KAUNO TECHNOLOGIJOS UNIVERSITETAS
MATEMATIKOS IR GAMTOS MOKSLŲ FALULTETAS

Edvinas Duoba

GAMYBOS PROCESŲ OPTIMIZAVIMO UŽDAVINIŲ IR ALGORITMŲ ANALIZĖ

Baigiamasis magistro projektas

Vadovas
doc. dr. Narimantas Listopadskis

KAUNAS, 2017

KAUNO TECHNOLOGIJOS UNIVERSITETAS
MATEMATIKOS IR GAMTOS MOKSLŲ FALULTETAS

GAMYBOS PROCESŲ OPTIMIZAVIMO UŽDAVINIŲ IR ALGORITMŲ ANALIZĖ

Baigiamasis magistro projektas
Taikomoji matematika (621G10003)

Vadovas
doc. dr. Narimantas Listopadskis

Recenzentas
doc. dr. Stasė Petraitienė

Projektą atliko
Edvinas Duoba

KAUNAS, 2017

KAUNO TECHNOLOGIJOS UNIVERSITETAS
MATEMATIKOS IR GAMTOS MOKSLŲ FALULTETAS

Edvinas Duoba
Taikomoji matematika (621G10003)

Gamybos procesų optimizavimo uždavinių ir algoritmų analizė

AKADEMINIO SĄŽININGUMO DEKLARACIJA

2017m. birželio mėn. __d.
Kaunas

Patvirtinu, kad mano, Edvino Duobos, baigiamasis darbas tema „Gamybos procesų optimizavimo uždavinių ir algoritmų analizė“ yra parašytas visiškai savarankiškai ir visi pateikti duomenys ar tyrimų rezultatai yra teisingi ir gauti sąžiningai. Šiame darbe nei viena darbo dalis nėra plagijuota nuo jokių spausdintinių ar internetinių šaltinių, visos kitų šaltinių tiesioginės ir netiesioginės citatos nurodytos literatūros nuorodose. Įstatymu nenumatytų piniginių sumų už šį darbą niekam nesu mokėjęs.

Aš suprantu, kad išaiškėjus nesąžiningumo faktui, man bus taikomos nuobaudos, remiantis Kauno technologijos universitete galiojančia tvarka

_____ _____ _____
(studento vardas ir pavardė, įrašyti ranka) (parašas)

Duoba, E. „Gamybos procesų optimizavimo uždavinių ir algoritmų analizė“. Taikomosios matematikos Magistro baigiamsis darbas. Vadovas doc. N. Listopadskis; Kauno Technologijos Universitetas, matematikos ir gamtos mokslų fakultetas.

Mokslo sritis ir kryptis: Fiziniai mokslai, Matematika (01 P)

Reikšmingi žodžiai: Kombinatorinis optimizavimas, tvarkaraščių sudarymas, genetinis algoritmas.

Kaunas, 2017 134 psl.

SANTRAUKA

Darbe bus trumpai apžvelgiami bendrieji optimizavimo uždaviniai, klasės, į kurias yra skirstomi šie uždaviniai. Taip pat bus apžvelgiami algoritmai, naudojami šiems uždaviniams spręsti. Plačiau bus aptariami tvarkaraščių sudarymo uždaviniai, šių uždavinių klasifikacijos ir naudojami algortimai, bei pagrindiniai sprendžiami tvarkaraščių sudarymo uždaviniai.

Antroje darbo dalyje bus detaliai nagrinėjamas Genetinis algoritmas, jo idėja, schema. Bus nagrinėjamos programoje realizuotos genetinio algoritmo modifikacijos – kryžminimas ir mutacijos.

Trečioje dalyje bus pristatoma programa, jos veikimas, pateikiama valdymo instrukcija. Toliau bus pateikiami ir aptariami rezultatai, daromos išvados.

Duoba, E. „Optimization of production processes problems and algorithms analysis“. Master‘s final project in applied mathematics/ supervisor assoc. dr N.Listopadskis; Kaunas University of Technology, Faculty of mathematics and natural sciences.

Research area and field: Natural sciences, Mathematics (01 P)

Key words: Combinatorial optimization, scheduling, genetic algorithm.

Kaunas, 2017 134psl

SUMMARY

The work will be carried out in generic optimization review – what algorithms are used to solve thsese challenges, in what classes these challenges ar devided. More will be dealt with scheduling tasks and their classification, decision alorithms.

In the second part is presented genetic algorithm - the idea, scheme. Crossing and mutation features in created program will be discussed in detail.

Program management instruction, results, analysis and conclusions will be presented in third part.

TURINYS

Santrauka/Summary 4
Įžanga 6
1 Gamybos tvarkaraščių sudarymo uždavinių ir jų sprendimo algoritmų literatūros apžvalga 7
1.1 Optimizavimas 7
1.2 Kobinatorinio optimizavimo uždavinių klasifikacija 9
1.3 Tvarkaraščių sudarymo uždavinių klasifikacija 9
1.4 Tipiniai tvarkaraščių uždaviniai 10
1.4.1 Statiniai ir dinaminiai tvarkaraščiai 11
1.4.2 Tvarkaraščiai ir išteklių tipai 11
1.4.3 Tvarkaraščių ribojimai 11
1.4.4 Kaip apibrėžiami tvarkaraščių duomenys 12
1.5 Tvarkaraščių uždaviniams spręsti naudojami algoritmai 13
1.6 Pagrindiniai gamybos tvarkaraščių sudarymo uždaviniai 14
2. Genetinio algoritmo panaudojimas gamybinių tvarkaraščių sudarymui 16
2.1 Genetinis algoritmas 16
2.1.1 Algoritmo schema 16
2.1.2 Chromosomos kodavimas 17
2.1.3 Kryžminimas ir mutacija 17
2.4.1 Selekcija 19
3. Genetinių algoritmų parametrų įtakos tyrimas 20
3.1 Sukurta programa 20
3.1.1 Programos, tyrimo tikslai 20
3.1.2 Duomenų failo paruošimas 20
3.1.3 Vartotojo sąsaja, programos paaiškinimai 24
3.2 Sukuriamas rezultatų .txt failas 31
3.3 Tyrimo rezultatai 32
3.3.1 Parinkti skaičiavimų duomenys 32
3.3.2 Kryžminimo ir mutacijų grafikai 32
3.3.3 Skaičiavimų rezultatai atlikus 200 iteracijų, tikslo funkcija – prastovų minimizavimas 34
3.3.4 Skaičiavimų rezultatai atlikus 500 iteracijų, tikslo funkcija – prastovų minimizavimas 35
3.3.5 Skaičiavimų rezultatai atlikus 200 iteracijų, tikslo funkcija – gamybos trukmės minimizavimas 36
3.3.6 Skaičiavimų rezultatai atlikus 500 iteracijų, tikslo funkcija – gamybos trukmės minimizavimas 37
Išvados 39
Literatūra 40
1. priedas Programos kodas 41

ĮŽANGA

Vis spartėjant gyvenimo tėkmei, siekiama kiek įmanoma efektyviau panaudoti laiką, fizinius išteklius. Norima išspręsti iškylančius uždavinius: pasiekti tikslą, per mažiausią galimą laiką, aplankyti kelias skirtingas vietoves taip, kad kuro sąnaudos būtų minimalios, gaminti gaminius darbą paskirstant taip, kad visos staklės dirbtų maksimaliu pajėgumu ir prastovos būtų minimalios, kaip paskirstyti darbą, kad numatyti produkcijos kiekiai būtų pagaminti per nustatytą laiko tarpą ir kiti uždaviniai. Visų šių, kiekvieną dieną iškylančių, rūpesčių geriausio sprendimo ieškojimas vadinamas - optimizavimu.

Optimizavimas, tai yra procesas, kuriuo siekiama rasti geriausią sprendinį arba šiek tiek patobulinti jau turimą. Sprendžiant uždavinius, tokius kaip tiesinė lygtis, galime rasti tikslų sprendinį ar sprendinių aibę, tačiau sprendžiant net ir paprasčiausią tvarkaraščių sudarymo uždavinį, optimalų sprendinį galima surasti tikrai ne visada. Tokiu atveju dėmesys yra kreipiamas į tikslo funkciją ar funkcijas, stengiamasi atitikti jų sąlygas.

Yra begalo daug įvairių optimizavimo uždavinių. Jeigu kriterijus tik vienas, optimizavimas vadinamas vienakriteriniu, jeigu uždaviniuose yra daugiau nei vienas kriterijus, tokie uždaviniai yra vadinami daugiakriteriniais optimizavimo uždaviniais. Didėjant kriterijų skaičiui, optimizavimas tampa vis sudetingesnis.

Šiame darbe aptarsime optimizavimo uždavinius, jiems spręsti naudojamus algoritmus, (tikslieji, euristiniai ir metaeuristiniai) bei tvarkaraščių sudarymo uždavinių klasifikacijas. Daugiau dėmesio skirsime aptarti gamybos tvarkaraščių sudarymo uždavinius.

Antroje dalyje yra pristatoma sukurta programa, pateikiami duomenų failo fragmentai ir sąlygos, keliamos duomenų failui, kad veiktų programa, aprašoma vartotojo sąsaja. Taip pat bus nagrinėjami programos pagalba gauti genetinio algoritmo modifikacijų rezultatai:

- Keičiama mutacijos rūšis. Pasirinktinai naudoti vieną iš penkių skirtingų pateiktų mutacijų,
- Keičiama tikslo funkcija.

Kryžminimų ir mutacijų specifika yra pateikta darbo eigoje.

1. Gamybos tvarkaraščių sudarymo uždavinių ir jų sprendimo algoritmų literatūros apžvalga

1.1 Optimizavimas

Optimizavimas – elemento paieška iš apibrėžtos aibės, kurio reikšmė būtų minimali arba maksimali, atsižvelgiant į tikslo funkcijas. Matematikos šaka, kuri nagrinėja tokių uždavinių sprendimą,vadinama matematiniu programavimu.

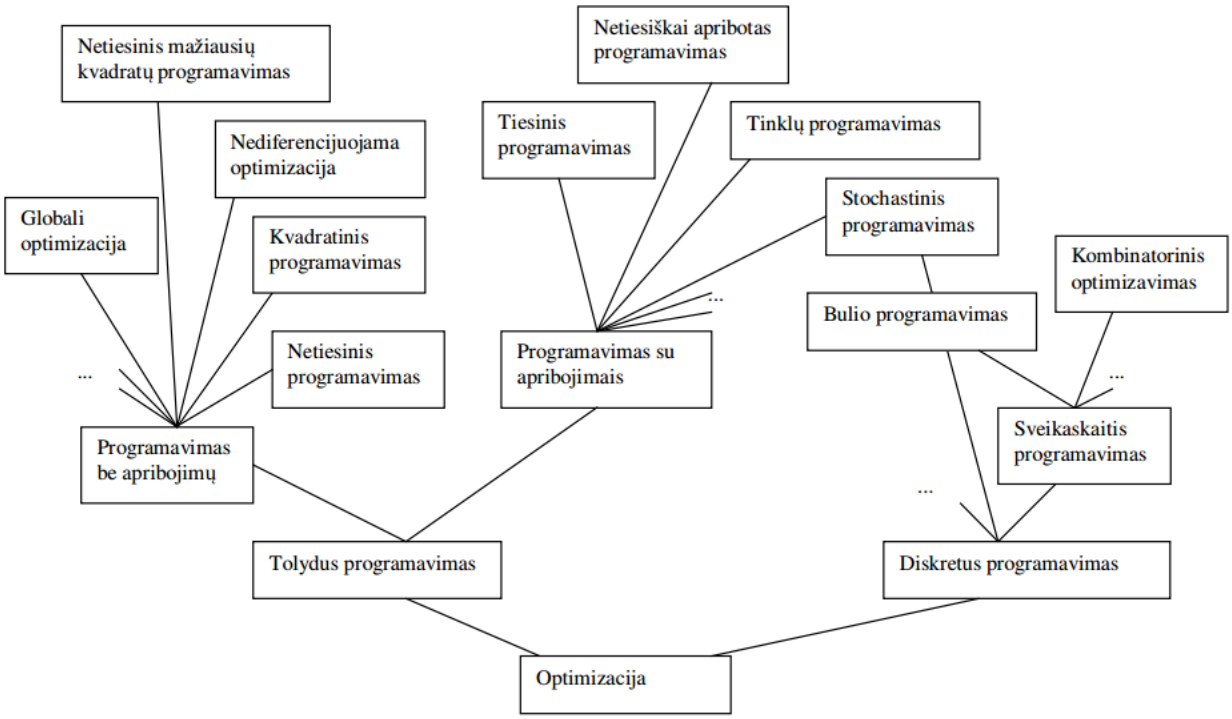
Optimizavimo uždaviniai, tai labai plačiai paplitęs tyrimo objektas. Keletas tai nulėmusių priežasčių – labai suintensyvejęs efektyvių būdų ieškojimas, siekiant spręsti optimizavimo uždavinius, itin spartus kompiuterijos vystymąsis leidžia atlikti milžiniškus skaičiavimų kiekius per priimtiną laiko tarpą. To pasekoje pastoviai atsiranda vis naujų skaičiavimų algoritmų.

Optimizavimo uždavinys gali būti užrašomas [12]:

Duota: f - tikslo funkcija f:A→R

Rasti: elementą x_0 aibėje A, tokį kad $f(x_0) \leq f(x)\ \forall x \in A$ - minimumas, arba tokį, kad $f(x_0) \geq f(x)\ \forall x \in A$ - maksimumas.

Aibė A yra Euklido erdvės R^n poaibis apibrėžiamas lygybiniais arba nelygybiniais apribojimais. Aibės A elementai yra vadinami sprendiniais. Optimalūs sprendiniais, tai tie aibės A elementai, kurie minimizuoja arba maksimizuoja tikslo funkciją ar funkcijas. Pagal tai, kaip apibrėžiamos tikslo funkcijos, optimizacija yra skirstoma įvairiais aspektais. Optimizacijos klasifikacijos fragmentas (žr. 1.1.1 pav.).



1.1.1 pav. Optimizacijos klasifikacijos fragmentas

Toliau bus apžvelgiamos tam tikroms klasėms būdingos savybės, taikymo sritys ir sprendimo metodai [8].

Tiesinis programavimas (TP) – arba tiesinis optimizavimas – tai yra tiesinė tikslo funkcija. Jos leistinoji sritis yra apibrėžta tiesinėmis bei netiesinėmis lygybėmis [12][1].

Matematiškai užrašomas [12]:

$\min\{c^T x: Ax \leq b, x \geq 0\}$, kur x yra kintamųjų vektorius, c ir b yra koeficientų vektoriai, A yra koeficientų matrica, c^T - transponuota c matrica.

Tiesinis programavimas gali būti taikomas įvairiose studijų srityse. Jis naudojamas verslo ir ekonomikos srityse, tinkamas panaudoti sprendžiant kai kurias inžinerines problemas, sėkmingai naudojamas pramonės srityse, sprendžiant transportavimo, energetikos, telekomunikacijų ir gamybos uždavinius.

Kvadratinis programavimas (QP angl.) – tai kvadratinė tikslo funkcija, kuri apribota tiesinėmis lygtymis ir nelygybėmis. Taikomas sprendžiant portfelio optimizavimo, elektros energijos gamybos, bei inžinerinio projektavimo optimizavimo uždavinius [12].

Matematinis užrašymas:

$\min\{\frac{1}{2}x^T Qx + c^T: Ax \leq b\}$, c yra n-matis vektorius, Q yra $n \times n$ simetrinė matrica, A yra $m \times n$ matrica, b yra m-matis vektorius.

Sveikaskaitinis programavimas - programavimo būdas, kai visi elementai privalo būti sveiki skaičiai. Daugeliu atveju sveikaskaitinis programavimas yra siejamas su sveikų skaičių tiesiniu programavimu (integer linear programming ILP angl), kurio apribojimai yra tiesinės lygtys ir nelygybės [12].

Matematinė sveikaskaitinio programavimo išraiška:

$\max\{c^T x: Ax \leq b, x \geq 0, x \in Z^n\}$.

Matematinė sveikų skaičių tiesinio programavimo išraiška:

$\max\{c^T x: Ax + s = b, s \geq 0, x \in Z^n\}$.

Čia c ir b yra sveikų skaičių vektoriai, o A - sveikų skaičių matrica.

Netiesinis programavimas – tai toks programavimas, kai tikslo funkcijas ir leistinąją sritį apibrėžiančios lygybės ir nelygybės gali būti netiesinės [12].

Matematiškai užrašomas:

$\min_{x \in X} f(x)$ arba $\max_{x \in X} f(x)$, kur $f: R^n \to R, x \in R^n$

Stochastinis programavimas – programavimo būdas, kai nagrinėjamuose programavimo uždaviniuose yra neapibrėžtumas. Kadangi deterministinės problemos formuluojamos su žinomais parametrais, sprendžiant realias problemos, beveik visada būna parametrų, kurių nežinome. Taigi, šio programavimo tikslas rasti sprendimą, kuris būtų tinkamas visiems duomenims ir būtų optimalus tam tikra prasme. Plačiai taikomi dviejų lygių tiesinio programavimo stochastiniai uždaviniai.

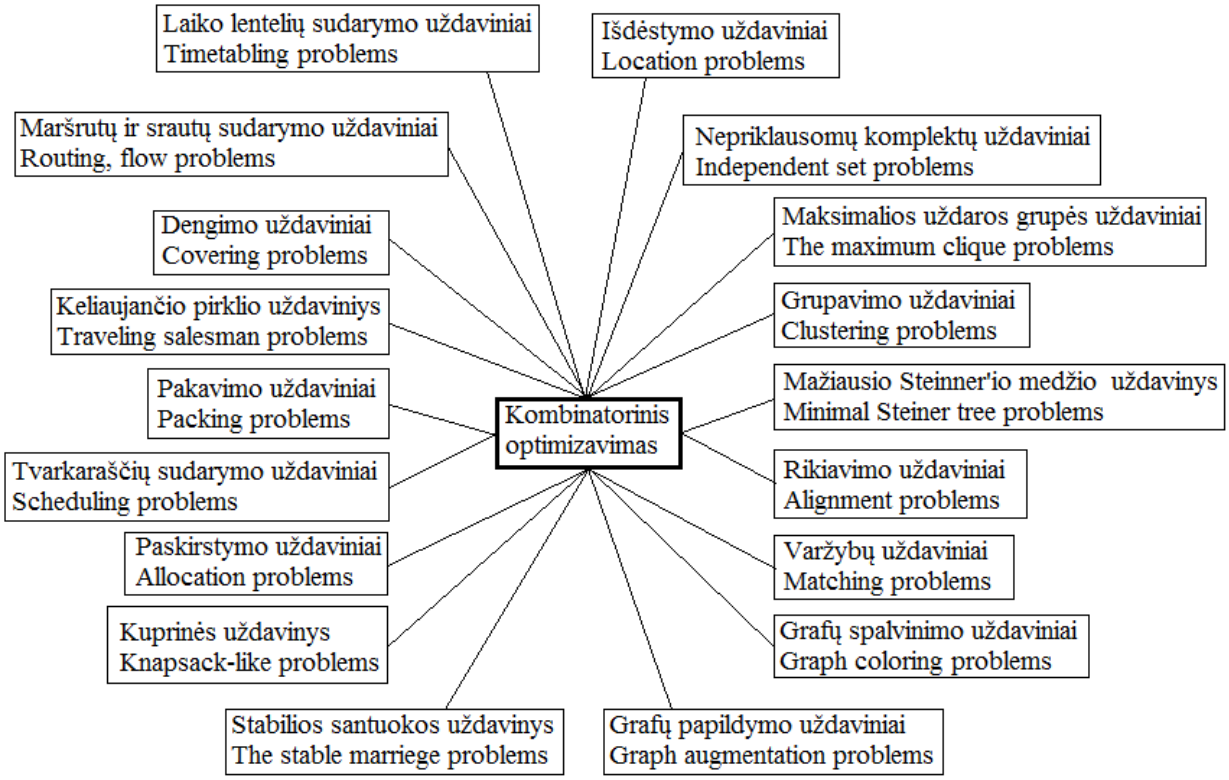
Sakykime, kad pirmame lygmenyje yra priimamas sprendimas, po kurio įvyksta įvykis, darantis įtaką šiam sprendimui. Tuomet antro lygio sprendimas yra priimamas toks, kad būtų pataisytas pirmasis sprendimas, jeigu buvo atlikti pokyčiai yra neigiami.

Stochastinis programavimas yra labai plačiai taikomas, pradedant finansais, transportavimu ir baigiant energijos paskirstymo optimizavimu [12][8].

Kombinatorinis optimizavimas – nagrinėja uždavinius su diskretinio tipo kintamaisiais (sveikaisiais skaičiais, sveikų skaičių rinkiniais, poaibiais, grafais, perstatymais ir pan). Kintamų reikšmių aibė yra baigtinė. Formaliai kombinatorinio optimizavimo uždavinys gali būti aprašomas pora (S,f). Pirmasis narys (S,f) yra sprendinių aibė, antrasis – tikslo funkcija. Jos apibrėžimo sritis aibė S, o reikmšių aibė yra realieji skaičiai [2].

1.2 Kobinatorinio optimizavimo uždavinių klasifikacija

Pramonės, komercinės veiklos ir įvairiose kitose sferose yra sprendžiami veiklos efektyvumo uždaviniai: sandėlio patalpos projektuojamos taip, kad turima erdvė būtų maksimaliai panaudota sandėliavimui. Gamybos patalpos projektuojamos taip, kad visi turimi įrenginiai būtų kuo mažiau atitolę vienas nuo kito, eismo srautų paskirstymo, transportavimo, gamybos darbų planavimas, resursų panaudojimas ir kiti. Jeigu uždavinio apimtis ar struktūra yra labai didelė, tokio tipo uždavinius spręsti yra itin sudėtinga. Šių uždavinių optimalų rezultatą retai kada galime surasti naudojant įprastus metodus [10]. Šioje srityje labai pagelbsti kombinatorinis optimizavimas. Pagal kombinatorinio optimizavimo uždavinių formuluotę ir ieškomą sprendinį, uždaviniai yra skirstomi į klases (žr. 1.2.1 pav.) [3].



1.2.1 pav. Kombinatorinio optimizavimo uždavinių klasės

1.3 Tvarkaraščių sudarymo uždavinių klasifikacija

Visais laikais, brangiausias žmonių turtas buvo, yra ir bus – laikas, tad efektyvus jo planavimas - tvarkaraščių sudarymo uždaviniai - yra aktualūs visose gyvenimo srityse.

