržai	J Klaipėda	K Kaunas	Atsargos
		...	...	...	...
...		0	0	0	35
		0	0	0	35
		0	0	0	35
		0	0	0	35
		0	0	0	35
30		0	0	0	35
31		0	0	0	35
32		0	0	0	27
33		0	0	15	0 15
Poreikis	...	110	280	37 52	

Iš stulpelio, į kurį įrašytas skaičius, poreikių sumos atimamas į lentelę įrašytas skaičius ir likutis įrašomas į eilutę „Poreikiai“ atitinkamo stulpelio langelį, prieš tai užbraukiame tame langelyje buvusią poreikių sumą.

Jei atitinkamos eilutės arba stulpelio duomenys rodo, kad sunaudotos visos atsargos ir patenkinti visi poreikiai, ši eilutė ar stulpelis yra išbraukiami ir toliau skirstyme nebedalyvauja. Mūsų atveju sandėlio Nr.33 visos atsargos panaudotos, vadinasi šią eilutę braukiame iš lentelės.

Toliau vėl viską kartojame iš pradžių: lentelę peržiūrime, randame mažiausią elementą ir taip toliau. Toks skaičiavimas baigiasi išbraukus visas eilutes ir stulpelius lentelėje. Tuomet skaičiuojame pradinio sprendinio tikslo funkcijos reikšmę.

Lentelė su gautais paskirstytais pervežimais pateikiama priede Lentele Nr. 8. Įmonei „J“ Klaipėdoje 280 m^3 krovinio, krovinys pervežtas keturiais krovininiais automobiliais, kurių talpa 80 m^3, vienas iš vežė tik 40 m^3, pagal šį metodą, likusi nepanaudota automobilio talpa buvo skirta pervežti įmonei „H“ tai pat 40 m^3. Realiame gyvenime, toks paskyrimas netiktų, nes Klaipėdą ir Biržus skiria didelis atstumas, geriau skirsti du automobilius, arba kombinuoti su kitomis įmonėmis, kurios yra arčiau tų miestų.

Paskirstymų lentelė, šiuo atveju pildoma iš apačios kairiosios pusės, nes taip pasiskirsčiusios pervežimo kainos, pirmiausia buvo paskirstyti kroviniai Kauno mieste esančiai įmonei, po to Vilniaus miestui, Tauragei, Panevėžiui, Šilalei, Kupiškiui, Vabalninkui, Biržams ir galiausia Klaipėdai. Pervežimo kainos į Klaipėdą yra didžiausios.

Taigi gaunamas toks pradinis pasiskirstymas:

- iš 1 – 3 –ųjų: reikia po 80 m^3 talpos krovinio siųsti į "J";Šių maršrutų kainos yra po 968 Lt iš Lentelės Nr.7.
- iš 4 – ojo: reikia 40 m^3 siųsti į "H" ir 40 m^3 siųsti į „J". Būtų nekorektiška imti reiso kainą, kaip dviejų krovininių automobilių išsiuntimą ir sugrįžimą į Kauną.Reisų kainų modeliavimui pirmiausia reikia įvertinti kiekvieno automobilio, kuris perveža krovinį, vieno kubinio metro krovinio pervežimo kainą. Vieno kubinio metro krovinio talpos pervežimo kaina pateikta lentelėje priede (Lentelė Nr. 9). Šio reiso kainą galim skaičiuoti taip: kiekvieną pervežtą kubinį metrą krovinio reikia dauginti iš jo įkainio, 40*11,55+40*12,10 ir pridėti sandaugą, pravažiuotų kilometrų nuo Biržų iki Klaipėdos (275 km) padaugintų iš atitinkamo krovininio automobilio kilometro kainos (automobiliui, kurio talpa 80 m^3 , kaina yra 2,20 Lt).Automobilis pirmiausia važiuoja į Biržus, nes mažesnis atstumas, tada su juo į Klaipėdą su likusiu kroviniu. Jei automobilio krovinys skirtas dviems įmonėms, tai pirmiausia jis važiuoja į tą miestą, kuris yra arčiau. Vadinasi šio reiso kaina yra 1551,00 Lt.
- iš 5 – ojo: reikia 10 m^3 siųsti į "G" ir 70 m^3 siųsti į „H". Čia elgi kiekvieno taško pervežtą krovinio kubinį metrą dauginam iš atitinkamos kainos ir pridedame sumą, kuri gauta kilometrus (25 km) tarp Vabalninko ir Biržų dauginant iš 2,20 Lt. Šio reiso kaina 10*10,18+70*11,55+2,20*25=965,30 Lt
- iš 6 ir 7 –ojo: reikia po 80 m^3 siųsti į "G"; Kaina po 814 Lt;
- iš 8 – ojo: reikia 34 m^3 siųsti į "E" ir 46 m^3 siųsti į "G"; Reiso kaina 34*9,08+46*10,18+2,20*51=
889,20 Lt
- iš 9 – ojo : reikia 80 m^3 siųsti į "E", reiso kaina – 726 Lt;
- iš10 – ojo: reikia 8 m^3 siųsti į "D" ir 72 m^3 siųsti į "E". Reiso kaina 8*8,53+72*9,08+2,20*260= 1294 Lt;
- iš 11 – 13 – ųjų: reikia po 80 m^3 siųsti į "D", šių reisų kaina 682 Lt;
- iš 14 – ojo: reikia 12 m^3 siųsti į "D" ir 68 m^3 siųsti į "F", kaina:12*8,53+68*7,98+2,20*202= 1089,40 Lt;
- iš 15 – ojo: reikia 80 m^3 siųsti į "F", kaina 638 Lt;
- iš 16 – ojo: reikia 18 m^3 siųsti į "C" ir 62 m^3 siųsti į "F", kaina 18*7,98+7,98*62+2,20*200= 1078,40 Lt;
- iš 17 – 24 – ųjų : reikia po 80 m^3 siųsti į "C", reisų kainos po 638 Lt;

- iš 25 – ojo: reikia 35 m^3 siųsti į "C", kaina 464 Lt;
- iš 26 – ojo: reikia 25 m^3 siųsti į "B" ir 10 m^3 siųsti į "C", kaina 25*12,34+10*13,26+1,6*270=873,10 Lt;
- iš 27 – ojo: reikia 35 m^3 siųsti į "B", reiso kaina 464 Lt;
- iš 28 – ojo: reikia 18 m^3 siųsti į "A" ir 17 m^3 siųsti į "B", reiso kaina 18*11.43+17*12.34+ 1,60*40= 479,52 Lt
- iš 29 ir 30 – ųjų: reikia po 35 m^3 siųsti į "A", reisų kainos po 40 Lt;
- iš 31 – mojo: reikia po 25 m^3 siųsti į "A" ir 10 m^3 siųsti į "K", reiso kaina 25*11,43+20=420 Lt, šį kartą skaičiuojam šiek tiek kitaip, kadangi sandėlis yra Kaune, tai už išsikrovimą Kaune skaičiuojame tik išsikrovimo tarifą;
- iš 32 – ojo: reikia 27 m^3 siųsti į "K", kaina 48 Lt ;
- iš 33 – ojo: reikia 15 m^3 siųsti į "K", kaina 36 Lt.

Sudėję visų tos dienos pervežimų kainas gauname bendrą pervežimų sumą, kuri lygi 23497,92Lt.

2.6.2 PRADINIO SPRENDINIO RADIMAS FOGELIO APROKCIMACIJOS METODU

Pradinė duomenų lentelė yra priede (Lentelė Nr. 7). Reikia sudaryti pasiskirstymo planą. Pagrindinė lentelės dalis nesiskiria nuo prieš tai nagrinėtų pradinio sprendinio radimo metodų lentelių. Atsiranda papildomos eilutės ir papildomi stulpeliai skirti skaičiavimo duomenims įrašyti. Šis uždavinys sprendžiamas keletu etapų, Jų skaičius neišsprendus uždavinio, iš anksto negali būti nustatytas. Visuose cikluose atliekamų operacijų turinys yra toks pat. Skaičiuojama tol, kol užpildomos visos eilutės ir stulpeliai.

Skaičiavimo tvarka tokia:

1 ciklas. Kiekvienai lentelės eilutei ir stulpeliui surandamas dviejų mažiausių vertės koeficientų skirtumas. Žiūrint priede lentelę Nr. 7 matome, eilutėse skirtumai yra trijų dydžių 550, 400 ir 300, o stulpelių mažiausių dviejų elementų skirtumai yra tokie, atitinkamiems stulpeliams:

I	II	III	IV	V	VI	VII	VIII	IX	X
100	108	116	124	132	116	148	168	176	12

Šiuos skirtumus surašome į I ciklo skaičiavimo stulpelį ir eilutę. Tuomet tame stulpelyje ir eilutėje ieškome didžiausio skirtumo, tai būtų 550, Tuomet kur buvo rastas didžiausias skirtumas pradinėje lentelės dalyje surandamas geriausio paskyrimo langelis – šiuo atveju tas, kurio vertė yra mažiausia tame stulpelyje: (1,10 langelis, iš Kauno bazės sandėlio Nr. 1 vežamas krovinys į įmonę „K“, pervežimo kaina 66 Lt). Į šį langelį įrašomas mažesnis iš dviejų skaičius – atitinkamos eilutės atsargų dydžio arba atitinkamo stulpelio poreikis. Šiuo atveju įrašyta 52 m^3, gavėjo „K“ poreikis.