Tvarkaraščių sudarymo uždaviniai dažniausiai yra sprendžiami tokiose srityse kaip: gamyba, ryšių ir kompiuterinių tinklų architektūros parinkimas bei informacijos srautų valdymas, logistika, vadyba, verslas, projektavimas, ekonomika ir kitos sritys [7].

Gamyboje sprendžiami tokie uždaviniai: gamybos plano optimizavimas – produkcijos gamyba, pelno maksimizavimas, efektyvus resursų panaudojimas, išmetamų žaliavų likučių minimizavimas.

Gamybos procesų optimizavimo uždaviniuose turima informacija, tai duomenys apie atliekamas operacijas, jų nuoseklumo sąryšius, trukmes, būtinus laukimo laikus, įrenginių ir kitus

ribojimus. Tokių uždavinių sprendimo tikslas – nustatyti tokią darbų atlikimo eigą, gaminių gamybos eiliškumą, kad visų darbų atlikimo laikas būtų minimalus, kiek įmanoma sumažinti būtino laukimo laiką, sąnaudas ir kaštus, atsižvelgiant į nustatytus apribojimus, tikslo funkcijas.

Įvairūs srautinių problemų uždaviniai (srautų apdorojimas) yra sprendžiami telekomunikacijų tinkluose. Šie uždaviniai labai svarbūs GSM tinkluose, ten, kur tenka apdoroti duomenų paketus, analizuoti duomenų pliūpsnių srautus, reguliuoti pranešimų srautus. Su panašaus tipo uždaviniais tenka susidurti ir kompiuteriniuose tinkluose, kur apdorojami siunčiamų duomenų srautai.

Transporto planavimo ir logistikos uždaviniai yra be galo svarbūs šiomis dienomis. Šiai uždavinių kategorijai priskiriami transporto pasirinkimo ir jo skirstymo uždaviniai. Iš didelės sprendinių aibės (transportavimo priemonių ir galimų maršrutų parinkimas) reikia parinkti tokius sprendinius, kad būtų optimizuojamos tikslo funkcijos – minimizuojamos vežimo išlaidos ir transportavimo laikas.

Sudėtingi optimizavimo uždaviniai sprendžiami ir viešojo transporto judėjimui sektoriuje. Reikia optimizuoti maršrutus, minimizuojant sąnaudas, maksimizuojant pasirinktos teritorijos padengiamumą, optimizuoti reikalingų transporto priemonių kiekį, patenkinti keleivių srautams, sudaryti tokias laiko lenteles, kad keleivių laukimo laikas būtų minimalus.

Su panašiais, tačiau sudėtingesniais uždaviniais tenka susidurti sprendžiant traukinių bei lėktuvų eismo valdymo uždaviniuose, kur yra be galo svarbus aukšto lygio tikslumas ir patikimumas.

Jūrų uostuose yra sprendžiami konteinerių krovimo, sandėliavimo, tvarkymo bei išvežimo uždaviniai. Labai aktualu tinkamai suorganizuoti visus pakrovimo darbus taip, kad kroviniai uostuose būtų kuo trumpiau. Tokie patys uždaviniai sprendžiami ir gamyboje, kai nėra sandėliavimo patalpų – kaip planuoti atliekamas gamybos operacijas, kad atlikus vieną operaciją, nereikėtų laukti, kol bus galima atlikti sekančią.

Tvarkaraščių sudarymas labai svarbus ir švietimo įstaigose ir tai yra itin sudėtinga, kalbant aukštųjų ir profilinių mokyklų užsiėmimus. Dažniausiai šiuose uždaviniuose planuojami ir optimizuojami savaitiniai užsiėmimų tvarkaraščiai studentų, bei moksleivių grupėms, priskiriant grupei ar pogrupiui dalyko mokytoją ar dėstytoją, patalpas. Labai dažnai susiduriama su įvairiais apribojimais (pvz. auditorijos ar klasės vietų skaičius ir grupės dydis ir kitais.) [7].

1.4 Tipiniai tvarkaraščių uždaviniai

Įvairiose srityse kylantys tvarkaraščių uždaviniai labai dažnai turi tas pačias ypatybes, todėl galima formuluoti apibendrintus tvarkaraščių uždavinius. Pagal uždavinių formuluotes, jie skirtomi į uždavinių grupes, kurioms jau yra preliminariai parinkti gerai veikiantys sprendimo metodai. Dažnai visiškai skirtingų sričių uždaviniai būna suvedami į vienodą uždavinių klasę. Yra be galo daug tvarkaraščių uždavinių, todėl jie klasifikuojami pagal tai, kokia informacija yra reikalinga, kad apibūdinti tvarkaraštį. Tvarkaraščiams sudaryti naudojama informacija, tai užduočių trukmės, jų nuoseklumo sąryšiai, naudojami ištekliai, jų atsargos ir specifiniai ribojimai, darbo centrų pajėgumai, tvarkaraščio realizavimo ypatybės ir panašiai.

Tvarkaraščių uždaviniams klasifikuoti dažniausiai išskiriami šie kriterijai [7]:

- Duomenų srautų pobūdžiai – statiniai ir dinaminiai srautai,
- Išteklių tipai,
- Ribojimų pobūdis,
- Duomenų apibrėžimas.

1.4.1 Statiniai ir dinaminiai tvarkaraščiai

Kaip pateikta antraštėje - tvarkaraščių planavimas yra statinis arba dinaminis. Kai planavimas yra statinis, naudojami duomenys apie darbų srautą yra iš anksto žinomi, fiksuoti. Tikslas - parinkti geriausią darbų eiliškumą, atsižvelgiant į visus turimus apribojimus (darbų nuoseklumas, žaliavų ir finansų ištekliai, skirtingų staklių įvairios savybės, darbuotojų darbo sugebėjimai atliekant darbus ir pan). Šių uždavinių sprendimui naudojami kalendorinio ir tinklinio planavimo metodai.

Dinaminiuose uždaviniuose atsižvelgiama darbo jėgos, techninių įrenginių, finansinius ir laiko apribojimus. Transporto ir krovimo srityse, gamybos planavime, telekomunikacijų ir ryšių tinklų valdymo srityse yra sprendžiami būtent šio tipo uždaviniai. Dinaminio tipo uždaviniuose pradinė informacija neturi būti žinoma, ji gali paaiškėti procesų eigoje [7].

1.4.2 Tvarkaraščiai ir išteklių tipai

Tvarkaraščių uždaviniai gali būti skirstomi pagal tai, kokie ištekliai yra vertinami uždavinyje. Išskiriamos šios grupės:

- Uždaviniai, turintys apribojimų vykdymui - darbų pertraukimais, darbo jėgos ir darbo centrų, įrengimų apribojimai. Šiuose uždaviniuose yra tiriamas darbo resursų paskirstymas statiniuose ir dinaminiuose uždaviniuose.
- Uždaviniai, turintys medžiagų apribojimų – išskiriami uždaviniai kai yra medžiagų perteklius arba stygius, atliekamas medžiagų kokybės tikrinimas. Šie uždaviniai sutinkami sprendžiant medžiagų atsargų paskirstymo uždavinius.
- Uždaviniai, kai nėra turima visa informacija – tokiuose uždaviniuose tenka priimti įvairius sprendimus, kai nėra turima reikiama informacija konkrečiai situacijai, arba informacija yra pasenusi ir netinkama. Žinių ištekliai, tai faktai, strategijos ir technologios. Pastovus žinių išteklių vystymas ir tobulinimas garantuoja, kad bus surandami tvarkaraščių sudarymo uždavinių sprendiniai [7].

1.4.3 Tvarkaraščių ribojimai

Tvarkaraščių uždaviniuose sutinkami įvairūs ribojimai. Atsitiktiniu laiko momentu gali pritrūkti išteklių, pažeisti nuoseklumo sąryšiai, reikalaujama paisyti nurodytų laiko terminų ir t.t.

Tvarkaraščių uždavinius galima suskirstyti į kelias grupes, pagal uždavinių apribojimų sritis [7]:

- Tvarkaraščiai, kurie turi laiko aprobbojimų – pateiktos darbų pradžių ar pabaigų taisyklės. Kiekvienas tvarkaraščio uždavinys turi induvidualius laiko apribojimus.
- Tvarkaraščiai, esant išteklių apribojimams – ilgalaikiai ar fiziniai medžiagų, darbo jėgos ir techninių įrengimų apribojimai..
- Tvarkaraščiai, kai būtina išlaikyti konkretų nuoseklumą – tai uždavinių grupė, kur yra būtina išlaikyti veiklos nuoseklumą. Atskiras atvejis, kai nagrinėjami tvarkaraščiai, kuriuose nėra svarbus eiliškumas, nėra eiliškumo apribojimų.

Lengva suprasti, kad kuo daygiau yra nuoseklumo apribojimų, tuo sunkiau sudaryti efektyvų tvarkaraštį.

Pagal tvarkaraščių apribojimus, visi tvarkaraščių uždaviniai skaidomi į šias grupes:

- Tvarkaraščiai, turintys laikinus aprbojimus – uždaviniuose ribojimai taikomi laiko intervalams, ribojama atliekamų darbų, operacijų pradžia, pabaiga, trukmė, taip pat gali būti ribojami ir resursų paskirstymai. Atliekant skaičiavimus, laikini ribojimai būna arba patenkinami ir naikinami, arba atnaujinami.
- Tvarkaraščiai, turintys nenumatytus ribojimus – tokiuose uždaviniuose ribojimai nusako netikėtus įvykius – ribojimai, kurie nebuvo iš anksto nenumatyti, tačiau jie keičia prioritetus ar išteklių paskirstymą (pvz. sugenda staklės ir pan.). Atsiradus naujiems apribojimams, atsižvelgiama į esamus ir atliekamas tvarkaraščio perdarymas.
- Tvarkaraščia, turintys pastovius ribojimus – dažną kartą tai būna fiziniai ribojimai, darantys įtaką tvarkaraščio sudarymo procesui, resursų paskirstymui. Įprastai kiti apribojimai nedaro įtakos šiems.

Tvarkaraščių teorijoje, uždaviniai su ribotais ištekliais, yra vieni iš svarbiausių, plačiai praktikoje pritaikomų uždavinių. Teorija teigia, kad kad net sudarius gerus tvarkaraščių uždavinių sprendimo algoritmus, yra vistiek paliekama pakankamai daug neatsakytų klausimų.

1.4.4 Kaip apibrėžiami tvarkaraščių duomenys

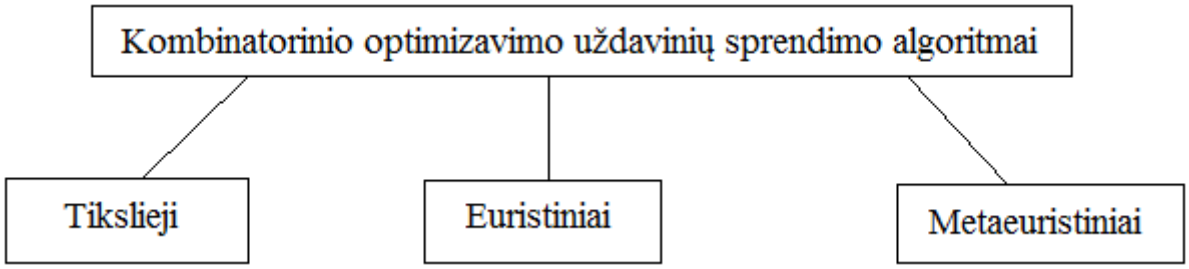
Pagal tai, kaip apibrėžiami tvarkaraščių uždavinių duomenys, uždaviniai klasifikuojami į:

- Deterministiniai uždaviniai – kai yra turima visa reikalinga informacija, kad sudaryti tvarkaraštį.
- Stochastiniai uždaviniai – kai duomenys pateikiami tikimybiniais modeliais.
- Nedeterministiniai, dar vadinami nestochastiniai uždaviniai – kai dalis duomenų uždaviniuose nėra apibrėžti, o ši neapibrėžtis apibūdinama netikimybiniais modeliais (intervalinė aritmetika, blausioji logika ir pan.)
- Uždaviniai, kurie nėra pilnai apibrėžti – dalis turimų duomenų gali būti klaidingi, praleisti arba nežinomi.

Labiausiai yra išnagrinėti deterministiniai tvarkaraščių sudarymo uždaviniai. Tiriant stochastinius ir nestochastinius, arba nedeterministinius tvarkaraščių sudarymo metodus, vis tiek būna neišspręstų uždavinių. Uždaviniai, kurie nėra pilnai apibrėžti, pirmiausia suvedami į deterministinius arba nedeterministinius uždavinius ir tik tada sprendžiami .

1.5 Tvarkaraščių uždaviniams spręsti naudojami algortimai

Visi kombinatorinio optimizavimo algoritmai, naudojami tvarkaraščiams sudaryti, yra skirstomi į tris rūšis - tikslieji, euristiniai ir metaeuristiniai (žr. 1.5.1 pav.).



1.5.1 pav. Kombinatorinio optimizavimo algoritmų rūšys

Tikslieji algortimai, tai tokie algoritmai, kurie užtikrina, kad bus surastas geriausias uždavinio sprendinys. Tačiau tiek bendrai, beveik visų kombinatorinio optimizavimo uždavinių sprendimo trukmė, tiek tvarkaraščių uždavinių sprendimo trukmė tiksliaisiais metodais ddidėja eksponentiškai, didėjant uždavinio duomenų apimčiai [10][6]. Todėl tikslieji metodai taikytini sprendžiant tik nedidelės apimties uždavinius.

Tiksliųjų algoritmų pvz.:

- kertančiųjų plokštumų,
- šakų ir ribų,
- šakų ir rėžių.

Euristinis algoritmas, tai skaičiavimų metodas, kai siekiama surasti sąlyginai gerą sprendinį per priimtiną laiko tarpą, tačiau nebūtinai optimalų. Tai yra esminis skirtumas tarp euristinių algoritmų ir tikslių metodų. Euristiniai algoritmai paremti pagal tam tikrų taisyklių (euristikų) pritaikymu sprendžiant konkrečius kombinatorinio optimizavimo uždavinius. Tokios taisyklės nurodo koks sprendimas turi būti priimtas įvairiose, kad potencialiai būtų gaunamas geresnis sprendinys. Euristiniai algoritmai taikomi tam tikrai uždavinių grupei, kai galioja tam tikros prielaidos ar apribojimai [10].

Euristinių metodų pvz.:

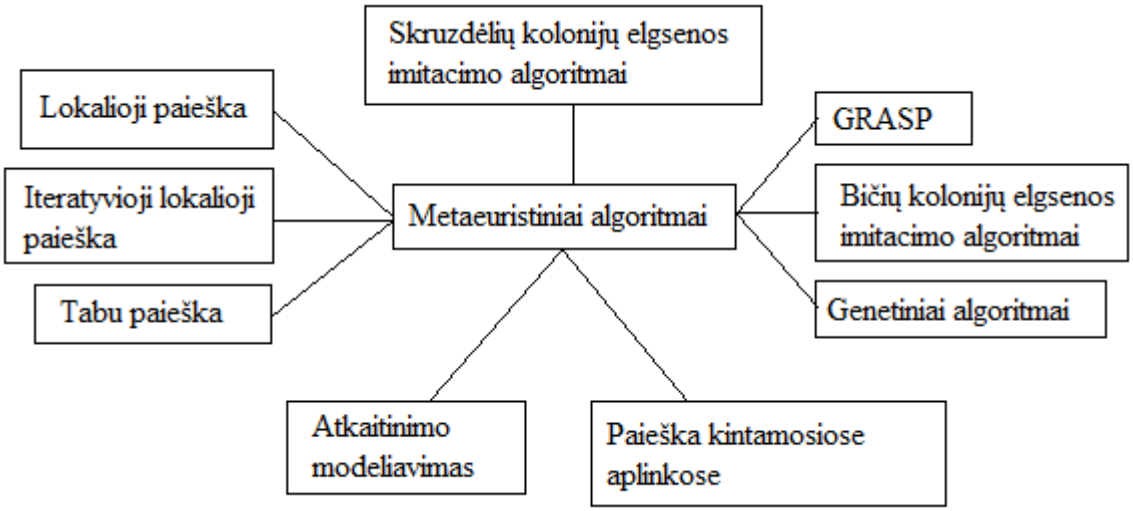
- prisitaikančios atminties,
- nuolatinio judėjimo.

Metaeuristiniai metodai, tai rinkinys abstrakčių nurodymų. Skirtingai nei euristiniai algoritmai, šie nurodymai skirti tam, kad formaliai aprašyti sprendimo metodiką kuriai nors uždavinių klasei. Sąvokoje „metaeuristinis algoritmas“ telpa daugiau, nei sąvokoje - „euristinis metodas“ [10].

Keletas metaeuristinių algoritmų.:

- tabu paieška,
- lokalioji paieška,
- genetiniai algoritmai,
- atkaitinimo modeliavimas,
- iteratyvioji lokalioji paieška,
- paieška kintamosiose aplinkose,
- bičių kolonijų elgsenos imitavimo algoritmai.
- skruzdėlių kolonijų elgsenos imitavimo algoritmai,

- godžiosios randomizuotos adaptyvios paieškos procedūros (GRASP).



1.5.2 pav. Kombinatorinio optimizavimo uždavinių sprendimo metaeuristiniai algoritmai

Įvairių optimizavimo uždavinių, taip pat ir uždaviniuose, kur sudarinėjami tvarkaraščiai, galima taikyti praktiškai visas pateiktas metaeuristinių metodų idėjas. Tai yra didžiausias metaeuristinių metodų privalumas, lyginant su euristiniais algoritmais, nukreiptais spręsti kokį nors konkretų uždavinį. Kai norima rasti aukštesnės kokybės sprendinius, galima sujungti kelias skirtingas metaeuristinių algoritmų idėjas, dėl jų universalumo. Todėl šiuo metu pagrindiniai tyrinėjimo objektai yra metaeuristiniai algoritmai arba jų idėjų kombinacijos, kaip lokalioji paieška, tabu paieška ir atkaitinimo modeliavimas ar genetiniai algoritmai ir kt. Taip norima sukurti algoritmus, kurie kuo efektyviau spręstų kombinatorinio optimizavimo uždavinius – būtų randamas optimalus ar jam artimas sprendinys per priimtiną laiko tarpą, kurį užtrunka skaičiavimai [10].

1.6 Pagrindiniai gamybos tvarkaraščių sudarymo uždaviniai

Klasikinis gamybos tvarkaraščių sudarymo uždavinio modelis susideda iš mašinų ir darbų. Kiekvienas darbas (produkto gamyba) susideda iš aibės operacijų. Kiekviena operacija atliekama ant tam tikros mašinos ir užtrunka tam tikrą laiko tarpą. Operacijos negali vykti kartu – nepasibaigus vienai operacijai, ta pačia mašina negalima pradėti dar vienos operacijos. Vienu metu visos mašinos gali vykdyti tik vieną operaciją, o operacijai prasidėjus, ji negali būti nutraukta [11].

Yra trys išnagrinėti modeliai. Pirmasis - gamyba be nuoseklumo sąryšių (angl. open shop), Antrasis - iš dalies nuosekli gamyba (angl. job shop), Trečiasis - nuosekli gamyba (angl. flow shop).

Uždavinyje, kur gamyba yra be nuoseklumo sąryšių (open shop), atliekamos operacijos apdorojamos atsitiktine tvarka. Sakykime, kad mašinų skaičius $m \in Z^+$, o darbų aibė J. Visi darbai $j \in J$ susideda iš m operacijų $o_{i,j}, 1 \leq i \leq m$ ($o_{i,j}$, o – operacija, i – massina su kuria apdorojama ši operacija), kiekvienos operacijos atlikimo trukmė $l_{i,j} \in N$. Darbų aibei J sudarytas tvarkaraštis, tai vienos mašinos tvarkaraščių rinkinys. Mašinai $i, 1 \leq i \leq m$, ieškoma ir randama atitiktis $f_i: J \to N$, čia $f_i(j) > f_i(j`)$ reiškia $f_i(j) \geq f_i(j`) + l_{i,j}$, o kiekvienam

atliekamam darbui $j \in J$ intervalai $[f_i(j), f_i(j) + l_{i,j})$ yra nesikertantys. Šio tvarkaraščio darbų pabaiga

$$\max_{1 \le i \le m, j \in J} f_i(j) + l_{i,j} .$$

Uždavinio tikslas yra surasti, kokia eilės tvarka turės būti atliekamos operacijos kiekvienam darbui ir tų darbų atlikimo eiliškumas kiekvienai mašinai [4][8].

Uždavinyje, kur gamyba yra iš dalies nuosekli (job shop), darbo operacijos yra atliekamos pagal tam tikrą darbo tvarką. Sakykime, mašinų skaičius $m \in Z^+$, darbų aibė J. Kiekvienas darbas $j \in J$ yra sudarytas iš operacijų $o_{i,j}, 1 \le i \le m$ sekos n_j, mašinos, kurios apdoroja kiekvieną operaciją $p_{i,j} \in [1..m]$ ir apdorojimo trukmės $l_{i,j} \in N$. Darbų aibei J sudaromas tvarkaraštis, tai tvarkaraščių rinkinys vienos mašinos. Atitiktis $f_p: \{o_{i,j}: p_{i,j} = p\} \to N$ randama kiekvienai mašinai , kur $f_p(o_{i,j}) > f_p(o_{i',j'})$ reiškia $f_p(o_{i,j}) \ge f_p(o_{i',j'}) + l_{i,j}$, o $f_p(o_{i+1,j}) \ge f_p(o_{i,j}) + l_{i,j}$. Tokio tipo tvarkaraščiao darbų pabaiga

$$\max_{j \in J} f_p\left(o_{n_j,j}\right) + l_{n_j,j}$$

Uždavinio tikslas – surasti, kokia eilės tvarka turėtų būti atliekamos operacijos ant visose mašinose [4][8].

Gamybos uždavinys, kur gamyba yra nuosekli, tai yra atskiras iš dalies nuoseklios gamybos uždavinio atvejis, kai visų darbų operacijų atlikimo tvarka yra vienoda, o atliekamų operacijų skaičius yra toks, koks yra mašinų skaičius. Sakykime, kad mašinų skaičius yra $m \in Z^+$, darbų aibė J. Kiekvienas darbas $j \in J$ sudarytas iš m operacijų $o_{i,j}, 1 \le i \le m$, operacija $o_{i,j}$ yra apdorojama i-taja mašina, o kiekvienos operacijos apdorojimo trukmė $l_{i,j} \in N$. Darbų aibei J sudarytas tvarkaraštis toks, kad kiekvienam darbui $j \in J$ ir $1 \le i \le m$, $f_{i+1}(j) \ge f_i(j) + l_{i,j}$.

Tokio tvarkaraščio darbų pabaiga:

$$\max_{j \in J} f_m(j) + l_{m,j}$$

Uždavinio tikslas – rasti darbų atlikimo eiliškumą kiekvienai mašinai [4][8].

Gamybos tvarkaraščio idėjos:

- maksimaliai sumažinti mašinų prastovų laiką - kad kiekviena mašina, atlikusi operaciją vienam darbui, kuo greičiau pradėtų atlikinėti operaciją kitam;
- kaip suplanuoti gamybą - darbų operacijų atlikimą, kad visi darbai būtų atlikti iki tam tikro pasirinkto laiko.

2. Genetinio algoritmo panaudojimas gamybinių tvarkaraščių sudarymui

2.1 Genetinis algoritmas

Genetiniai algoritmai, dažniau vadinami tiesiog GA, ir principas, pagal kurį jie sudaryti, buvo pasiūlytas XX amžiuje, Hollando. Nuo pat pirmųjų bandymų, genetiniai algoritmai sėkmingai pasirodė sprendžiant įvairius optimizavimo uždavinius, tokius kaip gamybos operacijų planavimas, elektroninių komponentų išdėstymas ir kt.

Genetinis algoritmas, tai algoritmas, pagrinde naudojamas kompiuterių moksle, ieškantis apytikrių sprendinių optimizavimo ir paieškos uždaviniuose. Genetiniai algoritmai naudoja įvairius metodus, kurie yra paremti evoliucinės biologijos principais – individų kryžminimas, mutacija, tam tikrų savybių paveldimumas.

Šie algoritmai paprastai realizuojami kaip kompiuterinė imitacija, kurioje - p dydžio populiacija, sudaryta iš abstrakčių c ilgiu sprendinių reprezentacijų x – chromosomų, vystosi ir taip bandoma surasti optimizuojamo uždavinio geresnius sprendinius. Evoliucija pradedama iš atsitiktinės chromosomų populiacijos. Kiekvienoje kartoje, skaičiavimų iteracijoje, įvertinami populiacijos nariai, tam tikri individai modifikuojami – būna sukryžminami ir/arba mutuoja – tam, kad suformuotų naują populiaciją, kuri bus tiriama kitoje iteracijoje [5].

2.1.1 Algoritmo schema

Genetinio algoritmo vykdymo etapai:

1. **Pradžia** - Sugeneruojama atsitiktinė populiacija, kurioje yra p chromosomų.
2. **Įvertinimas** - Įvertinamas populiacijos kiekvienos chromosomos tinkamumas pagal tam tikras tikslo funkcijas
3. **Nauja populiacija** – Sukuriama nauja populiacija, kartojant tokius veiksmus:
   - 3.1 **Atranka** – Pagal tinkamumą išrenkamos dvi tėvinės chromosomos.
   - 3.2 **Kryžminimas** – Iš tėvinių chromosomų sukuriama nauja chromosoma (vaikas).
   - 3.3 **Mutacija** – Pagal mutacijos tikimybę ir jos rūšį, keisti chromosomą.
   - 3.4 **Piėmimas** – Įdėti naują chromosomą į naują populiaciją.
4. **Pakeitimas** – Tolimesniuose veiksmuose naudoti naujai sugeneruotą populiaciją.
5. **Tikrinimas** – Jei pabaigos sąlyga tenkinama, algoritmas sustabdomas ir grąžinama geriausia chromosoma iš populiacijos.
6. **Ciklas** – Grįžtama į antrąjį žingsnį.

Genetinis algoritmas, tai skaičiavimų idėja, todėl kiekvieno uždavinio sprendimui, algoritmo schema gali skirtis, nuo bendrinio jos pateikimo.

2.1.2 Chromosomos kodavimas

Chromosomų kodavimas – pirmasis uždavinys su kuriuo susiduriama taikant genetinius algoritmus. Kodavimas priklauso nuo problemos specifikos, tad chromosomą galima užkoduoti labai įvairiai. Pats paprasčiausias ir dažniausiai naudojamas kodavimo būdas yra dvejetainis, binarinis (žr. 2.1.2.1 pav.). Jis buvo panaudotas pačiuose pirmuose genetinių algoritmų bandymuose.

0 1 0 0 1 1 0 1 0 0 0 1 0 0 1

2.1.2.1 pav. Binarinis chromosomos kodavimas

Perstatymas, dar vadinimas permutaciniu kodavimu, naudojamas uždaviniuose, kur aktualus rikiavimas. Pavyzdžiui, keliaujančio pirklio uždavinyje. Čia chromosoma – numerių eilutė (žr. 2.1.2.2 pav.).

1 9 7 10 5 6 2 11 8 3 4 0 12

2.1.2.2 pav. Permutacijo chromosomų kodavimo pavyzdys

Sprendžiant problemas, kur yra naudojami realūs skaičiai, labai problematiškas yra bitų kodavimas, jis sudėtingas, nepatogus ir nepraktiškas. Šio tipo uždaviniams spręsti padeda tiesioginis kodavimas. Čia genas parodo tikrą duomenų prasmę. Chromosoma atrodo taip – eilutė sudaryta iš atsitiktinių elementų (žr. 2.1.2.3 pav.).

0.24 2.45 1.02 0.1 6.45 2
(kairėn) (dešinėn) (žemyn)

2.1.2.3 pav. Tiesioginis chromosomų kodavimas

Medžių kodavimo metodas, tai dar sudėtingesnis metodas. Jis paprastai naudojamas išvystyti programas arba kokias nors išraiškas. Šiame metode chromosoma gali reprezentuoti iš programavimo kalbos funkcijų ir komandų sudarytą medį [9].

2.1.3 Kryžminimas ir mutacija

Kryžminimas ir mutacija - du pagrindiniai genetinio algoritmo operatoriai, kurie nulemia chromosomų vystymąsi. Jų pasirinkimas ir realizavimas priklauso nuo problemos, bei to, kaip yra užkoduotos chromosomos.

Binaraus kodavimo atveju kryžminime pasirenkamas vienas ar daugiau kryžminimo taškų. Jeigu yra vienas taškas, naujai chromosomai sudaryti paimama pirmoji dalis iki taško iš pirmosios tėvinės chromosomos ir likusi dalis – iš kitos (žr. 2.1.3.1 pav.).

3.1.2.1 pav. Vieno taško kryžminimas

Kodavimuose su perstatymais būti pastebėti, kad kiekvienas skaitmuo turi būti panaudotas tik vieną kartą. Vienataškis kryžminimas atrodo taip: formuojant naują chromosomą, dalis iki taško iš pirmojo tėvo yra perrašoma, o likusioji dalis yra perrašoma iš antrojo tėvo, tačiau perrašomi tiktais tie elementai, kurie nebuvo išrašyti anksčiau.

Medžių kodavimas. Abiejuose kryžminamuose medžiuose yra pasirenkamas taškas, nuo šio taško gaunamos šakos yra sukeičiamos.

Kryžminimo tikimybė dažniausiai yra pasirenkma, tačiau atliekant šį tyrimą programa yra parašyta taip, kad kryžminimo tikimybė yra 100%, tai reiškia, kad yra kryžminamos visos chromosomos. Programa yra sukurta taip, kad ji atskirai saugoja vieną tevų populiacijos geriausią chromosomą ir jeigų vaikų chromosomose nėra geresnio sprendinio, nei prieš tai surastas, atrinkta geriausia tėvinė chromosoma yra įdedama į naująją populiaciją ir su ja toliau vykdomas kryžminimas.

Kryžminimo pobūdis:

Šiama tyrime bus nagrinėjamas vieno taško kryžminimas. Principas, pagal kurį yra atliekamas kryžminimas:

Pradinės chromosomos:

7 18 8 13 16 <> 1 5 3 2 4 10 17 6 15 12 20 11 19 14 9
13 12 16 5 1 <> 17 2 9 4 3 14 7 18 11 8 15 20 19 6 10
16 9 5 14 2 <> 15 3 20 18 17 19 11 8 10 13 6 4 12 7 1

Kryžminimas atliekamas po 5 elemento

Naujos sugeneruotos chromosomos:

7 18 8 13 16 | 12 5 1 17 2 9 4 3 14 11 15 20 19 6 10
7 18 8 13 16 | 9 5 14 2 15 3 20 17 19 11 10 6 4 12 1
13 12 16 5 1 | 9 14 2 15 3 20 18 17 19 11 8 10 6 4 7

Po kryžminimo sugeneruotos populiacijos kiekio apskaičiavimo formulė:

$$\text{mm} = \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} m_{i,j}$$

Arba daug paprastesnė ir lengviau suskaičiuojama formulė:

$$\text{mm} = \frac{(\text{n}-1)^2 + \text{n} - 1}{2}$$

$$n = g + b;$$

g – atrinktų gerų chromosomų skaičius, b – atrinktų blogų chromosomų skaičius

(ką reiškia geros ir blogos chromosomos, bus paaiškinta aprašant programos veikimą).

Mutacija bitų kodavime – kiekvienam bitui yra priskiriama tikimybė, kad jis bus pakeistas priešingu – iš 1 keičiama į 0. Perstatymo metode du elementai būtų sukeitčiami vietomis. Tiesioginiame metode mutacija priklauso nuo reikšmių. Pavyzdžiui, dirbant su realiais skaičiais, mutacija gali reikšti, jog bus pridedamas ar atimamas skaičius. Priklausomai nuo uždavinio, mutacijos gali būti labai įvairios. Medžio kodavime keičiami yra mazgai [9].

Šaltiniuose teigiama, kad mutacijos tikimybė turėtų būti labai maža, maždaug 0,5-1% [9], tačiau šiame tyrime mutacijos tikimybė parinkta didesnė - 20 %.

Programoje yra užprogramuotos penkios mutacijų rūšys, jos visos bus analizuojamos, lyginamos tarpusavyje.

Mutacijos rūšys:

1) Chromosomos elementai yra atsitiktinai išmaišomi.
2) Sukeičiami pirmas ir paskutinis chromosomos elementai.
3) Sukeičiami antras ir prieš paskutinis chromosomos elementai.
4) Pirmas chromosomos elementas yra pakeičiamas priešpaskutiniu, priešpaskutinis – paskutiniu, paskutinis – antru, antras – pirmu.
5) Sumaišomi pirmi devyni chromosomos elementai.

2.1.4 Selekcija

Selekcija, tai operatorius, kuris nusako, kokios tėvinės chromosomos bus paimtos tolimesnei evoliucijai. Selekcija gali būti keičiama pagal pasirinkimą [9].

Pavyzdžiai:

- Č.Darvino teorija teigia, didesnę tikimybę išlikti, būti išrinktomis turi tos chromosomoms, kurios randa geresnius uždavinio sprendinius,
- Pastovios būsenos atranka – siekiama išlaikyti senąją populiaciją, o pakeisti tik blogesnes chromosomas naujai sukurtomis. Tokiu būdu išsaugomi geriausi sprendiniai,
- Elitizmas – metodas, kur senoji populiacija yra visiškai pakeičiama naujai gautosios, tačiau tam kad neprarasti galimo geriausio sprendinio, geriausias sprendinys iš senosios populiacijos perkeliamas į naująją.

3. Genetinių algoritmų parametrų įtakos tyrimas

3.1 Sukurta programa

3.1.1 Programos, tyrimo tikslai

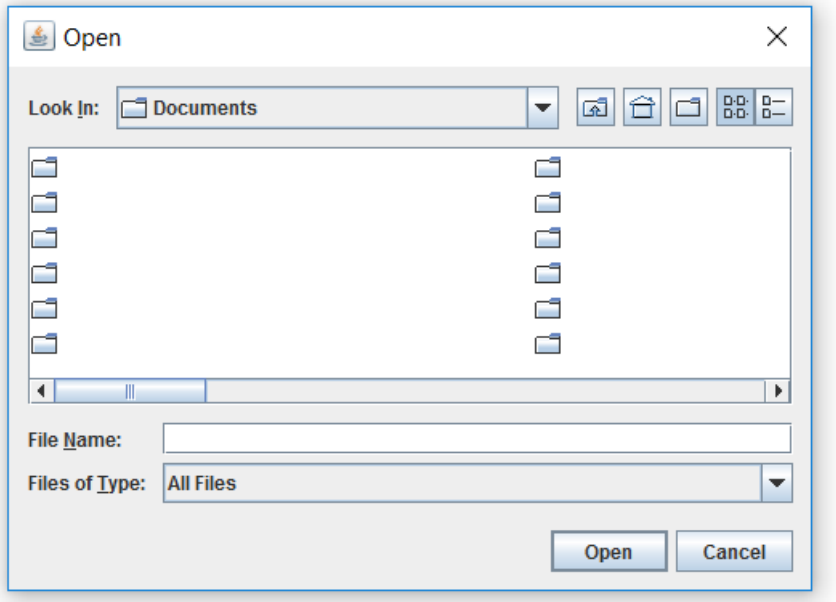
a) Palyginti, kuri mutacija iš pateikiamų penkių, suranda geriausią sprendinį. Jeigu įmanoma, ištinkti geriausią mutacijos rūšį,

b) Ištirti, ar gaunamas geresnis rezultatas, kai atliekamų iteracijų kiekis yra didinamas (tytimo metu skaičiavimai atliekami bus su 200 ir 500 iteracijų) ir nustatyti optimalų iteracijų skaičių,

c) Palyginti anksčiau gautus rezultatus ir rezultatus, kai keičiama tikslo funkcija. Pirmuoju atveju tikslo funkcija – prastovų minimizavimas, antruoju – gamybos trukmės minimizavimas.

3.1.2 Duomenų failo paruošimas

Įjungus programą, atsiranda pasirinkimo langas – reikia pasirinkti duomenų failą (žr. 3.1.2.1 pav.). (Aplankų pavadinimai buvo pašalinti).



3.1.2.1 pav. Duomenų failo pasirinkimas

Kad programa galėtų apdoroti dirbti, duomenys turi būti pateikti Excel faile, kuris turi būti išsaugotas („Save As“) Excel 97-2003 Workbook formate. Tyrime naudojamo duomenų failo pavadinimas - „Duomenys.xls“.

Duomenų failas duomenys turi būti pateikti pagal šiuos reikalavimus:

a) Jame turi būti šeši „Sheet“, lapai.

b) Jų pavadinimai: pirmasis – „Uzsakymai“, antrasis – „Techologija“, trečiasis – „OperMedis“, ketvirtasis – „OperMetalas“, penktasis – „OperAbu“, šeštasis – „Stakles“.
Visi pavadinimai turi būti parašyti lotyniškomis raidėmis – negali būti lietuviškų raidžių - ą, š ir kitų (esant lietuviškiems simboliams, programoje atsiranda klaidų).

c) Lape „Uzsakymai“ duomenys turi būti pateikti tokia tvarka: Pirma eilutė – Antraštės. Stulpeliai: Gaminio pavadinimas, Gaminio kodas, Gaminių kiekis, Mato vienetas (žr. 3.1.2.2 pav.).

Gaminio pavadinimas	Gaminio kodas	Gaminių kiekis	Mato vienetas
Plinth 95mm height	_73 PL 08/95 TT	1	m
Table small with 1 drawer for table Men Casual 1000x500x500mm, HPL H1599,	_73 T 06-1 M TT_1000	1	vnt
Table Men Casual 1200x1200x800mm, white 101PR	_73 T 09 M TT_1200	1	vnt
Table Men Casual 1600x800x800mm, white 101PR	_73 T 09 M TT_1600	1	vnt
Top cube for table Men Casual, Fikk brown, 600x400x200mm	_73 TPD 01-1 M TT'	1	vnt
Bench small for table Men Casual 460x460x420mm, leather grey	_73 BE 01 M TT_460	1	vnt
Bench for table Men Casual 1400x500x500mm, leather grey	_73 BE 01 M TT_1400	1	vnt
Gaminys8	Kodas Gam-a	1	vnt
Gaminys9	Kodas Gam-b	1	vnt
Gaminys10	Kodas Gam-c	1	vnt
Gaminys_11	Kodas Gam-11	1	vnt

3.1.2.2 pav. Duomenų failo lapas „Uzsakymai“

d) Lape „Technologija“ duomenys pateikiami tokia tvarka: Pirma eilutė – Antraštės. Stulpeliai: Gaminio kodas, Technologija, Gaminimo laikas min (žr. 3.1.2.3 pav.). Duomenys turi būti suvesti pagal gaminiui pagaminti atliekamų operacijų seką.

Gaminio kodas	Technologija	Gaminimo laikas min
_73 PL 08/95 TT	PjMed	23
_73 PL 08/95 TT	Pres	35
_73 PL 08/95 TT	MedAC	25
_73 PL 08/95 TT	ParAp	27
_73 PL 08/95 TT	Apd	35
_73 PL 08/95 TT	Kompl	23
_73 PL 08/95 TT	Laz	25
_73 PL 08/95 TT	Frez	37
_73 PL 08/95 TT	Tekin	30
_73 PL 08/95 TT	Suvirin	27
_73 T 06-1 M TT_1000	PjMed	35
_73 T 06-1 M TT_1000	Pres	65
_73 T 06-1 M TT_1000	Briau	37
_73 T 06-1 M TT_1000	MedAC	54
_73 T 06-1 M TT_1000	Surin	85
_73 T 06-1 M TT_1000	Kompl	51
_73 T 06-1 M TT_1000	Laz	50
_73 T 06-1 M TT_1000	Frez	33
_73 T 06-1 M TT_1000	Slif/pol	30
_73 T 09 M TT_1200	PjMed	25
_73 T 09 M TT_1200	Briau	37

Stulpelyje „Technologija“ parašoma turima atlikti operacija. Gaminimo laikas min – nurodoma trukmė, kiek laiko bus atliekama konkreti operacija konkrečiam gaminiui.

3.1.2.3 pav. Duomenų failo lapas „Technologija“

e) Lape „OperMedis“ turi būti pateikiami operacijų, atliekamų tiktais su staklėmis, skirtomis dirbti su medinėmis detalėms, trumpiniai. Turi būti išlaikomas bendrinis operacijų atlikimo eiliškumas (žr. 3.1.2.4 pav.).

Operacijos is eiles
PjMed
Pres
Briau
MedAC
ParAp
Apd

PjMed – medinės dalies išpjovimas
Pres – presavimas
Briau – briaunavimas
MedAC – medienos apdirbimo centrai
ParAp – paruošimas apdaila
Apd – apdaila

3.1.2.4 pav. Duomenų failo lapas „OperMedis“

Medienos apdirbimo centruose yra atliekamos kelios operacijos, tokios kaip – gręžimas, frezavimas, kalibravimas ir kitos. Operacijas PjMed, Pres, MedAC atlieka staklės, o operacijas ParAp ir Apd atlieka darbuotojai.

f) Lape „OperMetalas“ turi būti pateikiami operacijų, atliekamų tiktais su staklėmis, skirtomis dirbti su metalinėmis detalėms, trumpiniai. Turi būti išlaikomas bendrinis operacijų atlikimo eiliškumas (3.1.2.5 pav.).

Operacijos is eiles
Laz
PjMet
Frez
Tekin
Suvirin
Slif/pol

Laz – detalių pjovimas iš metalo lakšto
PjMet – detalių pjovimas iš ilginių metalinių medžiagų, tokių kaip vamzdžiai, strypai, profiliai.
Frez – metalinių detalių frezavimas
Tekin – strypų ir vamzdžių tekinimas
Suvirin – suvirinimas
Slif/pol – šlifavimas ir poliravimas

3.1.2.5 pav. Duomenų failo lapas „OperMetalas“

Operacijos Laz, Frez atliekama staklės, Tekin ir Slif/Pol gali atlikti tiek staklės, tiek darbuotojai, Suvirin atlieka tik darbuotojai.

g) Lape „OperAbu“ turi būti pateikiami operacijų, kurios atliekamos visų tipų detalėms ir pusfabrikačiams, trumpiniai. Turi būti išlaikomas bendrinis operacijų atlikimo eiliškumas eiliškumas (3.1.2.6 pav.).

Operacijos is eiles
Surin
Kompl

Surin – surinkimas atskirų detalių į pusfabrikačius arba galutinius gaminius.

Kompl – komplektacija

3.1.2.6 pav. Duomenų failo lapas „OperAbu“

h) Lape „Stakles“ pateikiamas sąrašas staklių, ant kurių bus atliekamos operacijos. (3.1.2.8 pav.).

Stakles
B1.PjMed_1
B1.PjMed_2
B1.PjMed_3
B1.Pres_1
B1.Pres_2
B1.Briau_1
B1.Briau_2
B1.MedAC_1
B1.MedAC_2
B1.MedAC_3
B1.MedAC_4
B1.ParAp_1
B1.ParAp_2
B1.Apd_1
B1.Apd_2
MC.Laz_1
MC.Laz_2
MC.PjMet_1
MC.PjMet_2
MC.Frez_1
MC.Frez_2
MC.Frez_3
MC.Tekin_1
MC.Tekin_2
MC.Suvirin_1
MC.Suvirin_2
MC.Slif/pol_1
MC.Slif/pol_2

Čia yra aprašomos visos staklės, su kuriomis bus dirbama šiame tyrime. Programa darbus skirstys ant šių staklių ir likusios dalies, kuri nesimato.

Yra lengvai pastebima, kad prie anksčiau aptartų medžio ir metalo operacijų, pradžio yra trumpi prierašai „B1“ ir „MC“. Šie prierašai indikuoja kur yra staklės ar darbo centrai, kur dirba darbuotojai.

B1 – Baldų pirmasis cechas, MC – Mechaninis cechas.

Operacijos metalui nėra atliekamos baldų ceche – ten nėra įrangimų, tinkamų dirbti su metalu. Taip pat ir mechaniniame ceche dirba yra tik su metalinėmis detalėmis.

Bendrosios operacijos atrodo taip: B1.Surin_X , B1_Kompl_X . Šiuo atveju, B1 indikuoja, kurioje vietoje vyksta pusfabrikačių ir pilnų gaminių surinkimas.