Kadangi kliento „K“ poreikiai patenkinti, tai likusiuose šio stulpelio langeliuose per kitus ciklus rašyme raidę N, kuri reiškia, kad šiame stulpelyje jau nebebus daroma įrašų. Analogiškai būtų pasielgta, jei kurios nors eilutės atsargos būtų visiškai išnaudotos.

II ciklas. Skaičiavimus kartojame. Vėl susirandame eilutėse ir stulpeliuose dviejų mažiausių elementų skirtumą: eilučių skirtumai – 44, 32 ir 24, o stulpelių

I	II	III	IV	V	VI	VII	VIII	IX	X
100	108	116	124	132	116	148	168	176	N

Ties 10 – ąja įmone „K“ yra N raidė, nes šios įmonės poreikiai jau išpildyti. Dabar didžiausias skirtumas randamas eilutėje „Poreikiai“, jis lygus 176, vadinasi IX stulpelyje ieškosim mažiausio elemento ir jam priskirsim galimą reikšmę. Šiuo atveju mažiausia kaina stulpelyje įmonės „J“ poreikiai randama ties 33 – uoju sandėliu. Sandėlio atsargos 15 m^3, o į „J“ įmonės poreikis 280 m^3. Vadinasi pervežimų lentelėje priskiriame skaičių 15, o „Poreikiai“ eilutėje skaičių sumažiname 15 vienetų. Dabar N raidė atsiranda II ciklo stulpelyje ties 33 – uoju sandėliu, nes iš jo jau išvežtos visos atsargos.

Tai skaičiavimus tęsime toliau, kol paskirstome visą krovinį.

Išsprendus šį uždavinį gauti 42 ciklai. Krovinio pasiskirstymo lentelė pateikta priede Lentelė Nr. 10.

Taigi gaunamas toks pasiskirstymas:

- iš 1 – ojo: reikia 28 m^3 siųsti į “A” ir 52 m^3 siųsti į “K”. Reiso kaina skaičiuojama tokiu pat būdu, kaip ir naudojant mažiausio elemento metodo paskirstymus (Skyrius 2.6.1). Taigi šio reiso kaina lygi 28*6,88+52*0,83+30=265,80 Lt;
- iš 2 – ojo: reikia 80 m^3 siųsti į “A”, kaina 550 Lt;
- iš 3 – ojo: reikia 5 m^3 siųsti į “A” ir 75 m^3 siųsti į “B”, kaina 5*6,88+75*7,43+2,20*40=679,65Lt;
- iš 4 – ojo: reikia 2 m^3 siųsti į “B” ir 78 m^3 siųsti į “C”, kaina 2*7,43+78*7,98+2,20*270=1231,30 Lt;

- iš 5 – 11-ųjų: reikia po 80 m^3 siųsti į "C" reisų kaina po 638 Lt;
- iš 12 – ojo: reikia 65 m^3 siųsti į "C" ir 15 m^3 siųsti į "F", kaina 65*7,98+15+7,98+2,20*200=1238 Lt;
- iš 13 ir 14 – tų: reikia po 80 m^3 siųsti į "F" kaina po 638 Lt;
- iš 15 – ojo: reikia 45 m^3 siųsti į "D" ir 35 m^3 siųsti į "F", kaina 45*8,53+35*7,98+2,20*202= 1107,55 Lt;
- iš 16 ir 17 – tų: reikia po 80 m^3 siųsti į "D", kaina po 682 Lt už reisą;
- iš 18 – ojo: reikia 55 m^3 siųsti į "D" ir 25 m^3 siųsti į "E", kaina 55*8,53+25*9,08+2,20*260= 1268,15 Lt;
- iš 19 ir 20 – to: reikia po 80 m^3 siųsti į "E", kaina po 726 Lt;
- iš 21 – ojo: reikia 1 m^3 siųsti į "E" ir 79 m^3 siųsti į "G", kaina 1*9,08+10,18*79+2,20*25=868,30 Lt;
- iš 22 – ojo: reikia 80 m^3 siųsti į "G", kaina 814 Lt;
- iš 23 – ojo: reikia 57 m^3 siųsti į "F" ir 23 m^3 siųsti į "H", kaina 57*10,18+23*11,55+2,20*25= 873,11 Lt;
- iš 24 – ojo: reikia 80 m^3 siųsti į "H", kaina 924 Lt;
- iš 25 – ojo: reikia 7 m^3 siųsti į "H" ir 28 m^3 siųsti į "J", kaina 7*19,2+28*20,11+1,60*275= 1137,48 Lt;
- iš 26 – 31 – mojo: reikia po 35 m^3 siųsti į "J", reisų kaina po 704,00 Lt;
- iš 32 – ojo: reikia 27 m^3 siųsti į "J", reiso kaina 704,00 Lt;
- iš 33 – ojo: reikia 15 m^3 siųsti į "J", kaina 528,00 Lt.

Sudėję visas dienos pervežimų kainas gauname bendrą pervežimo sumą, kuri lygi 24971,34 Lt.

2.6.3 SPRENDINIŲ PALYGINIMAS

Skirtingais metodais sprendžiant gautos lapkričio mėnesio antrojo tridienio pervežimų kainos: euristiniu būdu – 20865,40 Lt, mažiausio elemento – 23497,92 Lt ir Fogelio aproksimacijos – 24971,34 Lt.

Suprantama, euristiniu būdu sprendžiant ir turėjo būti geriausias rezultatas, nes čia atsižvelgiama į automobilių parinkimą (į artimiausius miestus siunčiami mažesni automobiliai, į tolimus – didžiausi, 80 kubinių metrų talpos), į atstumus ir kainas, kombinuojami partijų vežimai tarp artimiausių miestų (tarp Vilniaus įmonių, Tauragės, Šilalės ir Klaipėdos, Vabalninko ir Biržų, bei Kupiškio ir Panevėžio), jei likę ne per didelį kiekį pervežamo krovinio, krovinys gabenamas į du gretimus miestus ta pačia transporto priemone. Šiuo metodu reisų kainos skaičiuojamos kiekvienam tridieniui, tiriamas automobilių talpos išnaudojimas pervežant krovinius.

Fogelio aproksimacijos metodu pervežimo kaina gauta didžiausia. Remiantis šio metodo paskyrimais, bendrą pervežimų kainą labai iškėlė aštuonių mažatonių automobilių siuntimas į Klaipėdą, kuri nuo Kauno sandėlių yra toliausia, vien šie reisai sudarė 5456 Lt, t.y. 21, 85% visos pervežimų kainos, o turimas pervežti krovinys viso turimo pervežti kiekio sudaro 11,42%. Be to yra keletas nenaudingų krovinio grupavimų: Šilalė – Panevėžys, Šilalė – Kupiškis ir Biržai – Klaipėda.

Remiantis krovinių paskyrimais mažiausio elemento metodu, krovinių grupavimas geresnis nei Fogelio aproksimacijos metodu. Šiuo būdu į artimiausius taškus siunčiamos mažo tūrio mašinos, gauta, kad Kauno įmonei krovinį turi pristatyti 15 ir dvi 35 kubinių metrų talpos krovininiai automobiliai.

IŠVADOS

1. Suformuluotas transporto uždavinys su fiksuotu skaičiumi skirtingo trijų tipų krovininiais automobiliais.
2. Analizuojamos ir modeliuojamos pasirinktų įmonių atsargos, jų kitimo grafikas. Gaunami skaičiai rodantys kritinę atsargų ribą ir vidutinių atsargų skaičių.
3. Tiriamas krovinių pervežimo partijų dydis ir jų kiekis.
4. Sukonstruotas transporto uždavinys mažiausios kainos atžvilgiu sprendžiamas keletu metodų: euristiniu, mažiausio metodo, Fogelio aproksimacijos.
5. Mažiausia vieno tridienio kaina gauta sprendžiant euristiniu metodu, pervežimų kaina 20865,40Lt kai numatytas pervežti krovinių kiekis yra 2207 m^3.
6. Euristiniu metodu skaičiuojant, gauta, kad kroviniai užima 94% automobilyje kroviniams skirto tūrio.
7. Mažiausio elemento metodu gauta pervežimo kaina yra mažesnė nei Fogelio aproksimacijos metodu.