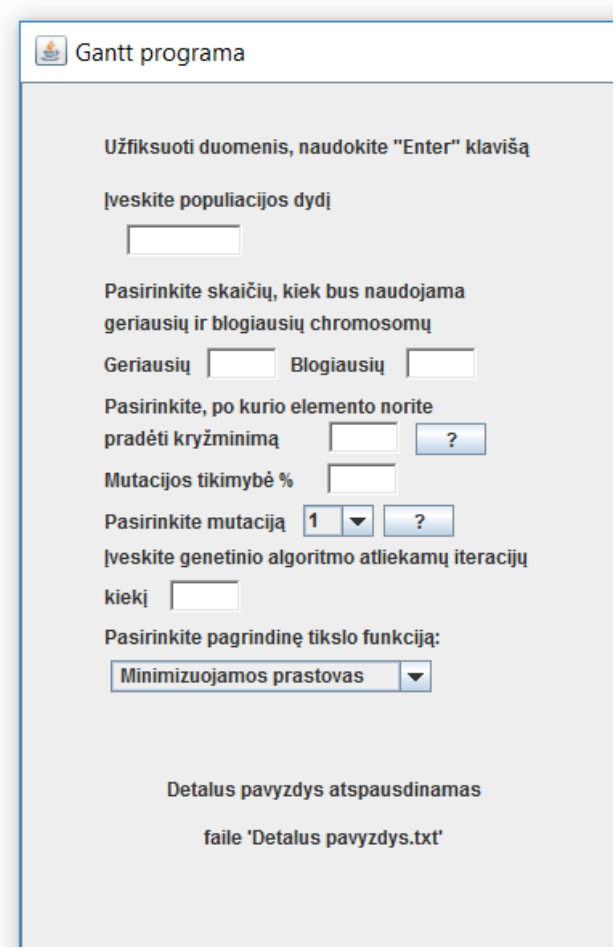
Prierašai pabaigoje indikuoja, kiek bus yra darbo centrų ar darbuotojų atlikti kiekvienai operacijai. Matome, kad operacijoms atlikti yra skirtingas kiekis darbo centrų.

3.1.2.8 pav. Duomenų failo lapas „Stakles“

3.1.3 Vartotojo sąsaja, programos paaiškinimai

Pasirinkus duomenų failą ir paspaudus mygtuką „Open“, atsiras programos langas (žr. 3.1.3.1 pav.).

(Pateikiama lango dalis, kur įvedami duoments, o ne visas langas. Įsijungus programą, daugiau įrašo nėra matoma).



Pasirinktiems duomenims išsaugoti, naudokite „Enter“ klavišą.

1) Įvedamas pradžioje apdorojamos populiacijos dydis,
2) Įvedama su kiek gerų ir blogų chromosomų bus toliau dirbama,
3) Įvedama po kurio elemento norima pradėti kryžminimą,
4) Įvedama mutacijos tikimybė,
5) Pasirenkama mutacijos rūšis,
6) Įvedamas atliekamų iteracijų skaičius,
7) Pasirenkama tikslo funkcija pagal kurią bus atliekami skaičiavimai: Minimizuojamos prastovos arba minimizuojama gamybos trukmė.

*Įvedus duomenis ir užfiksavus juos „Enter“ Klavišu, arba pasirinkus juos per „ComboBox“ Įvedimo, pasirinkimo langelis papilkuos – duomenų keisti nebus galima.

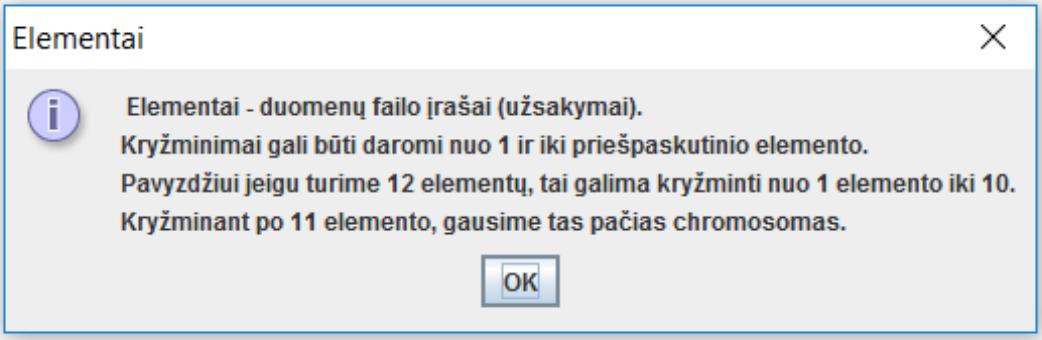
3.1.3.1 pav. Duomenų suvedimas

Programos veikimas:

- Įvestas populiacijos dydis, tai populiacija, kuri bus apdorojama pačioje pradžioje, vieną kartą - iš jos atrenkamas geriausių ir blogiausių chromosomų kiekis (nurodomas vėliau). Kitaip tariant – tai yra pirminis chromosomų rinkinys, nuo kurio pradedame darbą.
- Generuojama populiacija, su kuria bus dirbama algoritmo vykdymo metu. Gerų chromosomų skaičius – tai skaičius, kiek bus atrinkta geriausių chromosomų iš pirmosios chromosomų populiacijos. Atliekant pirmą antrą ir kitas iteracijas, renkamasi iš genetinio algoritmo sugeneruotos populiacjos. Blogų chromosomų skaičius – tai skaičius, kieks bus atrinkta blogiausių chromosomų iš populiacijos. Kadangi nežinome ar geriausias sprendinys bus gautas kryžminant tik geriausias chromosomas, dėl to į algortimo skaičiavimus įtraukiame ir nurodytą kiekį blogiausių chromosomų. Jeigu blogiausių chromosomų skaičius lygus 0, bus atliekamas elitinių chromosomų atrinkimo metodas (dirbama tik su geriausiomis chromosomomis).

- Parenkama, po kurio elemento bus daromas kryžminimas. Elementų skaičius – įrašų skaičius duomenų failo lape „Uzsakymai“. Jeigu duomenų faile 5 įrašai, tuomet kryžminimas turės prasmę tik 3 atvejais – po pirmojo, antrojo ir trečiojo elementų. Atliekant kryžminimą po ketvirtojo elemento, yra tik vienas elementas, kurį reikia atrinkti, tai reiškia, kad antroje tėvinėje chromosomoje jis yra toks pat kaip ir pirmoje, todėl nauja chromosoma bus identiška pirmajai.

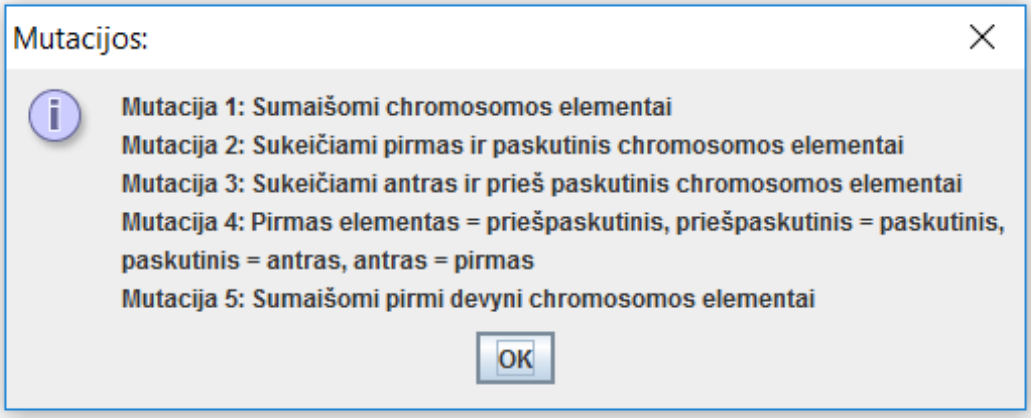
Paspaudus „?“ simboliu pažymėtą mygtuką, pateikiamas detalesnis paaiškinimas (žr3.1.3.2pav.).



3.1.3.2 pav. Kryžminimo paaiškinimas

- Įvedama kokia yra procentinė tikimybė, kad įvyks mutacija.
- Perenkama, su kuria mutacija iš penkių galimų norite atlikti skaičiavimus.

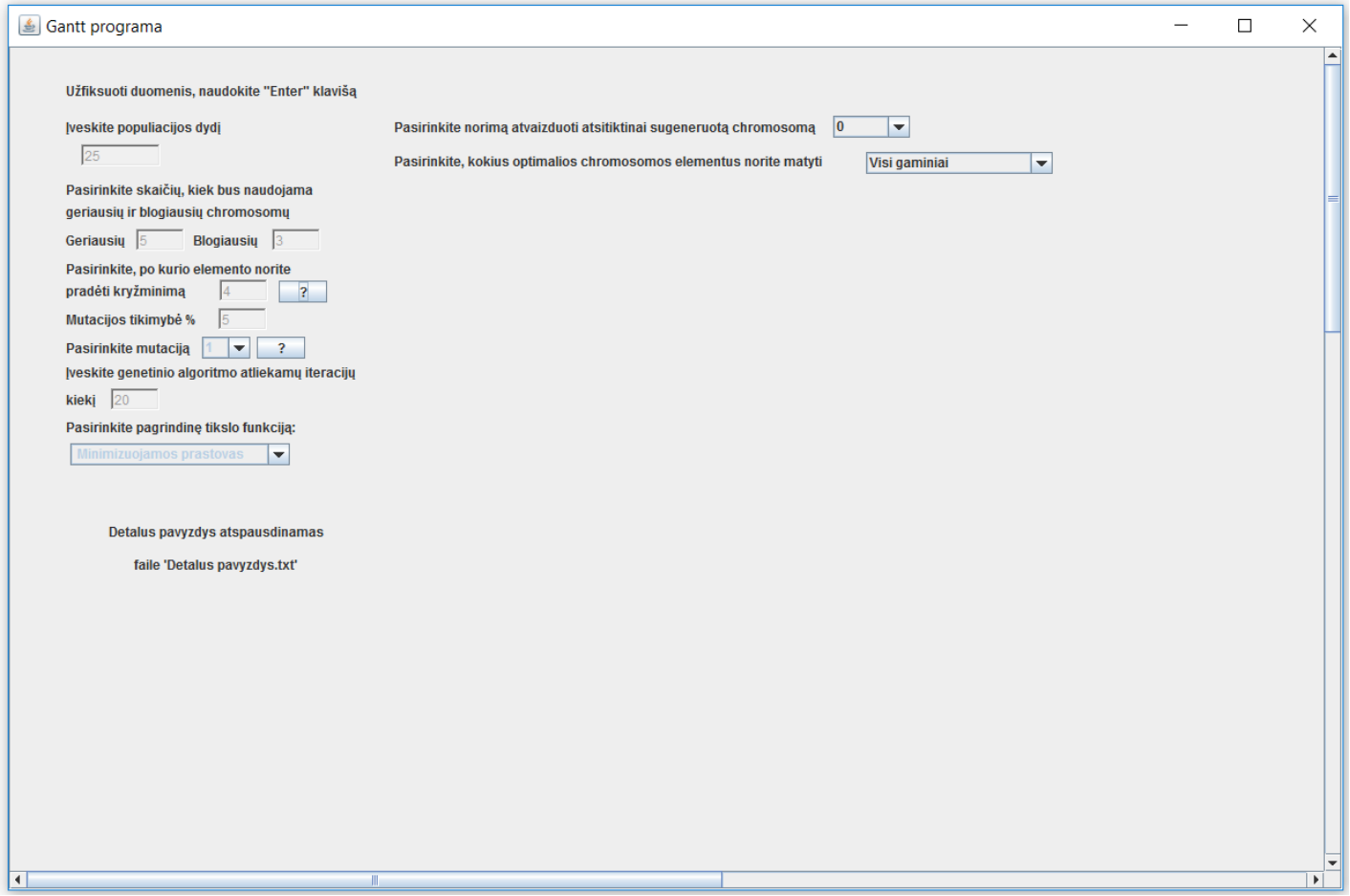
Paspaudus „?“ simboliu pažymėtą mygtuką, pateikiamas paaiškinimas, kuo tarpusavyje skiriasi šios mutacijos (žr. 3.1.3.3pav.).



3.1.3.3 pav. Mutacijų paaiškinimai

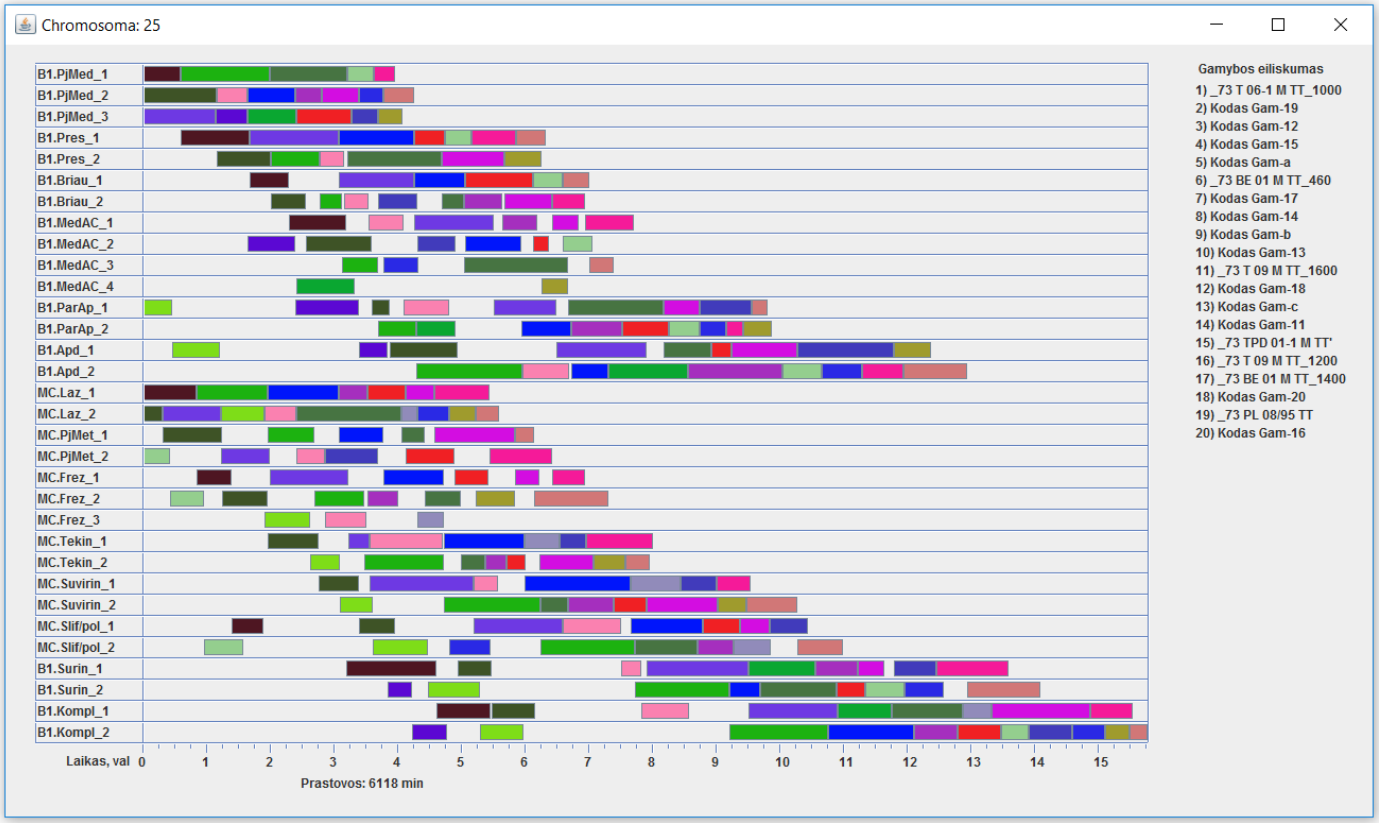
- Parenkama, kiek genetinio algoritmo iteracijų reiėks atliekti.
- Pasirenkama tikslo funkcija:
  a. Minimizuoti prastovas – tikslas sukurti tokį darbų atlikimo tvarkaraštį, kad staklių nedarbo, prastovų laikas būtų mažiausias,
  b. Minimizuoti gamybos trukmę - tikslas sukurti tokį darbų atlikimo tvarkaraštį, kad visi gaminiai būtų pagaminti per kuo trumpesnį laiko tarpą.

Visus duomenis užfiksavus „Enter“ klavišu ir pasirinkus tikslo funkciją, visi duomenų įvedimo ir pasirinkimo laukeliai bus papilkavę (neredaguojami) ir prasidės skaičiavimai. Atlikus skaičiavimus, programos lango kairėje pusėje apačioje atsiras pranešimas, kad detalus pavyzdys, kuriame pateikiami skaičiavimų žingsniai, yra atspausdintas tekstiniame faile. Taip pat, dešinėje pusėje nuo duomenų įvedimo laukų atsiras nauji pasirinkimai. Programos lango vaizdas, atlikus skaičiavimus, atrodo taip (žr. 3.1.3.4 pav.).



3.1.3.4 pav. Programos langas atlikus skaičiavimus

Antraštės „Pasirinkite norimą atvaizduoti atsitiktinai sugeneruotą chromosomą“ dešinėje pusėje, galime pasirinkti pavaizduoti bet kurią chromosomą iš pačios pirmosios populiacijos, kurios kiekį nurodėme pačioje pradžioje. Šiuo atveju populiacijos sudaryta iš 25 chromosomų. Tarkime pasirenkame 10-tą chromosomą (žr. 3.1.3.5 pav.).



3.1.3.5 pav. Pasirinktos chromosomos Gantt‘o diagrama ir gaminių kodų sąrašas

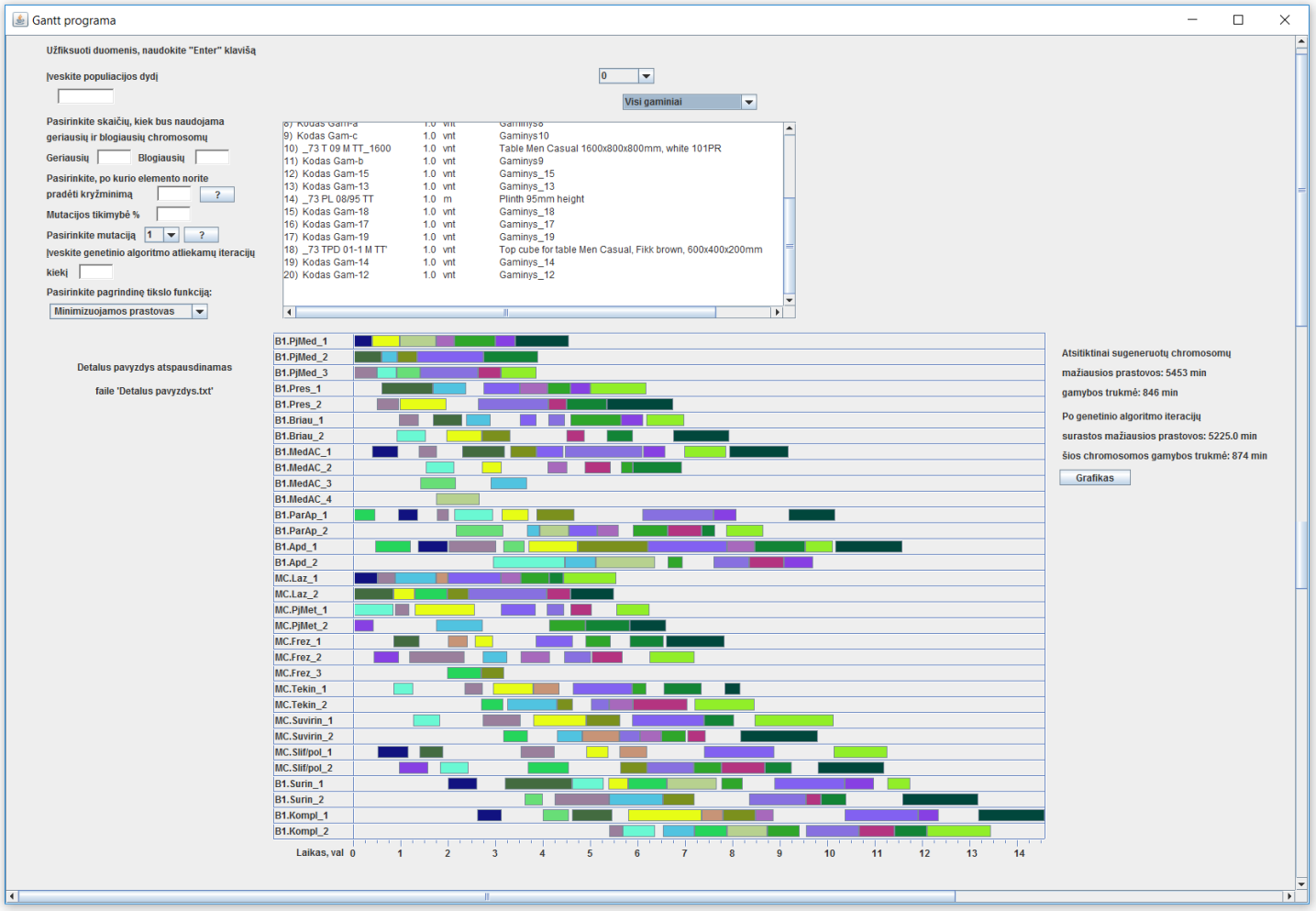
Pateikiama „Gantt“ diagrama pasirinktos chromosomos iš pirmosios populiacijos. Kairėje pusėje yra visų darbo centrų sąrašas, dešinėje - surašomi visų gaminių kodai. X- ašis, tai laikas, kiek laiko užtruks pagaminti visus gaminius. Po diagrama yra pateikiama, kokia bus pasirinktos chromosomos prastovų trukmė. Jei pasirinksime antrąją tikslo funkciją – minimizuoti gamybos trukmę, po diagrama bus pateikiama pasirinktos chromosomos gamybos trukmė minutėmis.

Antraštės „Pasirinkite, kokius optimalios chromosomos elementus norite matyti“ dešinėje pusėje, galime pasirinkti kokius surastos geriausios chromosomos (pagal atliktus genetinio algoritmo skaičiavimus) elementus norime matyti.

„Visi gaminiai“ – bus atvaizduojama visa informacija iš duomenų failo pirmojo lapo „Uzsakymai“.

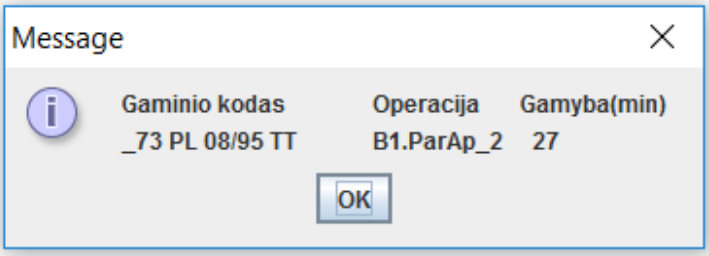
Gaminio kodas – parodoma informacija apie pasirinktą gaminį. Kodo, pavadinimas, kiekis, atliekamų tam gaminiui pagaminti operacijų sąrašas ir jų atlikimo trukmė.