LITERATŪRA

1. Baublys A. Keleivių ir krovinių vežimas kelių transportu. Vilnius: Technika, 1994, 171 p.
2. Batarlienė N. Transporto uždavinių matematinis modeliavimas. Vilnius: Technika, 2000. 107 p.
3. Ratkienė N. Matematinis programavimas. Kaunas: Technologija, 1992. 216 p.
4. Palšaitis R. Logistikos vadybos pagrindai. Vilnius: Technika, 355 p.
5. Ballou, Ronald H. Business logistics management. Prentice-Hall International, 1992. 688 p
6. Straipsnis. Andreas Westerlund. A Column Generation Scheme for the Fixed Fleet Heterogeneous Vehicle Routing Problem. Maud Göthe–Lundgren and Torbjörn Larsson Department of Mathematics Linköping University. 2005.
7. Fuh – Hwa Liu and Sheng – Yuan Shen. A Method for Vehicle Routing Problem with Multiple Vehicle Types and Time Windows. Proc. Natl. Sci. Counc. ROC(A) Vol. 23, No. 4, 1999. pp. 526-536.

Juseviciene K., Modeling of load flows in clique logistic system, Postgraduate's work, supervisor doc. dr. G. Rackauskas, Fundamental science faculty, department of mathematics, Kaunas university of technology, 2006, 69 p.

SUMMARY

We present an optimization procedure for solving the vehicle routing problem with a fixed heterogeneous fleet of vehicle. We want to minimize the passage price. We look and probe these methods: North west corner, minimal element, Vogel's Approximation and heuristic. The modeling vehicle routing problem is based on mathematical formulation. This paper present very well known problems – TSP Traveling Salesperson Problem and M-TSP. Vehicle routing problem is liked M-TSP with some specification, vehicle with a fixed carrying capacity must deliver order of goods to n customers from a single depot. Knowing the distance between customers, the problem is to find tours for the vehicles in such a way that:

the total distance traveled by the vehicles is minimized,

only one vehicle handles the deliveries for a given customer,

the total quantity of goods that a single vehicle delivers cannot be larger than cars capacity.

PRIEDAS

1 Pav. Darbe tiriamų bazių – sandėlių ir parduotuvių – gavėjų išsidėstymas

Lentelė Nr.1

Pervežimų partijos dydžiai

Eil. Nr.	Gavėjas	Pervežimo partijų dydžiai				Pervežimų skaičius bent kartą į __ dieną(nas)	Pervežimų skaičius M
		Lapkričio mėn.	Gruodžio mėn.	Sausio mėn.	Vasario mėn.		
1	"A" Vilnius	303	304	327	442	1	10
2	"B" Vilnius	809	1271	1464	1629	5	2
3	"C" Tauragė	5954	2244	2884	2907	10	1

4	"D" Šilalė	353	380	376	526	2	5
5	"E" Kupiškis	1326	1519	1122	1337	5	2
6	"F" Panevėžys	1499	2463	1016	1697	5	2
7	"G" Vabalninkas	929	750	1297	1003	5	2
8	"H" Biržai	599	170	161	149	2	5
9	"J" Klaipėda	1197	1019	1036	651	5	2
10	"K" Kaunas	259	398	210	153	2	5

Lentelė Nr. 2

Lapkričio mėnesio užsakymų lentelė

	Mėnuo	Lapkritis									
	Tridienis	1	2	3	4	5	6	7	8	9	10
Eil. Nr.	Gavėjas										
1	"A" Vilnius	270	113	333	186	326	303	320	343	400	433
2	"B" Vilnius	300	77	107	143	223	150	130	183	120	186
3	"C" Tauragė	664	703	589	626	589	565	543	516	493	663
4	"D" Šilalė	150	260	97	186	77	100	90	223	260	323
5	"E" Kupiškis	115	186	113	260	186	259	296	360	406	466
6	"F" Panevėžys	80	210	260	163	446	476	569	296	320	180
7	"G" Vabalninkas	110	216	180	163	143	253	100	0	433	260
8	"H" Biržai	186	110	703	330	410	186	33	216	599	223
9	"J" Klaipėda	79	280	410	113	77	766	0	186	150	333
10	"K" Kaunas	113	52	132	216	133	193	0	228	147	78
Krovinių suma		2067	2207	2924	2386	2610	3251	2081	2551	3328	3145

Lentelė Nr. 3

Gruodžio mėnesio užsakymų lentelė

	Mėnuo	Gruodis									
	Tridienis	1	2	3	4	5	6	7	8	9	10
Eil. Nr.	Gavėjas										
1	"A" Vilnius	150	326	886	186	186	253	223	260	276	296
2	"B" Vilnius	123	253	153	250	160	153	263	296	450	440
3	"C" Tauragė	213	150	77	163	776	223	157	190	157	140
4	"D" Šilalė	230	276	509	293	153	226	80	80	43	7
5	"E" Kupiškis	513	596	83	286	83	473	250	330	13	410
6	"F" Panevėžys	699	663	729	746	150	120	809	410	260	340
7	"G" Vabalninkas	263	157	77	17	120	196	113	160	0	396
8	"H" Biržai	330	117	90	43	77	113	40	40	0	0
9	"J" Klaipėda	330	90	236	75	183	173	246	210	17	476
10	"K" Kaunas	100	162	43	43	241	198	296	50	47	814
Krovinių suma		2951	2790	2883	2102	2129	2128	2477	2026	1263	3319

Lentelė Nr. 4

Sausio mėnesio užsakymų lentelė

	Mėnuo	Sausis									
	Tridienis	1	2	3	4	5	6	7	8	9	10
Eil. Nr.	Gavėjas										
1	"A" Vilnius	263	443	150	223	286	356	426	493	223	410
2	"B" Vilnius	157	556	190	253	316	286	410	163	186	410
3	"C" Tauragė	223	0	173	107	356	769	260	326	223	446
4	"D" Šilalė	143	107	123	296	153	150	53	263	270	323
5	"E" Kupiškis	113	330	213	163	123	186	410	153	256	296
6	"F" Panevėžys	350	233	196	163	330	316	77	140	77	150
7	"G" Vabalninkas	519	263	260	193	127	107	117	483	260	266
8	"H" Biržai	77	113	186	113	40	150	40	13	10	63
9	"J" Klaipėda	743	147	117	77	47	40	147	649	40	67
10	"K" Kaunas	69	141	150	210	44	39	122	227	28	20
Krovinių suma		2657	2333	1758	1798	1822	2399	2062	2910	1573	2451

Lentelė Nr. 5

Vasario mėnesio užsakymų lentelė

	Mėnuo	Vasaris									
	Tridienis	1	2	3	4	5	6	7	8	9	10
Eil. Nr.	Gavėjas										
1	"A" Vilnius	426	80	350	296	456	566	673	519	539	516
2	"B" Vilnius	300	296	376	483	410	296	340	263	293	200
3	"C" Tauragė	153	396	882	223	260	263	153	127	190	260
4	"D" Šilalė	663	77	316	226	326	130	223	150	296	223
5	"E" Kupiškis	403	226	283	376	263	413	77	67	223	343
6	"F" Panevėžys	113	663	117	410	410	453	113	516	446	153
7	"G" Vabalninkas	153	413	80	186	260	223	150	343	157	40
8	"H" Biržai	43	113	73	40	150	30	77	113	30	77
9	"J" Klaipėda	77	223	173	220	186	30	63	273	20	37
10	"K" Kaunas	58	63	51	295	19	17	77	153	21	12
Krovinių suma		2389	2550	2701	2755	2740	2421	1946	2524	2215	1861

Lentelė Nr. 6

Atstumų lentel ė

Įmonės	"A" Vilnius	"B" Vilnius	"C" Tauragė	"D" Šilalė	"E" Kupiškis	"F" Panevėžys	"G" Vabalninkas	"H" Biržai	"J" Klaipėda	"K" Kaunas	Kauno bazė
"A" Vilnius	/	40	240	243	165	145	182	210	310	115	125
"B" Vilnius	40	/	270	260	175	160	195	220	330	130	135
"C" Tauragė	240	270	/	35	250	200	240	210	113	120	145
"D" Šilalė	243	260	35	/	260	202	245	210	81	150	155
"E" Kupiškis	165	175	250	260	/	51	25	42	317	160	165
"F" Panevėžys	145	160	200	202	51	/	40	68	270	140	145
"G" Vabalninkas	182	195	240	245	25	40	/	25	310	180	185
"H" Biržai	210	220	210	210	42	68	25	/	275	205	210
"J" Klaipėda	310	330	113	81	160	270	310	275	/	215	220
"K" Kaunas	115	130	120	150	165	140	180	205	215	/	15
Kauno bazė	125	135	145	155	165	145	185	210	220	15	/

Lentelė Nr. 7

Lapkričio II tridienio duomenų matricos lentelė

Tiekimo punktai	Gavėjai										Atsargos, m^3
	"A" Vilnius	"B" Vilnius	"C" Tauragė	"D" Šilalė	"E" Kupiškis	"F" Panevėžys	"G" Vabalninkas	"H" Biržai	"J" Klaipėda	"K" Kaunas	
1	550	594	638	682	726	638	814	924	968	66	80
2	550	594	638	682	726	638	814	924	968	66	80
3	550	594	638	682	726	638	814	924	968	66	80
4	550	594	638	682	726	638	814	924	968	66	80
5-17	550	594	638	682	726	638	814	924	968	66	80
18	550	594	638	682	726	638	814	924	968	66	80
19	550	594	638	682	726	638	814	924	968	66	80
20	550	594	638	682	726	638	814	924	968	66	80
21	550	594	638	682	726	638	814	924	968	66	80
22	550	594	638	682	726	638	814	924	968	66	80
23	550	594	638	682	726	638	814	924	968	66	80
24	550	594	638	682	726	638	814	924	968	66	80
25	400	432	464	496	528	464	592	672	704	48	35
26	400	432	464	496	528	464	592	672	704	48	35
27	400	432	464	496	528	464	592	672	704	48	35
28	400	432	464	496	528	464	592	672	704	48	35
29	400	432	464	496	528	464	592	672	704	48	35
30	400	432	464	496	528	464	592	672	704	48	35
31	400	432	464	496	528	464	592	672	704	48	35
32	400	432	464	496	528	464	592	672	704	48	27
33	300	324	348	372	396	348	444	504	528	36	15
Poreikis, m^3	113	77	703	260	186	210	216	110	280	52	2207

Mažiausio elemento metodu gautas pradinis sprendinys **Lentelė Nr. 8**

Tiekimo punktai	Gavėjai										Atsargos m^3
	"A" Vilnius	"B" Vilnius	"C" Tauragė	"D" Šilalė	"E" Kupiškis	"F" Panevėžys	"G" Vabalninkas	"H" Biržai	"J" Klaipėda	"K" Kaunas	
1	0	0	0	0	0	0	0	0	80	0	80
2	0	0	0	0	0	0	0	0	80	0	80
3	0	0	0	0	0	0	0	0	80	0	80
4	0	0	0	0	0	0	0	40	40	0	80
5	0	0	0	0	0	0	10	70	0	0	80
6	0	0	0	0	0	0	80	0	0	0	80
7	0	0	0	0	0	0	80	0	0	0	80
8	0	0	0	0	34	0	46	0	0	0	80
9	0	0	0	0	80	0	0	0	0	0	80
10	0	0	0	8	72	0	0	0	0	0	80
11	0	0	0	80	0	0	0	0	0	0	80
12	0	0	0	80	0	0	0	0	0	0	80
13	0	0	0	80	0	0	0	0	0	0	80
14	0	0	0	12	0	68	0	0	0	0	80
15	0	0	0	0	0	80	0	0	0	0	80
16	0	0	18	0	0	62	0	0	0	0	80
17	0	0	80	0	0	0	0	0	0	0	80
18	0	0	80	0	0	0	0	0	0	0	80
19	0	0	80	0	0	0	0	0	0	0	80
20	0	0	80	0	0	0	0	0	0	0	80
21	0	0	80	0	0	0	0	0	0	0	80
22	0	0	80	0	0	0	0	0	0	0	80
23	0	0	80	0	0	0	0	0	0	0	80
24	0	0	80	0	0	0	0	0	0	0	80
25	0	0	35	0	0	0	0	0	0	0	35
26	0	25	10	0	0	0	0	0	0	0	35
27	0	35	0	0	0	0	0	0	0	0	35
28	18	17	0	0	0	0	0	0	0	0	35
29	35	0	0	0	0	0	0	0	0	0	35

30	35	0	0	0	0	0	0	0	0	0	35
31	25	0	0	0	0	0	0	0	0	10	35
32	0	0	0	0	0	0	0	0	0	27	27
33	0	0	0	0	0	0	0	0	0	15	15
Poreikis, m^3	113	77	703	260	186	210	216	110	280	52	2207

Vieno pervežamo kubinio metro talpos krovinio kaina litais **Lentelė Nr. 9**

Tiekimo punktas, Automobilis	Gavėjai									
	A Vilnius	B Vilnius	C Tauragė	D Šilalė	E Kupiškis	F Panevėžys	G Vabalninkas	H Biržai	J Klaipėda	K Kaunas
1	6.88	7.43	7.98	8.53	9.08	7.98	10.18	11.55	12.10	0.83
2	6.88	7.43	7.98	8.53	9.08	7.98	10.18	11.55	12.10	0.83
3	6.88	7.43	7.98	8.53	9.08	7.98	10.18	11.55	12.10	0.83
4	6.88	7.43	7.98	8.53	9.08	7.98	10.18	11.55	12.10	0.83
5	6.88	7.43	7.98	8.53	9.08	7.98	10.18	11.55	12.10	0.83
6	6.88	7.43	7.98	8.53	9.08	7.98	10.18	11.55	12.10	0.83
7	6.88	7.43	7.98	8.53	9.08	7.98	10.18	11.55	12.10	0.83
8	6.88	7.43	7.98	8.53	9.08	7.98	10.18	11.55	12.10	0.83
9	6.88	7.43	7.98	8.53	9.08	7.98	10.18	11.55	12.10	0.83
10	6.88	7.43	7.98	8.53	9.08	7.98	10.18	11.55	12.10	0.83
11	6.88	7.43	7.98	8.53	9.08	7.98	10.18	11.55	12.10	0.83
12	6.88	7.43	7.98	8.53	9.08	7.98	10.18	11.55	12.10	0.83
13	6.88	7.43	7.98	8.53	9.08	7.98	10.18	11.55	12.10	0.83
14	6.88	7.43	7.98	8.53	9.08	7.98	10.18	11.55	12.10	0.83
15	6.88	7.43	7.98	8.53	9.08	7.98	10.18	11.55	12.10	0.83
16	6.88	7.43	7.98	8.53	9.08	7.98	10.18	11.55	12.10	0.83
17	6.88	7.43	7.98	8.53	9.08	7.98	10.18	11.55	12.10	0.83
18	6.88	7.43	7.98	8.53	9.08	7.98	10.18	11.55	12.10	0.83
19	6.88	7.43	7.98	8.53	9.08	7.98	10.18	11.55	12.10	0.83
20	6.88	7.43	7.98	8.53	9.08	7.98	10.18	11.55	12.10	0.83
21	6.88	7.43	7.98	8.53	9.08	7.98	10.18	11.55	12.10	0.83
22	6.88	7.43	7.98	8.53	9.08	7.98	10.18	11.55	12.10	0.83
23	6.88	7.43	7.98	8.53	9.08	7.98	10.18	11.55	12.10	0.83
24	6.88	7.43	7.98	8.53	9.08	7.98	10.18	11.55	12.10	0.83
25	11.43	12.34	13.26	14.17	15.09	13.26	16.91	19.20	20.11	1.37
26	11.43	12.34	13.26	14.17	15.09	13.26	16.91	19.20	20.11	1.37
27	11.43	12.34	13.26	14.17	15.09	13.26	16.91	19.20	20.11	1.37

28	11.43	12.34	13.26	14.17	15.09	13.26	16.91	19.20	20.11	1.37
29	11.43	12.34	13.26	14.17	15.09	13.26	16.91	19.20	20.11	1.37
30	11.43	12.34	13.26	14.17	15.09	13.26	16.91	19.20	20.11	1.37
31	11.43	12.34	13.26	14.17	15.09	13.26	16.91	19.20	20.11	1.37
32	14.81	16.00	17.19	18.37	19.56	17.19	21.93	24.89	26.07	1.78
33	20.00	21.60	23.20	24.80	26.40	23.20	29.60	33.60	35.20	2.40

Lentelė Nr. 10 Fogelio aproksimacijos metodu gautas pradinis sprendinys

Tiekimo punktai	Gavėjai										Atsargos m³
	"A" Vilnius	"B" Vilnius	"C" Tauragė	"D" Šilalė	"E" Kupiškis	"F" Panevėžys	"G" Vabalninkas	"H" Biržai	"J" Klaipėda	"K" Kaunas	
1	28	0	0	0	0	0	0	0	0	52	80
2	80	0	0	0	0	0	0	0	0	0	80
3	5	75	0	0	0	0	0	0	0	0	80
4	0	2	78	0	0	0	0	0	0	0	80
5	0	0	80	0	0	0	0	0	0	0	80
6	0	0	80	0	0	0	0	0	0	0	80
7	0	0	80	0	0	0	0	0	0	0	80
8	0	0	80	0	0	0	0	0	0	0	80
9	0	0	80	0	0	0	0	0	0	0	80
10	0	0	80	0	0	0	0	0	0	0	80
11	0	0	80	0	0	0	0	0	0	0	80
12	0	0	65	0	0	15	0	0	0	0	80
13	0	0	0	0	0	80	0	0	0	0	80
14	0	0	0	0	0	80	0	0	0	0	80
15	0	0	0	45	0	35	0	0	0	0	80
16	0	0	0	80	0	0	0	0	0	0	80
17	0	0	0	80	0	0	0	0	0	0	80
18	0	0	0	55	25	0	0	0	0	0	80
19	0	0	0	0	80	0	0	0	0	0	80
20	0	0	0	0	80	0	0	0	0	0	80
21	0	0	0	0	1	0	79	0	0	0	80
22	0	0	0	0	0	0	80	0	0	0	80
23	0	0	0	0	0	0	57	23	0	0	80
24	0	0	0	0	0	0	0	80	0	0	80
25	0	0	0	0	0	0	0	7	28	0	35
26	0	0	0	0	0	0	0	0	35	0	35
27	0	0	0	0	0	0	0	0	35	0	35
28	0	0	0	0	0	0	0	0	35	0	35