Abiem atvejais taip pat bus atspausdinama optimalios chromosomos Gantt diagrama. Dešinėje jos pusėje pateikiamas minimalus prastovų skaičius iš 25 atsitiktinai sugerneruotų chromosomų ir optimaliausios chromosomos prastovų skaičius. Taip pat yra pateikiami antrosios tikslo funkcijos duomenys (žr. 3.1.3.6pav.). Pasirinkus kitą tikslo funkciją, duomenys ir komentarai bus atitinkamai kitokie.
*Pradžioje sugeneruotos chromosomos prastovos gali būti mažesnės, nei atlikus tam tikrą iteracijų skaičių. Taip nutikti gali todėl, kad tyrimo duomenys pradedami registruoti atlikus pirmąją iteraciją.



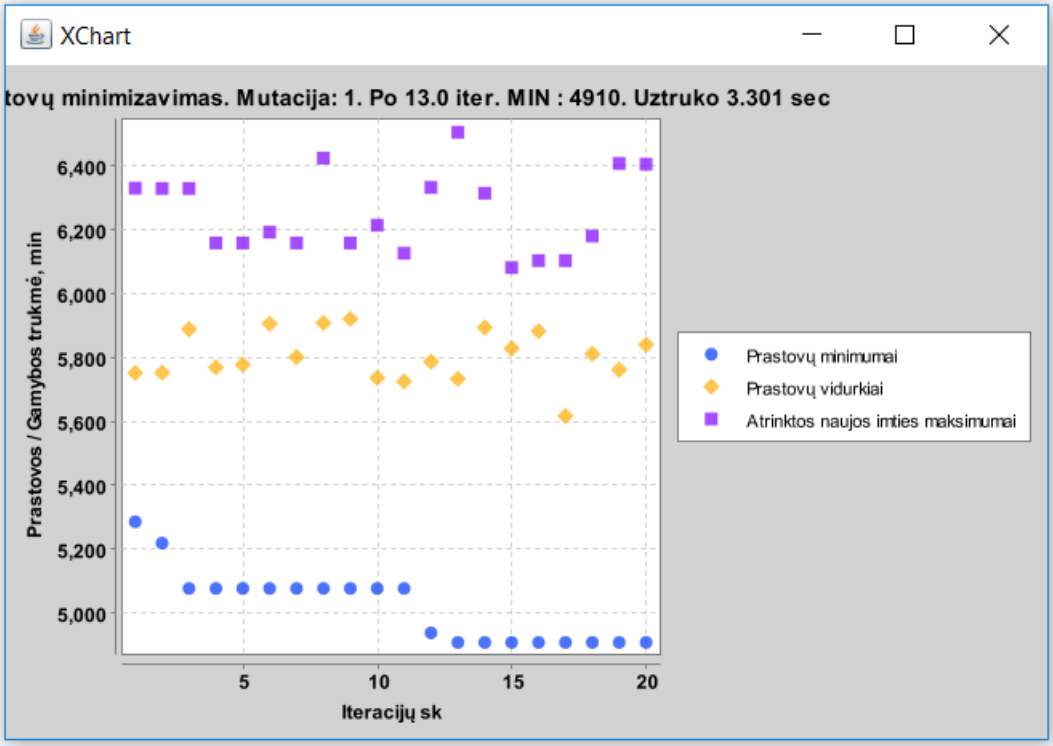
3.1.3.6 pav. Programos langas, atvaizdavus optimalią rastą chromosomą

Paspaudus ant bet kurio Gantt diagramos elemento, atsiras aiškinamasis pranešimas (žr. 3.1.3.7pav.). Jame nurodomas pasirinkto gaminio kodas, kokia operacija ir kuriame darbo centre suplanuoti atlikti ir kokia šios opercijos trukmė minutėmis.



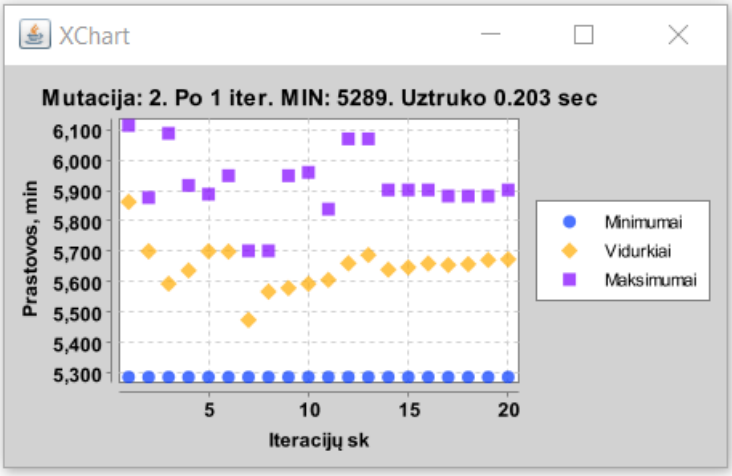
3.1.3.7 pav. Gantt diagramos elemento aiškinamasis pranešimas

Paspaudus mygtuka „Grafikas“, nubraižomas prastovų minimumų ir vidurkių grafikas (žr. 3.1.3.8 pav.). Grafikas braižytas ne pagal anksčiau pavaizduotus į programą suvestus duomenis. Tyrime bus atliekamas didesnis iteracijų kiekis.

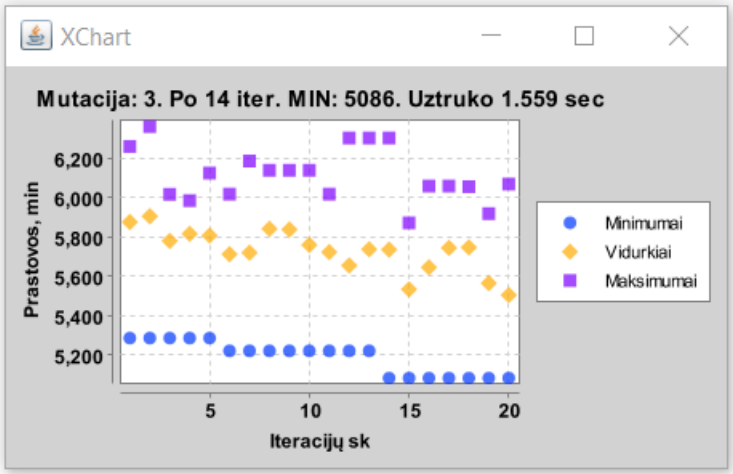


3.1.3.8 pav. Prastovų vidurkių ir minimumų grafikas

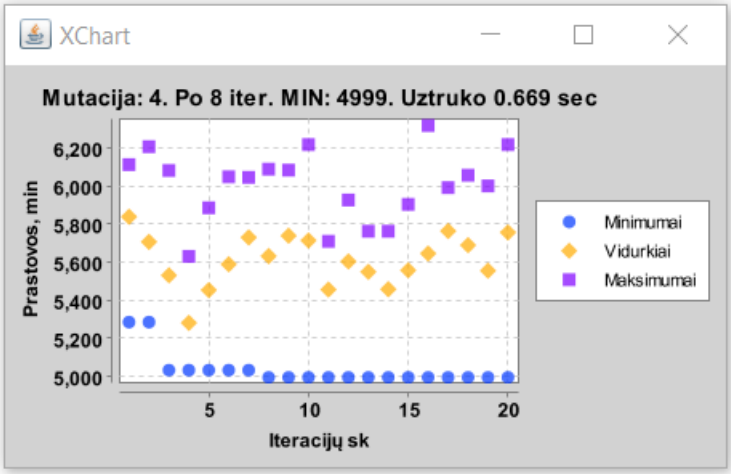
Grafiko viršuje pateikiama: Pasirinkta tikslo funkcija (šiek tiek „nukąsta“ pradžia dėl grafiko „lango“ dydžio), mutacija, su kuria buvo atliekami skaičiavimai, po kelintos iteracijos buvo rastas geriausias sprendinys, rastas sprendinys ir kiek laiko užtruko skaičiavimai, kol buvo gautas šis rezultatas.
*Neišjunkite grafikų per „X“, nes tuomet išsijungs visa programa. Nuleiskite langą žemyn.
Atspausdinus grafiką, pagrindiniame programos lange, šalia mygtuko „Grafikas“ atsiras mygtukas „Mutacijų lyginimas“. Paspaudus jį, bus apskaičiuojamas visas algoritmas iš naujo, su tokia pačia pradine populiacija, tačiau vis su kita mutacija. Grafikai pateikti sekančiuose keturiuose paveikslėliuose.



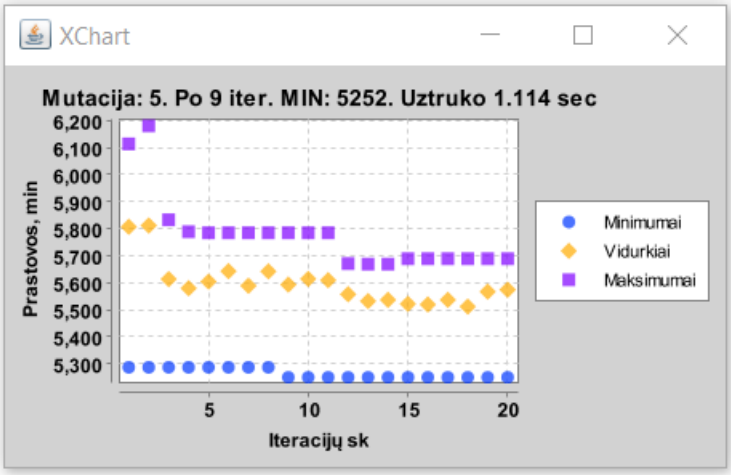
3.1.3.9 pav. Antrosios mutacijos rezultatų grafikas



3.1.3.10 pav. Trečiosios mutacijos rezultatų grafikas



3.1.3.11 pav. Ketvirtosios mutacijos rezultatų grafikas



3.1.3.12 pav. Penktosios mutacijos rezultatų grafikas

Grafikuose mėlynais rutuliukais žymimas randamas geriausias sprendinys, geltoni rombai rodo sugeneruotų populiacijų tikslo funkcijų reikšmių vidurkius, o violetiniai kvadratukai – blogiausias tikslo funkcijų reikšmes kiekvienoje sugeneruotoje populiacijoje.

3.2 Sukuriamas rezultatų .txt failas

Atlikus skaičiavimus su pasirinkta mutacija, yra sukuriamas rezultatų failas (žr. 3.2.1 pav.).

Faile pateikiama:

- Visos atsitiktinai sugeneruotos chromosomos ir jų tikslo funkcijos reikšmės.
- Iš sugeneruotų chromosomų mažiausią tikslo funkcijos reikšmę turinčios chromosomos numeris, tikslo funkcijos reikšmė ir chromosomos elemetų eiliškumas.
- Iteracijos numeris.
- Atrinktų gerų ir blogų chromosomų sąrašas.
- Po kryžminimo sugeneruotų chromosomų skaičius ir pačios chromosomos.
- Naujai sugeneruotų chromosomų sąrašas su šių chromosomų tikslo funkcijos reikšmėmis po mutacijos.
- Iš naujai sudaryto sąrašo surastos mažiausios prastovos ir bendras viso sąrašo prastovų vidurkis.

Failo pabaigoje pateikiama per visas algoritmo iteracijas surasta mažiausia tikslo funkcijos reikšmė (abi tikslo funkcijos minimizuoja) ir tos chromosomos eiliškumas.

*Tose vietose, kur daug taškų, yra daugiau įrašų. Čia pateikti principai, kaip sudarytas rezultatų failas.

```
        Sugeneruotu chromosomu, ju eiliskumo ir prastovu(min) sarasas:

Chromosoma 0:     10 5 12 19 17 6 1 20 11 7 15 13 3 14 18 16 4 8 2 9       Prastovos:  5975
Chromosoma 1:     7 13 16 14 5 11 18 20 2 8 15 10 19 1 3 17 9 6 4 12       Prastovos:  6142
Chromosoma 2:     18 19 7 10 5 15 12 4 13 6 17 8 9 16 2 14 11 3 1 20       Prastovos:  6158
Chromosoma 3:     18 5 6 4 17 12 19 15 7 20 2 10 11 1 9 16 8 13 14 3       Prastovos:  5815
Chromosoma 4:     1 12 3 15 2 5 4 11 16 6 17 13 14 7 18 10 9 19 20 8       Prastovos:  6121
.....................................................................................................

        Maziausios prastovos: 5289      Chromosomos nr: 27
        Uzsakymu gamybos eiliskumas: 10 11 9 12 8 19 4 7 14 18 16 5 6 2 17 20 3 15 13 1
        Bendras prastovu vidurkis: 5922.00

|||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||

                        ITERACIJOS NR: 1

                    Geru chromosomu: 3        Gaminiu eiliskumas
Prastova 5445 Chromosomos indeksas 5            15 16 19 9 12 14 6 10 1 17 5 8 18 2 3 20 4 7 11 13
Prastova 5557 Chromosomos indeksas 6            5 19 14 10 18 20 17 9 16 6 8 3 7 1 2 12 4 15 11 13
Prastova 5616 Chromosomos indeksas 8            1 3 16 15 13 5 8 10 6 18 20 11 9 4 7 17 2 19 14 12

                    Blogu chromosomu: 2
Prastova 6328 Chromosomos indeksas 9            17 7 1 8 12 14 15 2 6 13 4 16 11 18 19 10 9 3 20 5
Prastova 6158 Chromosomos indeksas 2            18 19 7 10 5 15 12 4 13 6 17 8 9 16 2 14 11 3 1 20

-----------------------------------------------------------------------------------------------------

        Po kryzminimo sugeneruotu chromosomu skaicius: 10

15 16 19 9 12 5 14 10 18 20 17 6 8 3 7 1 2 4 11 13
15 16 19 9 12 1 3 13 5 8 10 6 18 20 11 4 7 17 2 14
15 16 19 9 12 17 7 1 8 14 2 6 13 4 11 18 10 3 20 5
15 16 19 9 12 18 7 10 5 4 13 6 17 8 2 14 11 3 1 20
.....................................................................................................


-----------------------------------------------------------------------------------------------------

        Pasirinktos mutacijos tikimybe: 20 %

        15 16 19 9 12 5 14 10 18 20 17 6 8 3 7 1 2 4 11 13      Prastovos:     5748
        15 16 19 9 12 1 3 13 5 8 10 6 18 20 11 4 7 17 2 14      Prastovos:     5916
        15 16 19 9 12 17 7 1 8 14 2 6 13 4 11 18 10 3 20 5      Prastovos:     5902
Mutavo  6 1 12 13 11 14 16 7 9 10 5 15 17 18 3 19 20 2 8 4      Prastovos:     6230
        5 19 14 10 18 1 3 16 15 13 8 6 20 11 9 4 7 17 2 12      Prastovos:     5743
        5 19 14 10 18 17 7 1 8 12 15 2 6 13 4 16 11 9 3 20      Prastovos:     6004
        5 19 14 10 18 7 15 12 4 13 6 17 8 9 16 2 11 3 1 20      Prastovos:     6116
        1 3 16 15 13 17 7 8 12 14 2 6 4 11 18 19 10 9 20 5      Prastovos:     5808
        1 3 16 15 13 18 19 7 10 5 12 4 6 17 8 9 2 14 11 20      Prastovos:     5920
        17 7 1 8 12 18 19 10 5 15 4 13 6 9 16 2 14 11 3 20      Prastovos:     6129

        Maziausios prastovos: 5289
        Bendras prastovu vidurkis: 5951.60
```

3.2.1 pav. Rezultato failo fragmentai

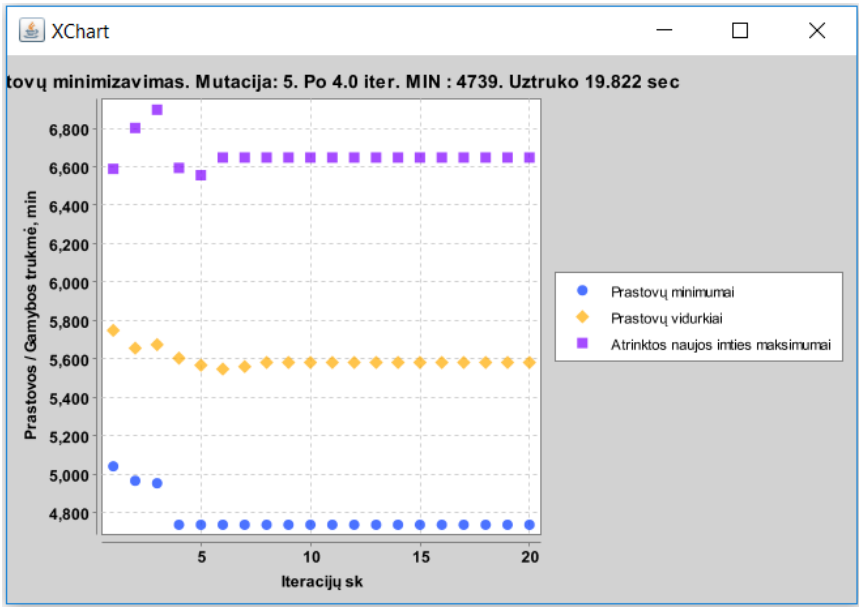
3.3 Tyrimo rezultatai

3.3.1 Parinkti skaičiavimų duomenys

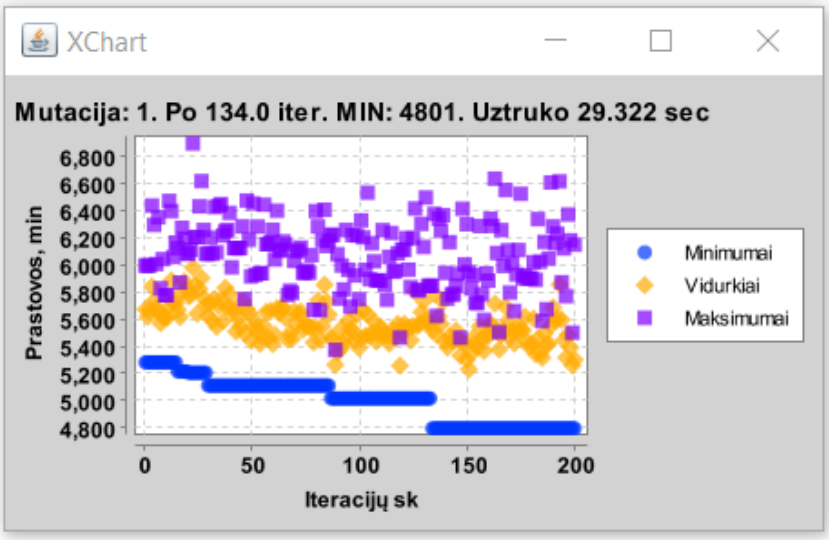
Tyrimo metu naudojamas pradinės populiacijos dydis 30 chromosomų. Darbinė populiaciją bus naudojama nedidelė, jos chromomų skaičius – 10 [9]. Darbinė populiacija naudojama nedidelė, nes duomenų faile yra 20 įrašų ir didinant darbinę populiaciją labai sparčiai auga skaičiavimų laikas. Ieškant optimalaus sprendinio, naudosime 4 geriausias chromosomas ir 1 blogiausią. Mutacijos tikimybė nustatoma – 20%. Kryžminima bus po 5 elemento.

3.3.2 Kryžminimo ir mutacijų grafikai

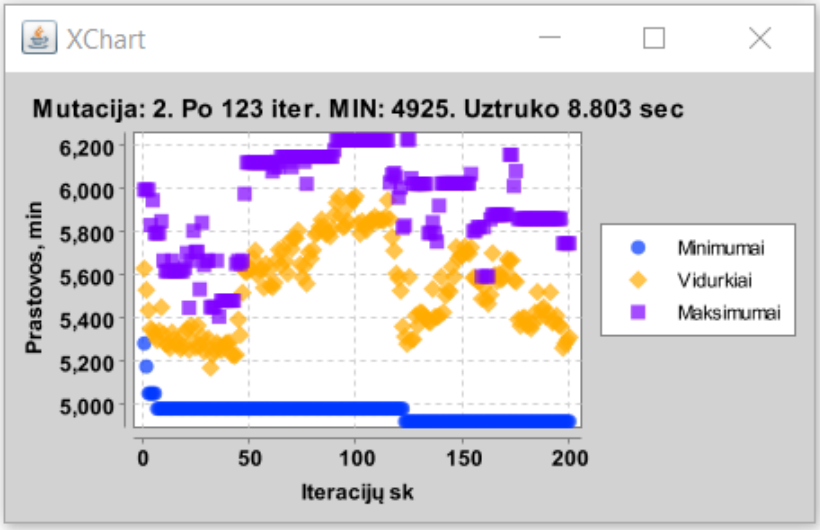
Toliau bus atskirai pateikiami kryžminimo ir mutacijų grafikai, kur atliekamas tiktais chromosomų kryžminimas arba tiktais chromosomų mutacijos (žr. 3.3.2.1 - 3.3.2.6 pav.).