29	0	0	0	0	0	0	0	0	35	0	35
30	0	0	0	0	0	0	0	0	35	0	35
31	0	0	0	0	0	0	0	0	35	0	35
32	0	0	0	0	0	0	0	0	27	0	27
33	0	0	0	0	0	0	0	0	15	0	15
Poreikis, m^3	113	77	703	260	186	210	216	110	280	52	2207

1. Mažiausio elemento metodo programos tekstas

```
#include <vcl.h>
#pragma hdrstop
#include "Programos_langas.h"
//---------------------------------------------------------------------------
#pragma package(smart_init)
#pragma resource "*.dfm"
TLangas *Langas;
//---------------------------------------------------------------------------
__fastcall TLangas::TLangas(TComponent* Owner) : TForm(Owner){
   Nunulinimas();
};
//---------------------------------------------------------------------------
void TLangas::Nunulinimas(){
   n = 0;
   m = 0;
   Pasiula = NULL;
   Paklausa = NULL;
   Kainu_Matrica = NULL;
   Srautai = NULL;
   Nepanaudotos_Atsargos = NULL;
   Nepatenkinti_Poreikiai = NULL;
   Reiksmiu_Kiekis = 0;
   Reiksmes = NULL;
};
//---------------------------------------------------------------------------
void TLangas::Duomenu_Nuskaitymas(){
   FILE *F;
   char stringas[20];
   bool Pozymis;
   int l, temp;

   if((F = fopen(Atidaryti_Faila->FileName.c_str(), "rt")) != NULL){
      fscanf(F, "%i %i\n", &n, &m);
      Pasiula = (int *)malloc(sizeof(int) * n);
      Paklausa = (int *)malloc(sizeof(int) * m);
      Nepanaudotos_Atsargos = (int *)malloc(sizeof(int) * n);
      Nepatenkinti_Poreikiai = (int *)malloc(sizeof(int) * m);
      Kainu_Matrica = (int **)malloc(sizeof(int *) * n);
      Srautai = (int **)malloc(sizeof(int *) * n);
      for(int i = 0; i < n; i++){
         Kainu_Matrica[i] = (int *)malloc(sizeof(int) * m);
         Srautai[i] = (int *)malloc(sizeof(int) * m);
      };
      for(int i = 0; i < m; i++){
         fscanf(F, "%s", stringas);
         Paklausa[i] = atoi(stringas);
         Nepatenkinti_Poreikiai[i] = Paklausa[i];
      };
      for(int i = 0; i < n; i++){
         fscanf(F, "%s", stringas);
         Pasiula[i] = atoi(stringas);
         Nepanaudotos_Atsargos[i] = Pasiula[i];
         for(int j = 0; j < m; j++){
            fscanf(F, "%s", stringas);
            Kainu_Matrica[i][j] = atoi(stringas);
            Srautai[i][j] = 0;
         };
      };
      fclose(F);
   };
   for(int i = 0; i < n; i++)
      for(int j = 0; j < m; j++){
         Pozymis = false;
         for(int k = 0; k < Reiksmiu_Kiekis; k++)
            if(Reiksmes[k] == Kainu_Matrica[i][j])
               Pozymis = true;
```

```
            if(!Pozymis){
                Reiksmiu_Kiekis++;
                Reiksmes = (int *)realloc(Reiksmes, sizeof(int) * Reiksmiu_Kiekis);
                Reiksmes[Reiksmiu_Kiekis - 1] = Kainu_Matrica[i][j];
            }
        }
    l = 0;
    while(l < Reiksmiu_Kiekis - 1)
        if(Reiksmes[l] < Reiksmes[l + 1])
            l++;
        else{
            temp = Reiksmes[l];
            Reiksmes[l] = Reiksmes[l + 1];
            Reiksmes[l + 1] = temp;
            if(l > 0)
                l--;
        }

};
//---------------------------------------------------------------------------
void TLangas::Surasti_Minimumus(int *Kiekis, int **X_koordinate, int **Y_koordinate){
    int k;
    bool Stop;


    *Kiekis = 0;
    *X_koordinate = NULL;
    *Y_koordinate = NULL;
    Stop = false;
    k = 0;
    while((k < Reiksmiu_Kiekis) && (!Stop)){
        for(int i = 0; i < n; i++)
            for(int j = 0; j < m; j++)
                if(Kainu_Matrica[i][j] == Reiksmes[k]){
                    if((Nepanaudotos_Atsargos[i] != 0) && (Nepatenkinti_Poreikiai[j] != 0)){
                        Stop = true;
                        *Kiekis += 1;
                        *X_koordinate = (int *) realloc (*X_koordinate, sizeof(int) * (*Kiekis));
                        *Y_koordinate = (int *) realloc (*Y_koordinate, sizeof(int) * (*Kiekis));
                        (*X_koordinate)[*Kiekis - 1] = i;
                        (*Y_koordinate)[*Kiekis - 1] = j;
                    }
                }
        k++;
    }
};
//---------------------------------------------------------------------------
AnsiString TLangas::Sveikas_Skaicius(int Sk, int Plotis){
    AnsiString S = IntToStr(Sk);
    while(S.Length() < Plotis)
        S += " ";
    return S;
};
//---------------------------------------------------------------------------
int TLangas::Suma(){
    int S = 0;

    for(int i = 0; i < n; i++)
        for(int j = 0; j < m; j++)
           S += Kainu_Matrica[i][j] * Srautai[i][j];
    return S;
};
//---------------------------------------------------------------------------
void TLangas::Atspaudinti(int kiekis, int *Eilute, int *Stulpelis, int iteracija){
    AnsiString Eil, S;
    bool temp;

    Tekstas->Lines->Add("     " + IntToStr(iteracija) + " iteracija:");
```

```
    Eil = "+--------------+";
    for(int j = 0; j < m - 1; j++)
        Eil += "------------";
    Tekstas->Lines->Add(Eil + "-----------+-------------+------------------+");

    Eil = "| Sandeliavimo | Vartojimo punktai     ";

    for(int j = 0; j < m - 2; j++)
        Eil += "            ";
    Tekstas->Lines->Add(Eil + "|             | Dar nepanaudotos |");

    Eil = "|              ";
    for(int j = 0; j < m; j++)
        Eil += "+-----------";
    Tekstas->Lines->Add(Eil + "+  Atsargos   |                  |");

    Eil = "|    punktai   ";
    for(int j = 0; j < m; j++)
        Eil += "|      W" + Sveikas_Skaicius(j + 1, 4);
    Tekstas->Lines->Add(Eil + "|             |     atsargos     |");

    Eil = "+--------------";
    for(int j = 0; j < m; j++)
        Eil += "+-----------";
    Tekstas->Lines->Add(Eil + "+-------------+------------------+");

    for(int i = 0; i < n; i++){
/*        */
        Eil = "|              ";
        for(int j = 0; j < m; j++){
            temp = false;
            for(int k = 0; k < kiekis; k++)
                if((Eilute[k] == i) && (Stulpelis[k] == j))
                    temp = true;
            if(temp)
                Eil += "|*****| " + Sveikas_Skaicius(Kainu_Matrica[i][j], 4);
            else
                Eil += "|     | " + Sveikas_Skaicius(Kainu_Matrica[i][j], 4);
        }
        Tekstas->Lines->Add(Eil + "|             |                  |");

        Eil = "|      F" + Sveikas_Skaicius(i + 1, 4) + "   ";
        for(int j = 0; j < m; j++)
            Eil += "+     +-----";
        Tekstas->Lines->Add(Eil + "+   " + Sveikas_Skaicius(Pasiula[i], 10) + "|        " + Sveikas_Skaicius(Nepanaudotos_Atsargos[i], 
10) + "|");
        Eil = "|              ";
        for(int j = 0; j < m; j++)
            if(Srautai[i][j])
                Eil += "| " + Sveikas_Skaicius(Srautai[i][j], 10);
            else
                Eil += "|    ---    ";
        Tekstas->Lines->Add(Eil + "|             |                  |");
        Eil = "+--------------";
        for(int j = 0; j < m; j++)
            Eil += "+-----------";
        Tekstas->Lines->Add(Eil + "+-------------+------------------+");
    };

    Eil = "|  Poreikiai   ";
    for(int j = 0; j < m; j++)
        Eil += "|   " + Sveikas_Skaicius(Paklausa[j], 8);
    Tekstas->Lines->Add(Eil + "|");

    Eil = "+--------------";
    for(int j = 0; j < m; j++)
        Eil += "+-----------";
```

```
   Tekstas->Lines->Add(Eil + "|   Suma = " + IntToStr(Suma()));

   Eil = "| Nepatenkinti ";
   for(int j = 0; j < m; j++)
      Eil += "|   " + Sveikas_Skaicius(Nepatenkinti_Poreikiai[j], 8);
   Tekstas->Lines->Add(Eil + "|");

   Eil = "+--------------";
   for(int j = 0; j < m; j++)
      Eil += "+-----------";
   Tekstas->Lines->Add(Eil + "|");

   Tekstas->Lines->Add("");
   Tekstas->Lines->Add("");
};
//---------------------------------------------------------------------------
void TLangas::Skaiciavimas(){
   int Iteracijos_Numeris;
   int Minimumu_Kiekis;
   int* Minimumo_Stulpeliai;
   int* Minimumo_Eilutes;

   Minimumu_Kiekis = -1;
   Iteracijos_Numeris = 0;
   while(Minimumu_Kiekis != 0){
      Surasti_Minimumus(&Minimumu_Kiekis, &Minimumo_Eilutes, &Minimumo_Stulpeliai);
      if(Minimumu_Kiekis != 0){
         for(int i = 0; i < Minimumu_Kiekis; i++)
            if(Nepanaudotos_Atsargos[Minimumo_Eilutes[i]] > Nepatenkinti_Poreikiai[Minimumo_Stulpeliai[i]]){
               Srautai[Minimumo_Eilutes[i]][Minimumo_Stulpeliai[i]] = Nepatenkinti_Poreikiai[Minimumo_Stulpeliai[i]];
               Nepanaudotos_Atsargos[Minimumo_Eilutes[i]] -= Nepatenkinti_Poreikiai[Minimumo_Stulpeliai[i]];
               Nepatenkinti_Poreikiai[Minimumo_Stulpeliai[i]] = 0;
            }
            else{
               Srautai[Minimumo_Eilutes[i]][Minimumo_Stulpeliai[i]] = Nepanaudotos_Atsargos[Minimumo_Eilutes[i]];
               Nepatenkinti_Poreikiai[Minimumo_Stulpeliai[i]] -= Nepanaudotos_Atsargos[Minimumo_Eilutes[i]];
               Nepanaudotos_Atsargos[Minimumo_Eilutes[i]] = 0;
            }
      }
      Atspaudinti(Minimumu_Kiekis, Minimumo_Eilutes, Minimumo_Stulpeliai, ++Iteracijos_Numeris);
      free(Minimumo_Eilutes);
      free(Minimumo_Stulpeliai);
   }
   Tekstas->Lines->SaveToFile("Rezultatai.txt");
   free(Pasiula);
   free(Paklausa);
   for(int i = 0; i < n; i++)
      free(Kainu_Matrica[i]);
   free(Kainu_Matrica);
   for(int i = 0; i < n; i++)
      free(Srautai[i]);
   free(Srautai);
   free(Nepanaudotos_Atsargos);
   free(Nepatenkinti_Poreikiai);
   free(Reiksmes);
};
//---------------------------------------------------------------------------
void __fastcall TLangas::TekstasClick(TObject *Sender){
   if((n == 0) || (m == 0))
      if(Atidaryti_Faila->Execute()){
         Duomenu_Nuskaitymas();
         Skaiciavimas();
      };
};
```