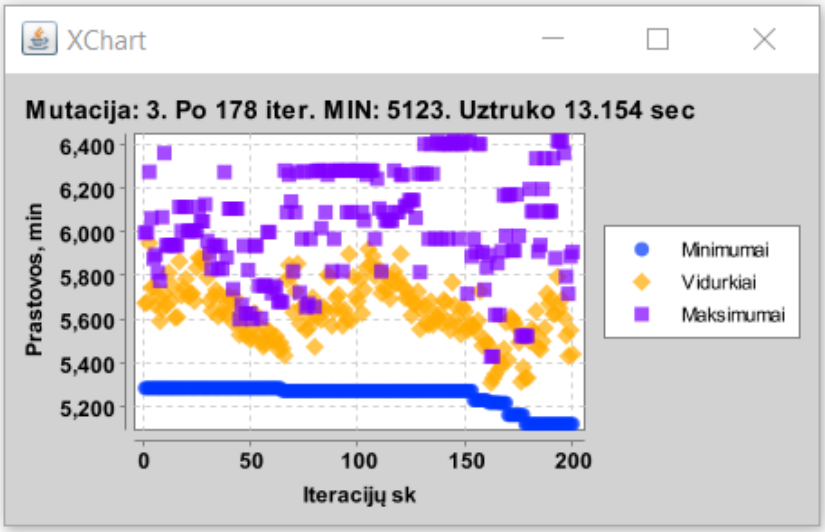
3.3.2.1 pav. Atliekamas tiktais chromosomų kryžminimas



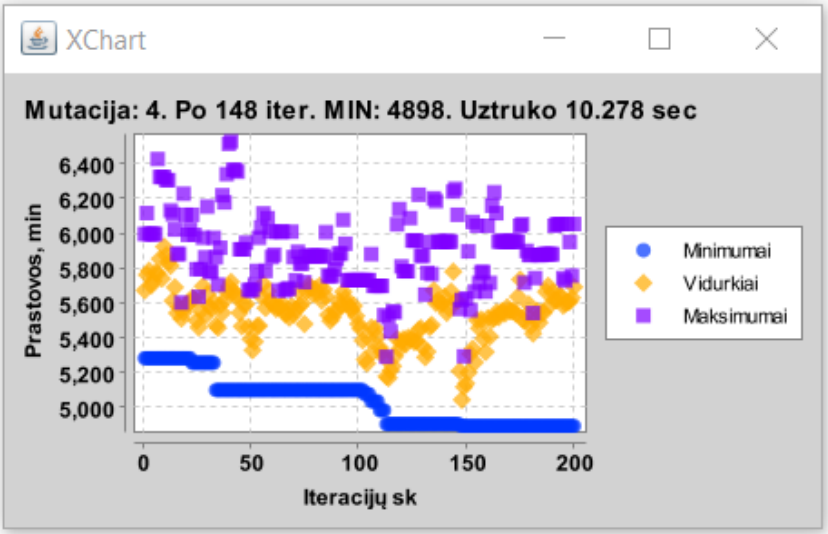
3.3.2.2 pav. Atliekama pirmoji mutacija



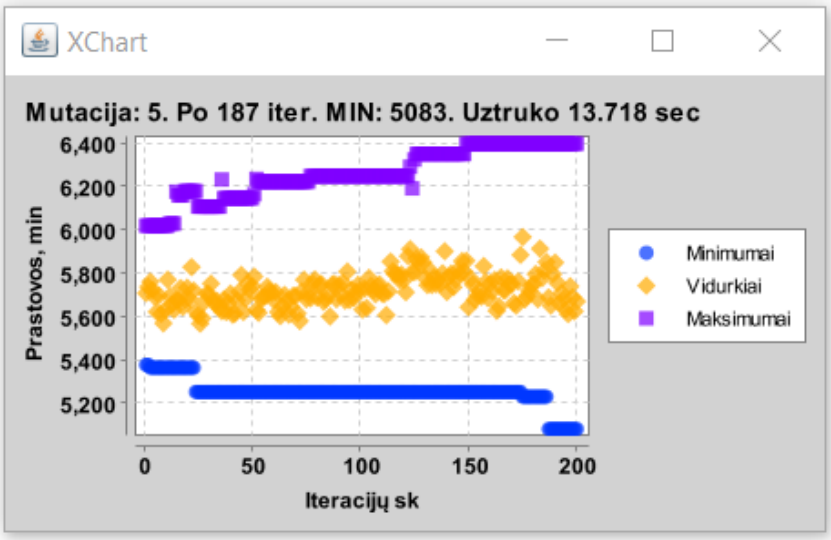
3.3.2.3 pav. Atliekama antroji mutacija



3.3.2.4 pav. Atliekama trečioji mutacija



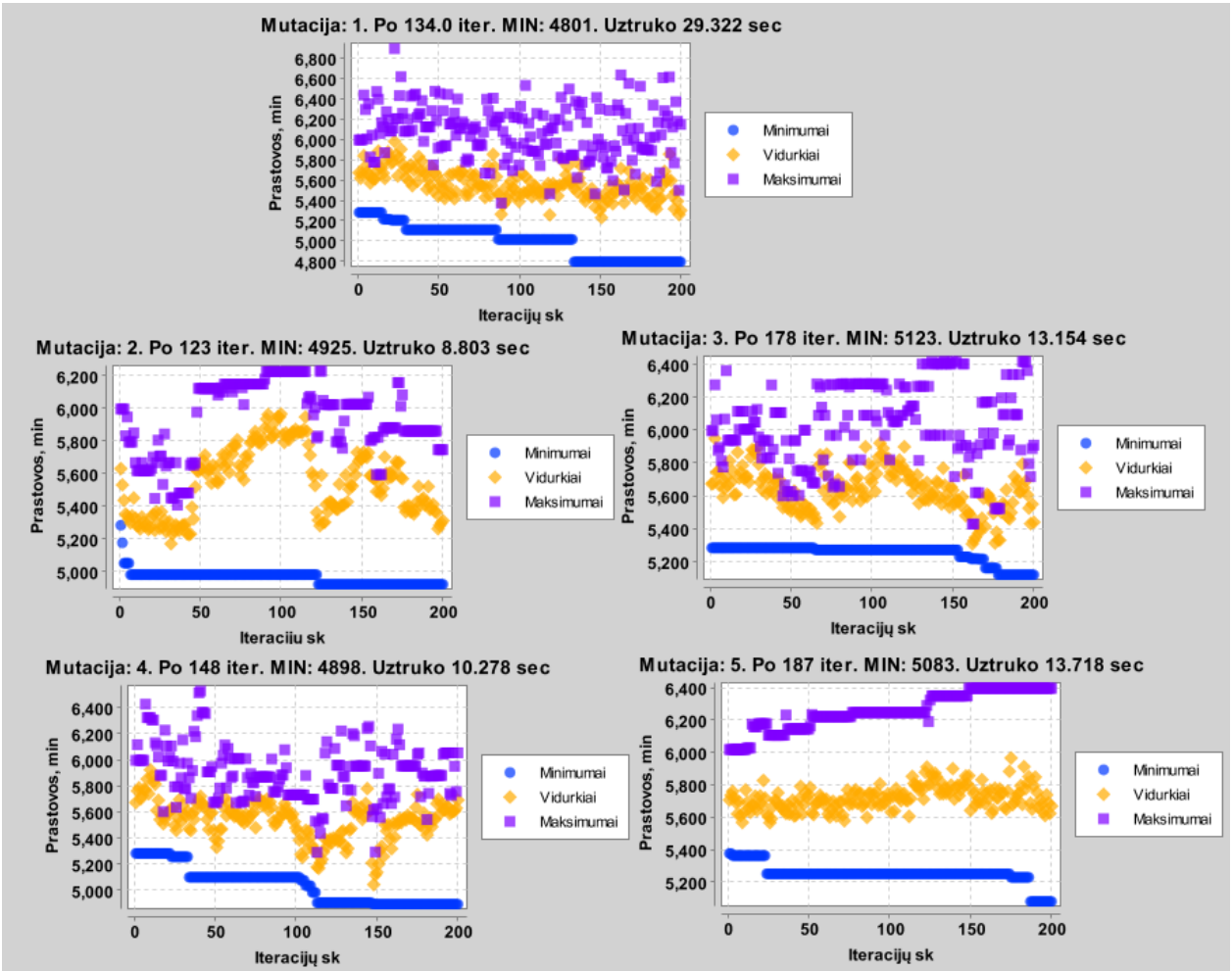
3.3.2.5 pav. Atliekama ketvirtoji mutacija



3.3.2.6 pav. Atliekama penktoji mutacija

3.3.3 Skaičiavimų rezultatai atlikus 200 iteracijų, tikslo funkcija – prastovų minimizavimas

Skaičiavimai buvo atlikti su 200 ir 500 iteracijų. Čia pateikiami 200 iteracijų skaičiavimo rezultatai. Pateikiami skaičiavimų su visomis mutacijų rūšimis grafikai (žr. 3.3.3.1 pav.).



3.3.3.1 pav. Skaičiavimų su visomis mutacijų rūšimis grafikai

Atliktus skaičiavimus, kurių dalis pavaizduoti grafikuose galima suvesti į lentelę:

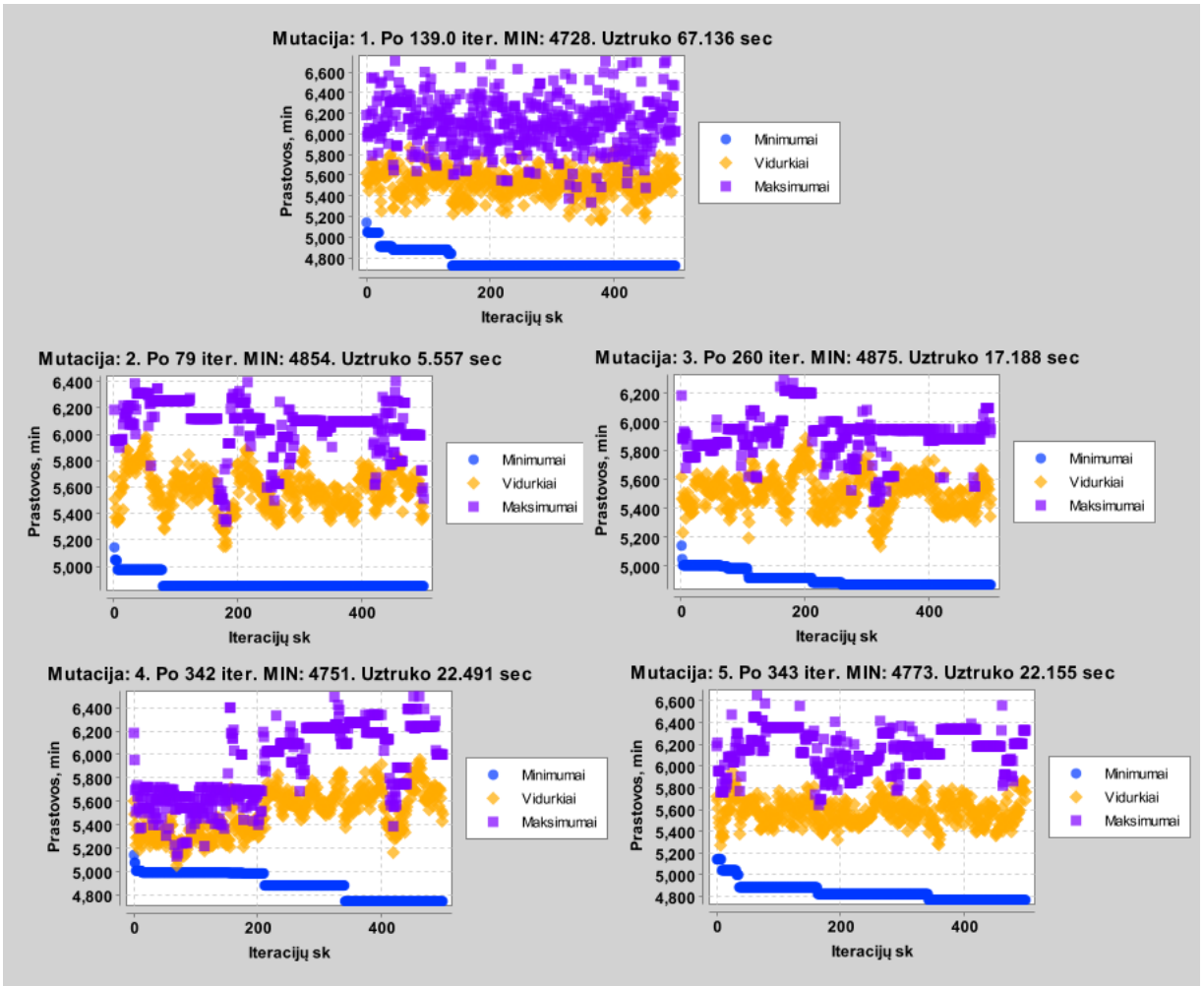
3.3.3.1 lentelė. 200 iteracijų skaičiavimo rezultatai

200 vidurkiai			200 minimumai		
Mutacija	Prastovos	Iteracijų kiekis	Mutacija	Prastovos	Iteracijų kiekis
1	**4779.4**	**82**	1	**4652**	**179**
2	4942.2	77.2	2	4813	162
3	5044.6	64.8	3	4967	2
4	4915.2	91.4	4	4787	168
5	4885.4	95.4	5	4825	31

Lentelėje 3.3.3.1 pateikti penkių skaičiavimų duomenys. Kairėje dalyje pateikti skaičiavimų vidurkiai, dešinėje – tų skaičiavimų geriausios reikšmės su kiekvienos rūšies mutacija. Lentelėje aiškiai matoma, kad tiek skaičiavimų vidurkiai, tiek optimalus sprendinys yra gaunamas su pirmąja mutacija.

3.3.4 Skaičiavimų rezultatai atlikus 500 iteracijų, tikslo funkcija – prastovų minimizavimas

Pateikiami 500 iteracijų skaičiavimo rezultatai su visomis mutacijų rūšimis grafikai (žr. 3.3.4.1 pav.).



3.3.4.1 pav. Skaičiavimų su visomis mutacijų rūšimis grafikai

Skaičiavimų rezultatai suvedami į lentelę:

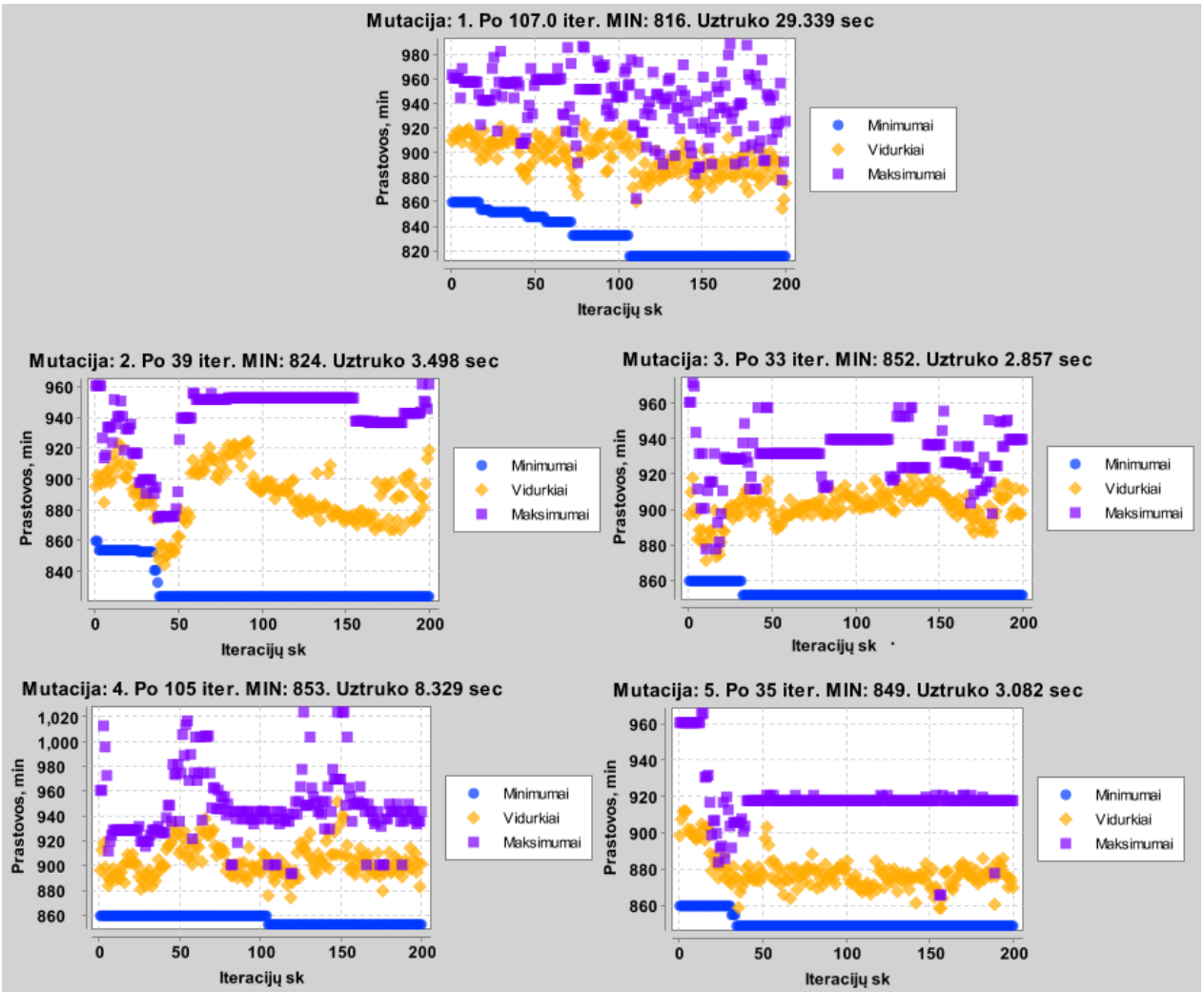
3.3.4.1 lentelė. 500 iteracijų skaičiavimo rezultatai

500 vidurkiai			500 minimumai		
Mutacija	Prastovos	Iteracijų kiekis	Mutacija	Prastovos	Iteracijų kiekis
1	4762.4	211.4	1	4711	8
2	4905.4	216.8	2	4830	232
3	4845	155.8	3	4744	84
4	4810.6	198.2	4	4634	498
5	**4751.6**	**198**	5	**4628**	**331**

Lentelėje 3.3.4.1 pateikti penkių skaičiavimų duomenys. Lentelėje pastebima, kad padidėjus atliekamų iteracijų kiekiui, geriausią sprendinį suranda penktoji mutacija. Pirmosios rušies mutacijos vidurkiai nedaug prastesni nei penktosios, tačiau penktoji mutacija surado ženkliai geresnį sprendinį, nei pirmoji, nors tai užtruko ir ilgiau – sprendinys rastas atlikus 331-ąją iteraciją. Prastovos 4628min yra mažiausias kada nors rastas tikslo funkcijos sprendinys.

3.3.5 Skaičiavimų rezultatai atlikus 200 iteracijų, tikslo funkcija – gamybos trukmės minimizavimas

200 iteracijų skaičiavimai su visomis mutacijų rūšimis (žr. 3.3.5.1 pav.).



3.3.5.1 pav. Skaičiavimų su visomis mutacijų rūšimis grafikai

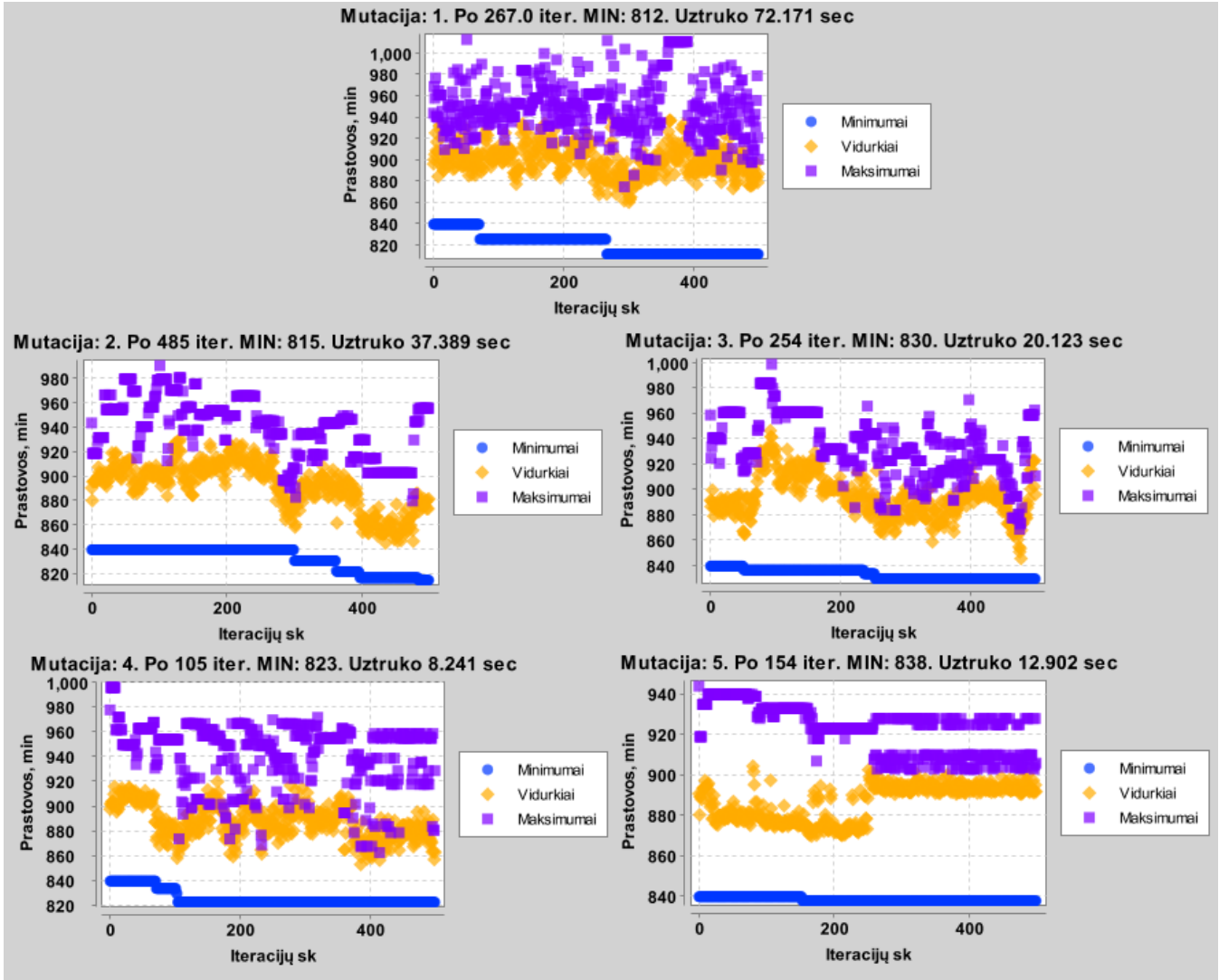
3.3.5.1 lentelė. 200 iteracijų skaičiavimo rezultatai

200 vidurkiai			200 minimumai		
Mutacija	Trukmė	Iteracijų kiekis	Mutacija	Trukmė	Iteracijų kiekis
1	**822.2**	**134.2**	1	**805**	**79**
2	834	105.8	2	832	169
3	849.6	30.6	3	836	55
4	839.8	86.4	4	825	77
5	847.2	50.8	5	832	76

Lentelėje 3.3.5.1 pateikti penkių skaičiavimų duomenys. Pastebime, pirmosios rūšies mutacijos skaičiavimų vidurkiai yra geriausi, o taip pat ir geriausias sprendinys buvo surastas su pirmosios rūšies mutacija.

3.3.6 Skaičiavimų rezultatai atlikus 500 iteracijų, tikslo funkcija – gamybos trukmės minimizavimas

500 iteracijų skaičiavimai su visomis mutacijų rūšimis (žr. 3.3.6.1 pav.).



3.3.6.1 pav. Skaičiavimų su visomis mutacijų rūšimis grafikai

3.3.6.1 lentelė. 500 iteracijų skaičiavimo rezultatai

500 vidurkiai			500 minimumai		
Mutacija	Trukmė	Iteracijų kiekis	Mutacija	Trukmė	Iteracijų kiekis
1	**817.2**	**310.6**	1	812	267
2	826.2	217.8	2	**802**	**32**
3	833.8	236.2	3	828	211
4	825.8	91.8	4	819	7
5	833.2	133.4	5	809	67

Lentelėje 3.3.6.1 pateikti penkių skaičiavimų duomenys. Lentelėje pastebima, kad pirmosios rūšies mutacija ir vėl gavo geriausius skaičiavimų vidurkius, tačiau šį kartą geriausias sprendinys buvo gautas su antrosios rūšies mutacija. Taip pat matome, kad šis sprendinys buvo gautas skaičiavimų pradžioje, atlikus likusias iteracijas, geresnis sprendinys nebuvo gautas. Ar tai yra geriausias galimas sprendinys – nežinome, nes nebuvo atliktas pilnasis perrinkimas – duomenų yra 20, taigi viso galimų variantų 2432902008176640000, trumpai tariant – daug. Tačiau iš visų atliktų skaičiavimų, 802min yra trumpiausia gamybos trukmė.

Išvados

Tikslo funkcija – prastovų minimizavimas:

- Atlikus iki 200 iteracijų, gaunamas sąlyginai geras rezultatas – prastovų trukmė – 4652 min.
- Tačiau atlikus iki 500 iteracijų, gaunamas dar geresnis rezultatas – prastovos – 4628 min.
- Pirmoji mutacija (sumaišomi visi chromosomos elementai) davė gerus rezultatus atlikus 200 iteracijų, tačiau ilgame bėgime – atlikus 500 iteracijų, geriausiai pasirodė penktoji mutacija (išmaišomi pirmi 9 elementai).

Tikslo funkcija – gamybos trukmės minimizavimas:

- Atlikus 200 iteracijų, pirmoji mutacijos rūšis gavo geriausius skaičiavimų vidurkius ir rado geriausią sprendinį, lyginant su kitomis mutacijomis. Gamybos trukmė – 805min.
- Atlikus 500 iteracijų, skaičiavimų vidurkių geriausias rezultatas priklauso taip pat pirmąjai mutacijai, tačiau optimalus sprendinys buvo rastas su antrosios rūšies mutacija. Gamybos trukmė – 802min.

Apibendrinant:

- Apžvelgus visus rezultatus matome, kad skaičiuojant su pirmosios rūšies mutacija, randamas geriausias sprendinys per priimtiną laiko tarpą.
- 200 iteracijų yra pakankamas iteracijų kiekis, kad surasti sąlyginai gerą sprendinį. Norint gauti geresnį sprendinį, reikia atlikti ženkliai daugiau iteracijų, o sprendinio kokybė gerėja labai nežymiai. Tyrime buvo norima nustatyti koks iteracijų kiekis yra pakankamas rasti sąlyginai gerą sprendinį, neatsižvelgiant į laiko sąnaudas.

Literatūros sąrašas

1. A. Žilinskas - „Matematinis programavimas“ 2005. Prieiga per internetą: http://www.mii.lt/zilinskas/uploads/MathematicalProgramming.pdf
2. Alfonsas Misevičius, Vytautas Bukšnaitis, Jonas Blonskis - „Kombinatorinio optimizavimo ir genetinių algoritmų aspektai. Prieiga per internetą: https://www.researchgate.net/publication/264893793_Kombinatorinio_optimizavimo_ir_genetiniu_algoritmu_aspektai
3. Combinatorial Optimization [interaktyvus]. Prieiga per internetą: http://www.mslevin.iitp.ru/CO.HTM
4. CRESCENZI, P.; KANN, V. A compendium of NP optimization problems. Nuoroda internete: http://www.csc.kth.se/~viggo/wwwcompendium/wwwcompendium.html
5. Edgaras Šakurovas - „Genetinio ir Tabu paieškos algoritmų naudijimo gamybinių tvarkaraščių sudarymui analizė“. Magistro darbas. KTU taikomosios matematikos katedra. Nuoroda internete: http://vddb.laba.lt/fedora/get/LT-eLABa-0001:E.02~2008~D_20080716_112811-81808/DS.005.0.01.ETD
6. Gediminas Kiraitis - „Tvarkaraščių sudarymo uždavinių ir jų algoritmų tyrimas“. KTU taikomosios matematikos katedra, 2010. Nuoroda internete: http://vddb.library.lt/fedora/get/LT-eLABa-0001:E.02~2010~D_20100825_101713-01675/DS.005.0.01.ETD
7. Gražvydas Felinskas - „Euristinių metodų tyrimas ir taikymas ribotų išteklių tvarkaraščiams optimizuoti“. Daktaro disertacija. VDU Matematikos ir informatikos institutas, 2007. Nuoroda internete: http://www.mii.lt/files/mii_dis_07_felinskas.pdf
8. Lina Rajeckaitė - „Gamybinių tvarkaraščių sudarymo uždavinių ir algoritmų analizė“. Magistro darbas. KTU taikomosios matematikos katedra, 2009. Nuoroda internete: http://vddb.library.lt/fedora/get/LT-eLABa-0001:E.02~2009~D_20090831_153239-72331/DS.005.0.01.ETD
9. Marek Obitko - „Introduction to Genetic Algorithms“, 1998. Nuoroda internete: http://obitko.com/tutorials/genetic-algorithms/
10. Narimantas Listopadskis - „Kombinatorinio optimizavimo uždaviniai ir jų sprendimo algoritmai“. Nuoroda internete: http://www.technologijos.lt/n/svietimas/kurstoti/kur_ir_ka_studijuoti/specialybes/straipsnis?name=S-18508
11. Naveen Garg, Sachin Jain and Chaitanya Swamy „A Randomized Algorithm for Flow Shop Scheduling“. Prieiga per internetą: http://www.math.uwaterloo.ca/~cswamy/papers/049.pdf
12. Wikipedia The Free Encyclopedia [interaktyvus]. Nuoroda internete: https://www.wikipedia.org/