2. Fogelio aproksimacijos metodo programos tekstas

```
#include <vcl.h>
#pragma hdrstop
```

```
#include "Programos_langas.h"
//---------------------------------------------------------------------------
#pragma package(smart_init)
#pragma resource "*.dfm"
TLangas *Langas;
//---------------------------------------------------------------------------
__fastcall TLangas::TLangas(TComponent* Owner) : TForm(Owner){
    Nunulinimas();
};
//---------------------------------------------------------------------------
void TLangas::Nunulinimas(){
    n = 0;
    m = 0;
    Pasiula = NULL;
    Paklausa = NULL;
    Kainu_Matrica = NULL;
    Srautai = NULL;
    Nepanaudotos_Atsargos = NULL;
    Nepatenkinti_Poreikiai = NULL;
    Horizontali_Matrica = NULL;
    Vertikali_Matrica = NULL;
    Pasirinkti_Horizontali_Matrica = NULL;
    Pasirinkti_Vertikali_Matrica = NULL;
};
//---------------------------------------------------------------------------
void TLangas::Duomenu_Nuskaitymas(){
    FILE *F;
    char stringas[20];
    bool Pozymis;
    int l, temp;

    if((F = fopen(Atidaryti_Faila->FileName.c_str(), "rt")) != NULL){
        fscanf(F, "%i %i\n", &n, &m);
        Pasiula = (int *)malloc(sizeof(int) * n);
        Paklausa = (int *)malloc(sizeof(int) * m);
        Nepanaudotos_Atsargos = (int *)malloc(sizeof(int) * n);
        Nepatenkinti_Poreikiai = (int *)malloc(sizeof(int) * m);
        Kainu_Matrica = (int **)malloc(sizeof(int *) * n);
        Srautai = (int **)malloc(sizeof(int *) * n);
        for(int i = 0; i < n; i++){
            Kainu_Matrica[i] = (int *)malloc(sizeof(int) * m);
            Srautai[i] = (int *)malloc(sizeof(int) * m);
        };
        for(int i = 0; i < m; i++){
            fscanf(F, "%s", stringas);
            Paklausa[i] = atoi(stringas);
            Nepatenkinti_Poreikiai[i] = Paklausa[i];
        };
        for(int i = 0; i < n; i++){
            fscanf(F, "%s", stringas);
            Pasiula[i] = atoi(stringas);
            Nepanaudotos_Atsargos[i] = Pasiula[i];
            for(int j = 0; j < m; j++){
                fscanf(F, "%s", stringas);
                Kainu_Matrica[i][j] = atoi(stringas);
                Srautai[i][j] = 0;
            };
        };
        fclose(F);
    };
};
//---------------------------------------------------------------------------
void TLangas::Pradines_Matricos(){
    int *Laikinas_Masyvas;
    int Masyvo_nariu_kiekis;
    int i;
    int j;
    int k;
```

```
    int temp;
    bool yra;

    iteracija = 1;
    Horizontali_Matrica = (int **)malloc(sizeof(int *) * n);
    Masyvo_nariu_kiekis = 0;
    Laikinas_Masyvas = NULL;
    for(i = 0; i < n; i++){
        for(j = 0; j < m; j++){
            yra = false;
            for(k = 0; k < Masyvo_nariu_kiekis; k++)
                if(Laikinas_Masyvas[k] == Kainu_Matrica[i][j])
                    yra = true;
            if(!yra){
                Masyvo_nariu_kiekis++;
                Laikinas_Masyvas = (int *)realloc(Laikinas_Masyvas, sizeof(int) * Masyvo_nariu_kiekis);
                Laikinas_Masyvas[Masyvo_nariu_kiekis - 1] = Kainu_Matrica[i][j];

            }
        }

        j = 0;
        while(j < Masyvo_nariu_kiekis - 1)
            if(Laikinas_Masyvas[j] <= Laikinas_Masyvas[j + 1])
                j++;
            else{
                temp = Laikinas_Masyvas[j];
                Laikinas_Masyvas[j] = Laikinas_Masyvas[j + 1];
                Laikinas_Masyvas[j + 1] = temp;
                if(j > 0)
                    j--;
            }
        Horizontali_Matrica[i] = (int *)malloc(sizeof(int) * iteracija);
        Horizontali_Matrica[i][iteracija - 1] = Laikinas_Masyvas[1] - Laikinas_Masyvas[0];
    }
    free(Laikinas_Masyvas);
    Vertikali_Matrica = (int **)malloc(sizeof(int *) * m);
    Masyvo_nariu_kiekis = 0;
    Laikinas_Masyvas = NULL; (int*)malloc(sizeof(int) * Masyvo_nariu_kiekis);
    for(j = 0; j < m; j++){
        for(i = 0; i < n; i++){
            yra = false;
            for(k = 0; k < Masyvo_nariu_kiekis; k++)
                if(Laikinas_Masyvas[k] == Kainu_Matrica[i][j])
                    yra = true;
            if(!yra){
                Masyvo_nariu_kiekis++;
                Laikinas_Masyvas = (int *)realloc(Laikinas_Masyvas, sizeof(int) * Masyvo_nariu_kiekis);
                Laikinas_Masyvas[Masyvo_nariu_kiekis - 1] = Kainu_Matrica[i][j];
            }
        }

        i = 0;
        while(i < Masyvo_nariu_kiekis - 1)
            if(Laikinas_Masyvas[i] <= Laikinas_Masyvas[i + 1])
                i++;
            else{
                temp = Laikinas_Masyvas[i];
                Laikinas_Masyvas[i] = Laikinas_Masyvas[i + 1];
                Laikinas_Masyvas[i + 1] = temp;
                if(i > 0)
                    i--;
            }
        Vertikali_Matrica[j] = (int *)malloc(sizeof(int) * iteracija);
        Vertikali_Matrica[j][iteracija - 1] = Laikinas_Masyvas[1] - Laikinas_Masyvas[0];
    }
    free(Laikinas_Masyvas);
};
```

```
//---------------------------------------------------------------------------
void TLangas::Matricu_Perskaiciavimas(bool Horizontali_Matrica_Pozymis, int indeksas){
  int *Laikinas_Masyvas;
  int Masyvo_nariu_kiekis;
  int i;
  int j;
  int k;
  int temp;
  bool yra;

  iteracija++;
  for(i = 0; i < n; i++){
    Masyvo_nariu_kiekis = 0;
    Laikinas_Masyvas = NULL;
    for(j = 0; j < m; j++)
      if((Nepanaudotos_Atsargos[i] != 0) && (Nepatenkinti_Poreikiai[j] != 0)){
        yra = false;
        for(k = 0; k < Masyvo_nariu_kiekis; k++)
          if(Laikinas_Masyvas[k] == Kainu_Matrica[i][j])
            yra = true;
        if(!yra){
          Masyvo_nariu_kiekis++;
          Laikinas_Masyvas = (int *)realloc(Laikinas_Masyvas, sizeof(int) * Masyvo_nariu_kiekis);
          Laikinas_Masyvas[Masyvo_nariu_kiekis - 1] = Kainu_Matrica[i][j];
        }
      }
    j = 0;
    while(j < Masyvo_nariu_kiekis - 1)
      if(Laikinas_Masyvas[j] <= Laikinas_Masyvas[j + 1])
        j++;
      else{
        temp = Laikinas_Masyvas[j];
        Laikinas_Masyvas[j] = Laikinas_Masyvas[j + 1];
        Laikinas_Masyvas[j + 1] = temp;
        if(j > 0)
          j--;
      }
    Horizontali_Matrica[i] = (int *)realloc(Horizontali_Matrica[i], sizeof(int) * iteracija);
    if((Horizontali_Matrica_Pozymis == true) && (indeksas == i) && (Nepanaudotos_Atsargos[i] == 0)){
      Horizontali_Matrica[i][iteracija - 1] = -1;
    }
    else
      if(Horizontali_Matrica[i][iteracija - 2] == -1)
        Horizontali_Matrica[i][iteracija - 1] = -1;
      else
        if(Masyvo_nariu_kiekis > 1)
          Horizontali_Matrica[i][iteracija - 1] = Laikinas_Masyvas[1] - Laikinas_Masyvas[0];
        else
          if(Masyvo_nariu_kiekis == 1)
            Horizontali_Matrica[i][iteracija - 1] = 0;
          else
            Horizontali_Matrica[i][iteracija - 1] = -1;
    free(Laikinas_Masyvas);
  }

  for(j = 0; j < m; j++){
    Masyvo_nariu_kiekis = 0;
    Laikinas_Masyvas = NULL;
    for(i = 0; i < n; i++)
      if((Nepanaudotos_Atsargos[i] != 0) && (Nepatenkinti_Poreikiai[j] != 0)){
        yra = false;
        for(k = 0; k < Masyvo_nariu_kiekis; k++)
          if(Laikinas_Masyvas[k] == Kainu_Matrica[i][j])
            yra = true;
        if(!yra){
          Masyvo_nariu_kiekis++;
          Laikinas_Masyvas = (int *)realloc(Laikinas_Masyvas, sizeof(int) * Masyvo_nariu_kiekis);
          Laikinas_Masyvas[Masyvo_nariu_kiekis - 1] = Kainu_Matrica[i][j];
```

```
            }
        }
    i = 0;
    while(i < Masyvo_nariu_kiekis - 1)
        if(Laikinas_Masyvas[i] <= Laikinas_Masyvas[i + 1])
            i++;
        else{
            temp = Laikinas_Masyvas[i];
            Laikinas_Masyvas[i] = Laikinas_Masyvas[i + 1];
            Laikinas_Masyvas[i + 1] = temp;
            if(i > 0)
                i--;
        }
    Vertikali_Matrica[j] = (int *)realloc(Vertikali_Matrica[j], sizeof(int) * iteracija);
    if((Horizontali_Matrica_Pozymis == false) && (indeksas == j) && (Nepatenkinti_Poreikiai[j] == 0)){
        Vertikali_Matrica[j][iteracija - 1] = -1;
    }
    else
        if(Vertikali_Matrica[j][iteracija - 2] == -1)
            Vertikali_Matrica[j][iteracija - 1] = -1;
        else
            if(Masyvo_nariu_kiekis > 1)
                Vertikali_Matrica[j][iteracija - 1] = Laikinas_Masyvas[1] - Laikinas_Masyvas[0];
            else
                if(Masyvo_nariu_kiekis == 1)
                    Vertikali_Matrica[j][iteracija - 1] = 0;
                else
                    Vertikali_Matrica[j][iteracija - 1] = -1;
    free(Laikinas_Masyvas);
  }
};
//---------------------------------------------------------------------------
bool TLangas::Surasti_Maksimuma(bool *Horizontalus_Masyvas, int *indeksas){
    int temp_indeksas_1;
    int temp_indeksas_2;
    bool Algoritmo_Stabdymas;

    Pasirinkti_Horizontali_Matrica = (int *)realloc(Pasirinkti_Horizontali_Matrica, sizeof(int) * iteracija);
    Pasirinkti_Vertikali_Matrica = (int *)realloc(Pasirinkti_Vertikali_Matrica, sizeof(int) * iteracija);

    temp_indeksas_1 = 0;
    for(int j = 1; j < m; j++)
        if(Vertikali_Matrica[temp_indeksas_1][iteracija - 1] < Vertikali_Matrica[j][iteracija - 1])
            temp_indeksas_1 = j;

    temp_indeksas_2 = 0;
    for(int i = 1; i < n; i++)
        if(Horizontali_Matrica[temp_indeksas_2][iteracija - 1] < Horizontali_Matrica[i][iteracija - 1])
            temp_indeksas_2 = i;

    if((Vertikali_Matrica[temp_indeksas_1][iteracija - 1] < 0) && (Horizontali_Matrica[temp_indeksas_2][iteracija - 1] < 0))
        Algoritmo_Stabdymas = true;
    else{
        Algoritmo_Stabdymas = false;
        if(Vertikali_Matrica[temp_indeksas_1][iteracija - 1] > Horizontali_Matrica[temp_indeksas_2][iteracija - 1]){
            *Horizontalus_Masyvas = false;
            *indeksas = temp_indeksas_1;
            Pasirinkti_Horizontali_Matrica[iteracija - 1] = -1;
            Pasirinkti_Vertikali_Matrica[iteracija - 1] = temp_indeksas_1;
        }
        else{
            *Horizontalus_Masyvas = true;
            *indeksas = temp_indeksas_2;
            Pasirinkti_Horizontali_Matrica[iteracija - 1] = temp_indeksas_2;
            Pasirinkti_Vertikali_Matrica[iteracija - 1] = -1;
        }
    }
    return Algoritmo_Stabdymas;
```

```
};
//---------------------------------------------------------------------------
AnsiString TLangas::Sveikas_Skaicius(int Sk, int Plotis){
    AnsiString S = IntToStr(Sk);
    while(S.Length() < Plotis)
        S += " ";
    return S;
};
//---------------------------------------------------------------------------
int TLangas::Suma(){
    int S = 0;

    for(int i = 0; i < n; i++)
        for(int j = 0; j < m; j++)
            S += Kainu_Matrica[i][j] * Srautai[i][j];
    return S;
};
//---------------------------------------------------------------------------
void TLangas::Atspaudinti(){
    AnsiString Eil, S;
    bool temp;

    Tekstas->Lines->Add("     " + IntToStr(iteracija) + " iteracija:");

    Eil = "+--------------+";
    for(int j = 0; j < m - 1; j++)
        Eil += "------------";
    Tekstas->Lines->Add(Eil + "-----------+-------------+------------------+");

    Eil = "| Sandeliavimo | Vartojimo punktai     ";

    for(int j = 0; j < m - 2; j++)
        Eil += "            ";
    Tekstas->Lines->Add(Eil + "|            | Dar nepanaudotos |         Skaièiavimø ciklai");

    Eil = "|              ";
    for(int j = 0; j < m; j++)
        Eil += "+-----------";
    Eil += "+  Atsargos   |                  ";
    for(int j = 0; j < iteracija; j++)
        Eil += "+-----------";

    Tekstas->Lines->Add(Eil + "+");

    Eil = "|    punktai   ";
    for(int j = 0; j < m; j++)
        Eil += "|      W" + Sveikas_Skaicius(j + 1, 4);
    Eil += "|            |     atsargos     ";
    for(int j = 0; j < iteracija; j++)
        Eil += "|    " + Sveikas_Skaicius(j + 1, 4) + "    ";
    Tekstas->Lines->Add(Eil + "|");

    Eil = "+--------------";
    for(int j = 0; j < m; j++)
        Eil += "+-----------";
    Eil += "+-------------+------------------";
    for(int j = 0; j < iteracija; j++)
        Eil += "+-----------";

    Tekstas->Lines->Add(Eil + "+");

    for(int i = 0; i < n; i++){
        Eil = "|              ";
        for(int j = 0; j < m; j++){
            Eil += "|     | " + Sveikas_Skaicius(Kainu_Matrica[i][j], 4);
            temp = false;
        }
        Eil += "|            |                  ";
```

```
    for(int j = 0; j < iteracija; j++)
      Eil += "|          ";
    Tekstas->Lines->Add(Eil + "|");

    Eil = "|      F" + Sveikas_Skaicius(i + 1, 4) + "    ";
    for(int j = 0; j < m; j++)
      Eil += "+     +-----";
    Eil += "+   " + Sveikas_Skaicius(Pasiula[i], 10) + "|        " + Sveikas_Skaicius(Nepanaudotos_Atsargos[i], 10);
    for(int j = 0; j < iteracija; j++)
      if(Pasirinkti_Horizontali_Matrica[j] == i)
        if(Horizontali_Matrica[i][j] >= 0)
          Eil += "| * " + Sveikas_Skaicius(Horizontali_Matrica[i][j], 4) + " *  ";
        else
          Eil += "|    N      ";
      else
        if(Horizontali_Matrica[i][j] >= 0)
          Eil += "|   " + Sveikas_Skaicius(Horizontali_Matrica[i][j], 4) + "    ";
        else
          Eil += "|    N      ";
    Tekstas->Lines->Add(Eil + "|");
    Eil = "|              ";
    for(int j = 0; j < m; j++)
      if(Srautai[i][j])
        Eil += "| " + Sveikas_Skaicius(Srautai[i][j], 10);
      else
        Eil += "|    ---    ";
    Eil += "|            |                 ";
    for(int j = 0; j < iteracija; j++)
      Eil += "|          ";
    Tekstas->Lines->Add(Eil + "|");
    Eil = "+--------------";
    for(int j = 0; j < m; j++)
      Eil += "+-----------";
    Eil += "+-------------+------------------";
    for(int j = 0; j < iteracija; j++)
      Eil += "+-----------";
    Tekstas->Lines->Add(Eil + "+");
  };

  Eil = "|  Poreikiai   ";
  for(int j = 0; j < m; j++)
    Eil += "|   " + Sveikas_Skaicius(Paklausa[j], 8);
  Tekstas->Lines->Add(Eil + "|");

  Eil = "+--------------";
  for(int j = 0; j < m; j++)
    Eil += "+-----------";
  Tekstas->Lines->Add(Eil + "|   Suma = " + IntToStr(Suma()));

  Eil = "| Nepatenkinti ";
  for(int j = 0; j < m; j++)
    Eil += "|   " + Sveikas_Skaicius(Nepatenkinti_Poreikiai[j], 8);
  Tekstas->Lines->Add(Eil + "|");

  Eil = "+--------------";
  for(int j = 0; j < m; j++)
    Eil += "+-----------";
  Tekstas->Lines->Add(Eil + "|");

  Eil = "| Skaiciavimo  ";
  for(int j = 0; j < m; j++)
    Eil += "|          ";
  Tekstas->Lines->Add(Eil + "|");

  Eil = "|    ciklai    ";
  for(int j = 0; j < m; j++)
    Eil += "|          ";
  Tekstas->Lines->Add(Eil + "|");
```

```
    Eil = "+--------------";
    for(int j = 0; j < m; j++)
        Eil += "+-----------";
    Tekstas->Lines->Add(Eil + "|");

    for(int i = 0; i < iteracija; i++){
        Eil = "|      " + Sveikas_Skaicius(i + 1, 8);
        for(int j = 0; j < m; j++)
            if(Pasirinkti_Vertikali_Matrica[i] == j)
                if(Vertikali_Matrica[j][i] >= 0)
                    Eil += "| * " + Sveikas_Skaicius(Vertikali_Matrica[j][i], 4) + " *  ";
                else
                    Eil += "|    N      ";
            else
                if(Vertikali_Matrica[j][i] >= 0)
                    Eil += "|   " + Sveikas_Skaicius(Vertikali_Matrica[j][i], 4) + "    ";
                else
                    Eil += "|    N      ";
        Tekstas->Lines->Add(Eil + "|");

        Eil = "+--------------";
        for(int j = 0; j < m; j++)
            Eil += "+-----------";
        Tekstas->Lines->Add(Eil + "|");
    }
    Tekstas->Lines->Add("");
    Tekstas->Lines->Add("");
};
//---------------------------------------------------------------------------
void TLangas::Skaiciavimas(){
    bool Horizontaliai;
    int Indeksas;
    int Temp_skaicius;
    int* Temp;
    int Kiekis;
    int i;
    int j;

    Atspaudinti();
    Tekstas->Lines->SaveToFile("Rezultatai.txt");
    Pradines_Matricos();
    while(!Surasti_Maksimuma(&Horizontaliai, &Indeksas)){
        if(Horizontaliai){
            Kiekis = 0;
            Temp = NULL;
            for(j = 0; j < m; j++)
                if((Nepanaudotos_Atsargos[Indeksas] != 0) && (Nepatenkinti_Poreikiai[j] != 0)){
                    Kiekis++;
                    Temp = (int *)realloc(Temp, sizeof(int) * Kiekis);
                    Temp[Kiekis - 1] = j;
                }
            j = 0;
            while(j < Kiekis - 1)
                if(Kainu_Matrica[Indeksas][Temp[j]] <= Kainu_Matrica[Indeksas][Temp[j + 1]])
                    j++;
                else{
                    Temp_skaicius = Temp[j];
                    Temp[j] = Temp[j + 1];
                    Temp[j + 1] = Temp_skaicius;
                    if(j > 0)
                        j--;
                }
            if(Nepanaudotos_Atsargos[Indeksas] > Nepatenkinti_Poreikiai[Temp[0]]){
                Srautai[Indeksas][Temp[0]] = Nepatenkinti_Poreikiai[Temp[0]];
                Nepanaudotos_Atsargos[Indeksas] -= Nepatenkinti_Poreikiai[Temp[0]];
                Nepatenkinti_Poreikiai[Temp[0]] = 0;
            }
```

```
            else{
                Srautai[Indeksas][Temp[0]] = Nepanaudotos_Atsargos[Indeksas];
                Nepatenkinti_Poreikiai[Temp[0]] -= Nepanaudotos_Atsargos[Indeksas];
                Nepanaudotos_Atsargos[Indeksas] = 0;
            }
            //Atspaudinti();
            //Tekstas->Lines->SaveToFile("Rezultatai.txt");
            free(Temp);
        }
        else{
            Kiekis = 0;
            Temp = NULL;
            for(j = 0; j < n; j++)
                if((Nepanaudotos_Atsargos[j] != 0) && (Nepatenkinti_Poreikiai[Indeksas] != 0)){
                    Kiekis++;
                    Temp = (int *)realloc(Temp, sizeof(int) * Kiekis);
                    Temp[Kiekis - 1] = j;
                }
            j = 0;
            while(j < Kiekis - 1)
                if(Kainu_Matrica[Temp[j]][Indeksas] <= Kainu_Matrica[Temp[j + 1]][Indeksas])
                    j++;
                else{
                    Temp_skaicius = Temp[j];
                    Temp[j] = Temp[j + 1];
                    Temp[j + 1] = Temp_skaicius;
                    if(j > 0)
                        j--;
                }
            if(Nepanaudotos_Atsargos[Temp[0]] > Nepatenkinti_Poreikiai[Indeksas]){
                Srautai[Temp[0]][Indeksas] = Nepatenkinti_Poreikiai[Indeksas];
                Nepanaudotos_Atsargos[Temp[0]] -= Nepatenkinti_Poreikiai[Indeksas];
                Nepatenkinti_Poreikiai[Indeksas] = 0;
            }
            else{
                Srautai[Temp[0]][Indeksas] = Nepanaudotos_Atsargos[Temp[0]];
                Nepatenkinti_Poreikiai[Indeksas] -= Nepanaudotos_Atsargos[Temp[0]];
                Nepanaudotos_Atsargos[Temp[0]] = 0;
            }
            //Atspaudinti();
            //Tekstas->Lines->SaveToFile("Rezultatai.txt");
            free(Temp);
        }
        Matricu_Perskaiciavimas(Horizontaliai, Indeksas);
        Tekstas->Lines->Add(IntToStr(iteracija) + " iteracija.");
    }
    Atspaudinti();
    Tekstas->Lines->SaveToFile("Rezultatai.txt");
    free(Pasiula);
    free(Paklausa);
    for(int i = 0; i < n; i++)
        free(Kainu_Matrica[i]);
    free(Kainu_Matrica);
    for(int i = 0; i < n; i++)
        free(Srautai[i]);
    free(Srautai);
    free(Nepanaudotos_Atsargos);
    free(Nepatenkinti_Poreikiai);
};
//---------------------------------------------------------------------------
void __fastcall TLangas::TekstasClick(TObject *Sender){
    if((n == 0) || (m == 0))
        if(Atidaryti_Faila->Execute()){
            Duomenu_Nuskaitymas();
            Skaiciavimas();
        };
};
```