1 priedas. Programos kodas

1) GANau.java

```
import java.awt.BorderLayout;
import java.awt.EventQueue;
import javax.swing.JButton;
import javax.swing.JComboBox;
import javax.swing.JFileChooser;
import javax.swing.JFrame;
import javax.swing.JOptionPane;
import javax.swing.JPanel;
import javax.swing.JScrollPane;
import javax.swing.JTextArea;
import javax.swing.ScrollPaneConstants;
import java.awt.Component;
import java.io.BufferedWriter;
import java.io.File;
import java.io.FileInputStream;
import java.io.FileWriter;
import java.io.IOException;
import java.io.PrintWriter;
import java.io.Writer;
import javax.swing.SwingConstants;
import org.apache.poi.hssf.usermodel.HSSFCell;
import org.apache.poi.hssf.usermodel.HSSFRow;
import org.apache.poi.hssf.usermodel.HSSFSheet;
import org.apache.poi.hssf.usermodel.HSSFWorkbook;
import java.awt.Dimension;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import javax.swing.JLabel;
import com.jgoodies.forms.factories.DefaultComponentFactory;
import java.util.Random;
import javax.swing.JSeparator;
import java.awt.Color;
import java.awt.TextField;
import java.awt.event.KeyAdapter;
import java.awt.event.KeyEvent;
import java.util.Arrays;


public class GANau extends JFrame {


            public static int[][][][] LaikaiMedis;
            public static int[][][][] LaikaiMetalas;
            public static int[][][][] LaikaiAbu;


            public static int[][][][] LaikaiTikrasMedis;
            public static int[][][][] LaikaiTikrasMetalas;
            public static int[][][][] LaikaiTikrasAbu;

            public static int[][][] IrasaiMedis;
            public static int[][][] IrasaiMetalas;
            public static int[][][] IrasaiAbu;

            public static int[][][] IrasaiTikrasMedis;
            public static int[][][] IrasaiTikrasMetalas;
            public static int[][][] IrasaiTikrasAbu;
```

```
public static int[] StaklesMedis;
public static int[] StaklesMetalas;
public static int[] StaklesAbu;


public static int kolkas=1;
public static int desinen=30;
public static int VK=260;       // Vaizdavimo baze is kaires
public static int VV=80;        // Vaizdavimo baze is virsaus
public static int vienasKartas=0;
public static int NuoK=0;
public static int NuoD=0;
public static int MSGG;
public static int[][] wee;
public static int Best;
public static int sand=0;
public static int iteracijos;
public static int MaxKryzmNr;
public static int Gaminsim=1;
public static int asd=0;
public static int n1;

public static int n2;
public static int n22;
public static int n23;



public static int n3;
public static double[] TT=new double[6];
public static int vnt;
public static double laikas;
public static String input;
private static JFrame frame;
public static String[][] S2dataT;
public static Double[][] S2dataN;
public static String[][] S1dataT;
public static Double[][] S1dataN;
public static double[][] BestCr;
public static double[] VID1;
public static double[] MIN1;
public static double[] MAX1;


public static int[] BestEil;
public static int[][] ch;
public static int[] da;
public static String[] OOO;// = new String [n3];

public static String[] OOMedis;// = new String [n2];
public static String[] OOMetalas;// = new String [n2];
public static String[] OOAbu;// = new String [n2];


public static int[][][] OOO2;

public static int[][][] OO2;
public static int[][][] OO2Metalas;
public static int[][][] OO2Abu;
```

```
            public static double[] Totalm;

            public static int MaxStaklMedis;
            public static int MaxStaklMetalas;
            public static int MaxStaklAbu;



            public static int[] VisosGamybosLaikas = new int[4];
            public static int[] GamybosPrastovos = new int[4];

            public static int TiksloFunkcija;
            // TiksloFunkcija = 1; - skaiciuoja pagal prastovas (minimuzuoja)
            // TiksloFunkcija = 2; - skaiciuoja pagal gamybos trukme
(minimuzuoja)
            public static int rowNum1;
            public static int colNum1;
            public static int rowNum;
            public static int colNum;
            public static int chromsk;
            public static int gerC;
            public static int bloC;
            public static int MutacRus;
            public static int MutacProc;
            public static int IteracNr;
            public static int KryzminimoTaskas;
            public static int mtc;
            public static int abc=0;
            public static int genvid=0;
            public static String indent = "                          ";


            public static String Gaminiai(String Sar[][],int b)
            {
                        String[][] Ba = new String[rowNum1][colNum1];
                        Ba[0][0] = "Visi gaminiai";
                        Ba[0][0] += indent.substring(0, indent.length() -
Ba[0][0].length());
                        Ba[0][0] = Ba[0][0].substring(0,28);
                        for (int i=1; i<rowNum1;i++)
                        {
                        for (int j=1; j<rowNum1;j++)
                                    if(!Sar[i][0].equals(Ba[j][0]) &&
Ba[j][0]==null)
                                    {Ba[j][0]=Sar[i][0];
                                    break;}
                          else
                                    if(Sar[i][0].equals(Ba[j][0]))
                                                break;     }
                        return Ba[b][0];
            }

            public static String Failas()
                {
                        String a;
                        JFileChooser chooser = new JFileChooser();
                int returnVal = chooser.showOpenDialog(frame);
                if(returnVal == JFileChooser.APPROVE_OPTION) {
                }
                        a=chooser.getSelectedFile().getAbsolutePath();
                        return a;
                }

            public static int Veliausias(int[] G)
```

```
            {
                        int max=0;
                        for(int j=1;j<rowNum;j++)
                        {
                        for(int i=1;i<n3;i++)
                        {
            if(Integer.valueOf(Gaminam(G,j,i,2))>max)
                                                {
            max=Integer.valueOf(Gaminam(G,j,i,2));
                                                }}}
                        return max;
            }

            public static int randInt(int min, int max)
            {
                Random rand = new Random();
                int randomNum = rand.nextInt((max - min) + 1) + min;
                return randomNum;
            }

            public static void MAISYMAS(int a[])
            {
                for (int i=1; i<a.length;i++)
                {       int randomPosition = i + (int) (Math.random() *
(a.length-i));
                            int temp = a[i];
                            a[i] = a[randomPosition];
                            a[randomPosition] = temp;
            }}

            public static int Format(int b)
            {
                        int k=180;
                        String[][] Op = new String[2][n3];
                                    for (int i=1; i<n3;i++)
                        {
                        Op[0][i]=OOO[i];
                        k=k+20;
                        Op[1][i]=String.valueOf(k);
                        }
                        int d=0;
                        for (int i=1;i<n3;i++)
                        {
                                    if(S2dataT[b][1].equals(Op[0][i]))
                        {
                        d=Integer.valueOf(Op[1][i]);
                        break;
                        }}
                        return k;
            }

            public static int EachOp(int b, String Operacijos)
            {
                        int k=300;
                        String[][] Op = new String[2][n3];
                        for (int i=1; i<n3;i++)
                        {
                                    Op[0][i]=OOO[i];
                                    k=k+20;
                                    Op[1][i]=String.valueOf(k);
                        }
                        int d=0;
                        for (int i=1;i<n3;i++)
                                    {
```

```
                    if(Operacijos.equals(Op[0][i]))
                              {
                              d=Integer.valueOf(Op[1][i]);
                              break;
                    }         }
          return d;
}

public static int Min(int[] Prad)
{
          int min=1000000;
          int i=0;
          while(Prad[i]!=0){
                    if(Prad[i]<min)
                              min=Prad[i];
                    i=i+1;
                    }
          return min;
}

public static int MinDouble(double[] Prad)
{
          double min=1000000;
          int i=0;
          while(Prad[i]!=0){
                    if(Prad[i]<min)
                              min=Prad[i];
                    i=i+1;
                    }
          int va = (int) min;
          return va;
}

public static int MinimumasPagalKiek(int[] Prad, int n)
{
          int min=1000000;

          for(int i=0;i<n;i++){
                    if(Prad[i]<min)
                              min=Prad[i];
          }

          return min;
}

public static int MAX(int[] Prad, int n)
{

          int max=0;
          for(int i=0;i<n;i++){
                    if(Prad[i]>max)
                              max=Prad[i];

                    }
          return max;
}

public static double MAXDouble(double[] Prad, int n)
{

          double max=0;
          for(int i=0;i<n;i++){
                    if(Prad[i]>max)
                              max=Prad[i];
```

```
                        }
                return max;
        }

        public static int MMMax(int[] Prad)
        {
                int max=-1;
                int i=0;
                while(Prad[i]!=0){
                        if(Prad[i]>max)
                                max=Prad[i];
                        i=i+1;
                        }
                return max;
        }

        public static int MinChrom(int[] Prad)
        {
                int min=1000000;
                for(int i=0;i<chromsk;i++){
                        if(Prad[i]<min)
                                min=Prad[i];          }
                return min;
        }

        public static int MinIkiInd(int[] Prad, int Mano, int operatorius)
        {
                int da=0;
                int min=1000000;
                for(int i=0;i<operatorius;i++){
                        if(Prad[i]<min && Prad[i]>Mano){
                                min=Prad[i];
                                da=i;
                        }          }
                return da;
        }

        public static int MinIkiIndDouble(double[] Prad, int Mano, int
operatorius)
        {
                int da=0;
                double min=1000000;
                for(int i=0;i<operatorius;i++){
                        if(Prad[i]<min && Prad[i]>Mano){
                                min=Prad[i];
                                da=i;
                        }          }
                return da;
        }

        public static int MaxIkiInd(int[] Prad, int Mano, int operatorius)
        {
                int da=0;
                int max=0;
                for(int i=0;i<operatorius;i++) {
        if(Prad[i]>max && Prad[i]<Mano) {
                                max=Prad[i];
                                da=i;
                        }          }
                return da;
        }

        public static double VidurkisDouble(double[] Prad,int operatorius)
```

```
	{
		double vid=0;
		for(int i=0;i<operatorius;i++)
		{
			vid=vid+Prad[i];
		}
		vid=vid/operatorius;
		return vid;
	}


	public static String Gaminam(int[] nn0, int nn1, int nn2, int nn3)
	{

		int VisosGamybosTrukme = 0;

		for(int a1=0;a1<n2;a1++)
		for(int i=0;i<StaklesMedis[a1];i++)
		for(int j=0;j<3;j++)

	for(int a2=0;a2<rowNum;a2++){

	LaikaiMedis[a1][i][j][a2]=0;

	LaikaiTikrasMedis[a1][i][j][a2]=0;
                                                }

		for(int a1=0;a1<n22;a1++)
			// i - kiekvienos rusies stakliu kiekis
			for(int i=0;i<StaklesMetalas[a1];i++)

	for(int a2=0;a2<rowNum;a2++){
	LaikaiMetalas[a1][i][0][a2]=0;
	LaikaiMetalas[a1][i][1][a2]=0;
	LaikaiMetalas[a1][i][2][a2]=0;
	LaikaiTikrasMetalas[a1][i][0][a2]=0;
	LaikaiTikrasMetalas[a1][i][1][a2]=0;
	LaikaiTikrasMetalas[a1][i][2][a2]=0;

                                                }

		for(int a1=0;a1<n23;a1++)
			// i - kiekvienos rusies stakliu kiekis
			for(int i=0;i<StaklesAbu[a1];i++)
				// pradzia, pabaiga, trukme
				for(int j=0;j<3;j++)
					// gaminiu kiekis
				for(int a2=0;a2<rowNum;a2++){

	LaikaiAbu[a1][i][j][a2]=0;

	LaikaiTikrasAbu[a1][i][j][a2]=0; // Man rodos, kad sitas masyvas
nera reikalingas
                                                }

		for (int q1=0;q1<n2;q1++)
			for (int q2=0;q2<StaklesMedis[q1];q2++)
			{
				IrasaiMedis[q1][q2]=0;
				IrasaiTikrasMedis[q1][q2]=0;
			}

		for (int q1=0;q1<n22;q1++)
			for (int q2=0;q2<StaklesMetalas[q1];q2++)
```

```
                    {
                              IrasaiMetalas[q1][q2]=0;
                              IrasaiTikrasMetalas[q1][q2]=0;
                    }

          for (int q1=0;q1<n23;q1++)
                    for (int q2=0;q2<StaklesAbu[q1];q2++)
                    {
                              IrasaiAbu[q1][q2]=0;
                              IrasaiTikrasAbu[q1][q2]=0;
                    }


          for(int i3=0;i3<3;i3++)
                    for(int i2=1;i2<rowNum;i2++)
                              for(int i1=1;i1<n3;i1++)

                                        OO2[i1][i2][i3]=0;

          int kl=0;
          int mk=0;

          int Prastovos=0;

          int[][] PrastEaMedis = new int[n2][MaxStaklMedis];
          for(int i1=1;i1<n2;i1++)
                    for(int i2=1;i2<StaklesMedis[i1];i2++)
                              PrastEaMedis[i1][i2]=0;

          int[][] PrastEaMetalas = new int[n22][MaxStaklMetalas];
          for(int i1=1;i1<n22;i1++)
                    for(int i2=1;i2<StaklesMetalas[i1];i2++)
                              PrastEaMetalas[i1][i2]=0;

          int[][] PrastEaAbu = new int[n23][MaxStaklAbu];
          for(int i1=1;i1<n23;i1++)
                    for(int i2=1;i2<StaklesAbu[i1];i2++)
                              PrastEaAbu[i1][i2]=0;

          String[][][] Pab = new String[rowNum][n3][4];
          Pab[0][0][0]=String.valueOf(0);

          String Trukme[][] = new String[3][rowNum1+1];
          int[] TrukmeN = new int[rowNum1+12];

          int aaa =rowNum;
          int[][][] DarbinisN = new int[rowNum+1][3][aaa+n2];
          int[][][] DarbinisMedisN = new int[rowNum+1][3][aaa+n2];
          int[][][] DarbinisMetalasN = new
          int[rowNum+1][3][aaa+n2];
          int[][][] DarbinisTikrasN = new
          int[rowNum+1][3][aaa+n2];

          String[][] DarbinisT = new String[rowNum+1][aaa+n2];
          String[][] DarbinisMedisT = new
String[rowNum+1][aaa+n2];
          String[][] DarbinisMetalasT = new
String[rowNum+1][aaa+n2];
          String[][] DarbinisTikrasT = new
String[rowNum+1][aaa+n2];
```

```
                                    for(int i2=0;i2<rowNum+1;i2++)
                                              for(int i1=0;i1<aaa+n2;i1++)
                                              {

                                                        DarbinisN[i2][0][i2]=0;
                                                        DarbinisN[i2][1][i2]=0;
                                                        DarbinisN[i2][2][i2]=0;


          DarbinisMedisN[i2][0][i2]=0;
          DarbinisMedisN[i2][1][i2]=0;
          DarbinisMedisN[i2][2][i2]=0;
          DarbinisMetalasN[i2][0][i2]=0;
          DarbinisMetalasN[i2][1][i2]=0;
          DarbinisMetalasN[i2][2][i2]=0;
          DarbinisTikrasN[i2][0][i2]=0;
          DarbinisTikrasN[i2][1][i2]=0;
          DarbinisTikrasN[i2][2][i2]=0;

                                              }

                              int[][] k = new int[2][rowNum];
                              int[] rew=new int[rowNum];



                              for ( int j=1; j< rowNum; j++) {

                                        Trukme[0][j]=S1dataT[j][1];

                              }


                              for ( int i=1; i< rowNum1; i++){
                                        double a=S2dataN[i][2];
                                        int b = (int)a;
                                        TrukmeN[i]=b;

                              }

                              // Einu per sheet1 ir ziuriu kokie gaminiai yra sarase
                              for ( int r=1; r<rowNum; r++)
                              {
                                        //System.out.println(" GAMINIO KODAS:
"+S1dataT[nn0[r]][1]);
                                        kl=0;
                                        // Einu per sheet2 ir ziuriu kokiu gaminiu
irasus su laikais turiu
                                        for ( int i=1; i< rowNum1; i++)
                                        {
                                                  // SUSKAICIUOJA KIEK KIEKVIENAM
GAMINIUI ATLIEKAMA OPERACIJU

                                                  /*
                                                  System.out.println("r "+r);
                                                  System.out.println("nn0[r]
"+nn0[r]);

          System.out.println("S1dataT[nn0[r]][1] "+S1dataT[nn0[r]][1]);
                                                  System.out.println("S2dataT[i][0]
"+S2dataT[i][0]);

                                                  */
```

```
                                if(S1dataT[nn0[r]][1].equals(S2dataT[i][0]))
                                {



                                          kl=kl+1;   mk=i;
                                //System.out.println("kl = "+kl);  //kelinta
operacija daroma
                                //System.out.println("mk = "+mk);  //nuo
kurios eilutes duomenu faule reikia pradeti skaiciuoti

                                // k[0][j]=  sitas k[j] - grazina kiek
operaciju yra kiekiename gaminyje

                                k[0][nn0[r]]=kl;
                                //System.out.println("k[0][nn0[r]]=kl   =
"+k[0][nn0[r]]);  //kelinta operacija daroma
                                k[1][nn0[r]]=mk-kl+1;    // k[1][j]=
skaiciuoja nuo kurio iraso is sheet2 pradeti
                                //System.out.println("k[1][nn0[r]]=mk-kl+1
= "+k[1][nn0[r]]+"   nuo kurios eilutes duomenu faile reikia pradeti
skaiciuoti");

                                }
                                }



                     rew[r]=1;
                     DarbinisN[1][1][rew[r]]=0;

                     for ( int o=k[1][nn0[r]]; o< k[1][nn0[r]]+k[0][nn0[r]];
o++)
                     {

          DarbinisT[r][rew[r]]=S2dataT[o][1];


          DarbinisMedisT[r][rew[r]]=S2dataT[o][1];

          DarbinisMetalasT[r][rew[r]]=S2dataT[o][1];

                                          //System.out.print("Operacija
"+DarbinisT[r][rew[r]]);


          DarbinisN[r][0][rew[r]]=TrukmeN[o];

          DarbinisMedisN[r][0][rew[r]]=TrukmeN[o];

          DarbinisMetalasN[r][0][rew[r]]=TrukmeN[o];


                                                          //System.out.print("
Trukme "+DarbinisN[r][0][rew[r]]);

                                                          //System.out.println("
Operacijos numeris = "+ rew[r]);
                                                          rew[r]=rew[r]+1;
                                                          // rew[r] - skaiciuoja
kiek yra kiekvieno gaminio atliekamu operaciju sarase
                                                          // r - gaminio numeris
is sheet1
```

```
                        }

                        DarbinisN[1][1][rew[r]]=0;
                        }



                        for ( int r=1; r<rowNum; r++)
                        {
                                    int FinishMedis=0;
                                    int FinishMetalas=0;
                                    //System.out.println(" GAMINIO KODAS:
"+S1dataT[nn0[r]][1]);

                                    int ArPirmaMetOp = 0;

                        for(int f1=1;f1<1+k[0][nn0[r]];f1++)
                        {


                                    for (int a1=1;a1<n2;a1++)
                                                {

                                          // cia iskarto suku dar viena
cikla ir einu per stakles
                                                                        if
(DarbinisT[r][f1].toLowerCase().contains(OOMedis[a1].toLowerCase()))
                                                                        {

            //System.out.println(OOMedis[a1]);

            int[] Kolkas= new int[StaklesMedis[a1]];
                                                                                    //
nunulinu reiksmes

            for (int a4=0;a4<StaklesMedis[a1];a4++)

            Kolkas[a4]=0;


            //if (r==rowNum-1)
                                                                                    //
            System.out.println(" PASKUTINIS IRASAS");


            for (int a4=0;a4<StaklesMedis[a1];a4++)

            {

            //System.out.println(OO[a1]+"  "+StaklesMedis[a4]);


            int Variantas = 0;



            if(IrasaiMedis[a1][a4]==0 && f1==1)

                        Variantas=1;

            else
```

```
            if(IrasaiMedis[a1][a4]==1 && f1==1)
                        Variantas=2;
            else
            if(IrasaiMedis[a1][a4]>1 && f1==1)
                        Variantas=3;
            else
            if(IrasaiMedis[a1][a4]==0 && f1>1)
                        Variantas=4;
            else
            if(IrasaiMedis[a1][a4]==1 && f1>1)
                        Variantas=5;
            else
            if(IrasaiMedis[a1][a4]>1 && f1>1)
                        Variantas=6;
            else
            if(Variantas==0)
                        System.out.println("Kazkas dar naujo - Variantas = " +
Variantas);




            switch (Variantas) {


            case 1: // (Irasai[a1][a4]==0 && f1==1) ++++++++++++++++++++++
                                    //System.out.println("Papuoleme ir case : " +
Variantas);
                                    DarbinisN[r][1][f1]=0;
                                    break;
            case 2: // (Irasai[a1][a4]==1 && f1==1) ++++++++++++++++++++++
                                    //System.out.println("Papuoleme ir case : " +
Variantas);
                                    // Ziuriu ar telpu iki iraso
```

```
            // Laikai[a1][a4][0][0] - paskutinis [0], nes yra tik vienintelis
irasas


            if(DarbinisN[r][0][f1]<LaikaiMedis[a1][a4][0][0])

                                                {          DarbinisN[r][1][f1]=0;
            }

                                    else // jeigu netelpu, pradedu gaminti po
iraso

                                                {
            DarbinisN[r][1][f1]=LaikaiMedis[a1][a4][1][0];        }

                                    break;

            case 3: // (Irasai[a1][a4]>1 && f1==1) ++++++++++++++++++++++

                                    //System.out.println("Papuoleme ir case : " +
Variantas);

                                    // Ziuriu ar mano pirmoji operacija telpa i
kuri nors tarpa tarp irasu

                                    for(int a2=0;a2<IrasaiMedis[a1][a4];a2++)

                                    {

                                    if (a2==0)

                                                {

            if(DarbinisN[r][0][f1]<LaikaiMedis[a1][a4][0][a2])


            {DarbinisN[r][1][f1]=0;
                        break;
                                    }
                        else {

            for(int a3=a2+1;a3<IrasaiMedis[a1][a4];a3++) // sitas ciklas sukasi
tol, kol tenkinama viena is tu dvieju operaciju - BUTINAI!
                                                                                //
--------                                                    // ieskau ar telpa i
bet kuri tarpa


            if(DarbinisN[r][0][f1]<LaikaiMedis[a1][a4][0][a3]-
LaikaiMedis[a1][a4][1][a3-1])

                                                                                {
            DarbinisN[r][1][f1]=LaikaiMedis[a1][a4][1][a3-1];     break;     }


            else // jeigu netelpa i tarpa ziuriu ar paskutine operacija


            if(a3+1==IrasaiMedis[a1][a4])

                                                                                {
            DarbinisN[r][1][f1]=LaikaiMedis[a1][a4][1][a3];       break;     }
                                                            }
```

```
                                                                    }
                                                                    else


          if(DarbinisN[r][0][f1]<LaikaiMedis[a1][a4][0][a2]-
LaikaiMedis[a1][a4][1][a2-1])

                                                {
          DarbinisN[r][1][f1]=LaikaiMedis[a1][a4][1][a2-1];     break;     }

                                                else // teoriskai cia galetu
kažkur nubegti sprendinys, taciau nemanau, nes lygtais nera salygos jam tai
padaryti


          if(a2+1==IrasaiMedis[a1][a4])

                                                {
          DarbinisN[r][1][f1]=LaikaiMedis[a1][a4][1][a2];       break;     }

          }

                                    break;

          case 4: // (Irasai[a1][a4]==0 && f1>1) ++++++++++++++++++++++

                                    //System.out.println("Papuoleme ir case : " +
Variantas);

                                    DarbinisN[r][1][f1]=DarbinisN[r][2][f1-1];

                                    break;

          case 5: // (Irasai[a1][a4]==1 && f1>1) ++++++++++++++++++++++

                                    //System.out.println("Papuoleme ir case : " +
Variantas);

                                    // jei pataikau IKI vienintelio iraso

                                    if (DarbinisN[r][2][f1-
1]<LaikaiMedis[a1][a4][0][0])

                                    {

                                              // jeigu   tilpau iki vienintelio
iraso

                                              if (DarbinisN[r][2][f1-
1]+DarbinisN[r][0][f1]<LaikaiMedis[a1][a4][0][0])

                                              {
          DarbinisN[r][1][f1]=DarbinisN[r][2][f1-1]; }

                                              // jeigu netilpau iki vienintelio
iraso

                                              else

                                              {
          DarbinisN[r][1][f1]=LaikaiMedis[a1][a4][1][0];        }

                                    }
```

```
                                // jei pataikau ANT vienintelio iraso

                                else if (DarbinisN[r][2][f1-
1]>=LaikaiMedis[a1][a4][0][0] && DarbinisN[r][2][f1-
1]<=LaikaiMedis[a1][a4][1][0])

                                                {
          DarbinisN[r][1][f1]=LaikaiMedis[a1][a4][1][0];        }

                                // jei pataikau PO  vienintelio iraso

                                else if (DarbinisN[r][2][f1-
1]>LaikaiMedis[a1][a4][1][0])

                                                {
          DarbinisN[r][1][f1]=DarbinisN[r][2][f1-1]; }

                                break;



          case 6: // (Irasai[a1][a4]>1 && f1>1)
                                                int ArTuriu = 0;



                                for(int a2=0;a2<IrasaiMedis[a1][a4];a2++)

          if (ArTuriu==1)

                                                                break;

                                                else

                                                if(a2==0)

                                                {
//
                                                                if(a1==8)
//
                                                                        //for (int
a2=0;a2<Irasai[a1];a2++)
//

          System.out.println("laikai   "+OO[a1]+"  Pradzia =
"+Laikai[a1][a4][0][a2]+"  Pabaiga = "+Laikai[a1][a4][1][a2]+"  Trukme =
"+Laikai[a1][a4][2][a2]);
//


                                                                // Jeigu baigiu iki
pirmojo iraso

                                                                if(DarbinisN[r][2][f1-
1]<LaikaiMedis[a1][a4][0][a2])                                          // [a2]

                                                                {// Jeigu telpu

                                                                        if(
DarbinisN[r][2][f1-1]+DarbinisN[r][0][f1]<LaikaiMedis[a1][a4][0][a2])

                                                                        {
```

```
            DarbinisN[r][1][f1]=DarbinisN[r][2][f1-1];

            //System.out.println(DarbinisN[r][1][f1]);

            ArTuriu = 1;

            break;
            }
                                                                    else {
                                                                    // Jeigu
netelpu
            for(int a3=a2+1;a3<IrasaiMedis[a1][a4];a3++) // sitas ciklas sukasi
tol, kol tenkinama viena is tu dvieju operaciju - BUTINAI!

                        // ieskau ar telpa i bet kuri tarpa

                        if(DarbinisN[r][0][f1]<LaikaiMedis[a1][a4][0][a3]-
LaikaiMedis[a1][a4][1][a3-1])

                        {
            DarbinisN[r][1][f1]=LaikaiMedis[a1][a4][1][a3-1];     ArTuriu = 1;
            break;     }

                        else // jeigu netelpa i tarpa ziuriu ar paskutine
operacija

                                    if(a3+1==IrasaiMedis[a1][a4])

                        {
            DarbinisN[r][1][f1]=LaikaiMedis[a1][a4][1][a3];       ArTuriu = 1;
            break;     }

                                                                                }
                                                            }
                                                            // Jeigu baigiu ant
pirmo iraso
                                                            else if
(DarbinisN[r][2][f1-1]>=LaikaiMedis[a1][a4][0][a2] && DarbinisN[r][2][f1-
1]<=LaikaiMedis[a1][a4][1][a2])
                                                            {
                                                                    if
(a2+1==IrasaiMedis[a1][a4])
                                                                    {
```

```
          DarbinisN[r][1][f1]=LaikaiMedis[a1][a4][1][a2];       ArTuriu = 1;
          break;     }

                                                                else

                                                                { // jeigu ne
paskutinis irasas ant stakliu, bet zinau, kad mano praeita operacija jau yra
pasibaigus


          for(int a3=a2+1;a3<IrasaiMedis[a1][a4];a3++) // sitas ciklas sukasi
tol, kol tenkinama viena is tu dvieju operaciju - BUTINAI!


          // ieskau ar telpa i bet kuri tarpa


          if(DarbinisN[r][0][f1]<LaikaiMedis[a1][a4][0][a3]-
LaikaiMedis[a1][a4][1][a3-1])


          {          DarbinisN[r][1][f1]=LaikaiMedis[a1][a4][1][a3-1];
          ArTuriu = 1;          break;     }


          else // jeigu netelpa i tarpa ziuriu ar paskutine operacija


                     if(a3+1==IrasaiMedis[a1][a4])


          {          DarbinisN[r][1][f1]=LaikaiMedis[a1][a4][1][a3];
          ArTuriu = 1;          break;     }

                                                                }

                                                          }\
                                                // Jeigu baigiu po pirmo iraso

                                                          else if
(DarbinisN[r][2][f1-1]>LaikaiMedis[a1][a4][1][a2])

                                                                {          if
(a2+1==IrasaiMedis[a1][a4])


          {          DarbinisN[r][1][f1]=DarbinisN[r][1][f1-1]; break;     }


          else


          { // jeigu ne paskutinis irasas ant stakliu, bet zinau, kad mano
praeita operacija jau yra pasibaigus


                     for(int a3=a2+1;a3<IrasaiMedis[a1][a4];a3++) // sitas
ciklas sukasi tol, kol tenkinama viena is tu dvieju operaciju - BUTINAI!


                                   // Irasas turi buti pasibaiges veliau, nei
mano operacija
```

```
                                    // Jei mano pabaiga mazesne uz iraso pabaiga
- baigiau ant iraso


           if(DarbinisN[r][2][f1]<LaikaiMedis[a1][a4][1][a3] &&
DarbinisN[r][2][f1]>=LaikaiMedis[a1][a4][0][a3])
                                    {
                                    if (a3+1==IrasaiMedis[a1][a4])

                                              {
           DarbinisN[r][1][f1]=LaikaiMedis[a1][a4][1][a3];       break;     }

           if(DarbinisN[r][0][f1]<LaikaiMedis[a1][a4][0][a3+1]-
LaikaiMedis[a1][a4][1][a3])
                                              {
           DarbinisN[r][1][f1]=LaikaiMedis[a1][a4][1][a3];  break;          }

                        if(DarbinisN[r][0][f1]<LaikaiMedis[a1][a4][0][a3+1]-
Math.max(DarbinisN[r][2][f1-1],LaikaiMedis[a1][a4][1][a3]))
                                              {
           DarbinisN[r][1][f1]=Math.max(DarbinisN[r][2][f1-
1],LaikaiMedis[a1][a4][1][a3]); break;     }

                                                                        }

                                    else
                        {

                                              if(a3+1==IrasaiMedis[a1][a4])


                                              {
           DarbinisN[r][1][f1]=LaikaiMedis[a1][a4][1][a3];       break;     }

                                    }}}}

                                              else

                                              { // irasas ne pirmas

                                                        // Pataikau i tarpa

                                                        if (DarbinisN[r][2][f1-
1]<LaikaiMedis[a1][a4][0][a2] && DarbinisN[r][2][f1-
1]>LaikaiMedis[a1][a4][1][a2-1])

                                                        {

                                                                  // ir jeigu
telpu  i tarpa


           if(DarbinisN[r][2][f1-
1]+DarbinisN[r][0][f1]<LaikaiMedis[a1][a4][0][a2])

                                                                  {
           DarbinisN[r][1][f1]=DarbinisN[r][2][f1-1]; break;     }

                                                                  else { //
jeigu i tarpa netilpau

                                                                                      //
jei tai buvo paskutinis irasas ant takliu
```

```
                                                                    if
(a2+1==IrasaiMedis[a1][a4])

                                                                    {
          DarbinisN[r][1][f1]=LaikaiMedis[a1][a4][1][a2];       break;     }


          else { // jeigu ne paskutinis irasas ant stakliu, bet zinau, kad
mano praeita operacija jau yra pasibaigus


          for(int a3=a2+1;a3<IrasaiMedis[a1][a4];a3++) // sitas ciklas sukasi
tol, kol tenkinama viena is tu dvieju operaciju - BUTINAI!


                    // ieskau ar telpa i bet kuri tarpa


                    if(DarbinisN[r][0][f1]<LaikaiMedis[a1][a4][0][a3]-
LaikaiMedis[a1][a4][1][a3-1])


                    {
          DarbinisN[r][1][f1]=LaikaiMedis[a1][a4][1][a3-1];     break;     }


                    else // jeigu netelpa i tarpa ziuriu ar paskutine
operacija


                              if(a3+1==IrasaiMedis[a1][a4])


                    {
          DarbinisN[r][1][f1]=LaikaiMedis[a1][a4][1][a3];       break;     }

          }}}else

                                                  {

                                                                    // jeigu
pataikau ant iraso

                                                                    if
(DarbinisN[r][2][f1-1]>=LaikaiMedis[a1][a4][0][a2] && DarbinisN[r][2][f1-
1]<LaikaiMedis[a1][a4][1][a2])
 {
                                                                    if
(a2+1==IrasaiMedis[a1][a4])

                                                                    {
          DarbinisN[r][1][f1]=LaikaiMedis[a1][a4][1][a2];       break;     }


          else

                                                                    {
// jeigu ne paskutinis irasas ant stakliu, bet zinau, kad mano praeita
operacija jau yra pasibaigus


          for(int a3=a2+1;a3<IrasaiMedis[a1][a4];a3++) // sitas ciklas sukasi
tol, kol tenkinama viena is tu dvieju operaciju - BUTINAI!
```

```
                    // ieskau ar telpa i bet kuri tarpa

                    if(DarbinisN[r][0][f1]<LaikaiMedis[a1][a4][0][a3]-
LaikaiMedis[a1][a4][1][a3-1])

                    {
          DarbinisN[r][1][f1]=LaikaiMedis[a1][a4][1][a3-1];     break;     }

                    else // jeigu netelpa i tarpa ziuriu ar paskutine
operacija

                              if(a3+1==IrasaiMedis[a1][a4])

                    {
          DarbinisN[r][1][f1]=LaikaiMedis[a1][a4][1][a3];       break;     }

          }}
                                                                  else

                                                                  {
          if (a2+1==IrasaiMedis[a1][a4])

                                                                  {
          DarbinisN[r][1][f1]=Math.max(DarbinisN[r][2][f1-
1],LaikaiMedis[a1][a4][1][a2]); break;     }

          //else // prasuka tusciai, nes irasas nera paskutinis, bet mano
operacija baigiasi veliau uz irasa

                                                                  //
          System.out.println("   DAR VIENAS KAZKOKS ATVEJIS, KURIO
NEAPGALVOJAU   ");                        }}}
                              break;

                    }

          Kolkas[a4]=DarbinisN[r][1][f1];

          // baigiasi a4 ciklas
                                                                  }




          int StakliuNr=0;

          StakliuNr = MinIkiInd(Kolkas,-1,StaklesMedis[a1]);



          DarbinisN[r][1][f1]=Kolkas[StakliuNr];


          DarbinisN[r][2][f1]=DarbinisN[r][1][f1]+DarbinisN[r][0][f1];
```

```
//
            DarbinisMedisN=DarbinisN;


            // IRASU MASYVAS TURI ATRODYTI: IrasaiMedis[a1][st] - a1  -
operacijos rusis, st - stakliu numeris

            // SITUS PASKUI REIKIA ATKOMENTUOTI IR SUTVARKYTI







            for (int a=1;a<n3;a++)

                        if
(OOO[a].toLowerCase().contains(DarbinisT[r][f1].toLowerCase()))

                        {
                        DarbinisT[r][f1]=OOO[a+StakliuNr];

            DarbinisMedisT[r][f1]=OOO[a+StakliuNr];

            //DarbinisMedisT[r][f1]=DarbinisT[r][f1]; NEBANDYK SITO NAUDOT


            //DarbinisN[r][2][f1]=DarbinisN[r][1][f1]+DarbinisN[r][0][f1];

            DarbinisMedisN[r][2][f1]=DarbinisN[r][2][f1];
            DarbinisMedisN[r][1][f1]=DarbinisN[r][1][f1];
            DarbinisMedisN[r][0][f1]=DarbinisN[r][0][f1];


                                    break;

                        }



            LaikaiMedis[a1][StakliuNr][0][IrasaiMedis[a1][StakliuNr]]=DarbinisN
[r][1][f1];

            LaikaiMedis[a1][StakliuNr][1][IrasaiMedis[a1][StakliuNr]]=DarbinisN
[r][2][f1];

            LaikaiMedis[a1][StakliuNr][2][IrasaiMedis[a1][StakliuNr]]=DarbinisN
[r][0][f1];


                                    IrasaiMedis[a1][StakliuNr] =
IrasaiMedis[a1][StakliuNr] + 1;


            if (FinishMedis < DarbinisN[r][2][f1])

                        {FinishMedis=DarbinisN[r][2][f1];

                        //System.out.println("Paskutinis Medzio irasas : "+
a21);

                        }
```

```
            break;
                                                                    }

                                                                }



                                    for (int a1=1;a1<n22;a1++)
                                    {

                            // cia iskarto suku dar viena cikla ir einu per stakles

                                                    if
(DarbinisT[r][f1].toLowerCase().contains(OOMetalas[a1].toLowerCase()))
                                                    {
                                                    //          if (a1==6)
                                                    //
            System.out.println(OOMetalas[a1]);

                                                                ArPirmaMetOp =
ArPirmaMetOp + 1;

                                                                int[] Kolkas= new
int[StaklesMetalas[a1]];
                                                                // nunulinu reiksmes
                                                                for (int
a4=0;a4<StaklesMetalas[a1];a4++)
                                                                Kolkas[a4]=0;

                                                                //if (r==rowNum-1)
                                                                //
            System.out.println(" PASKUTINIS IRASAS");

                                                                for (int
a4=0;a4<StaklesMetalas[a1];a4++)
                                                                                {

            //System.out.println(OO[a1]+"  "+StaklesMedis[a4]);

                                                                                int Variantas
= 0;


            //ArPirmaMetOp

                                                                                // KAIP BUVO
//
            if(IrasaiMetalas[a1][a4]==0 && f1==1)
//
            Variantas=1;
//
//
            if(IrasaiMetalas[a1][a4]==1 && f1==1)
//
            Variantas=2;
//
//
            if(IrasaiMetalas[a1][a4]>1 && f1==1)
//
            Variantas=3;
```

```
            if(IrasaiMetalas[a1][a4]==0 && ArPirmaMetOp==1)

            Variantas=1;
                                                                    else

            if(IrasaiMetalas[a1][a4]==1 && ArPirmaMetOp==1)

            Variantas=2;
                                                                    else

            if(IrasaiMetalas[a1][a4]>1 && ArPirmaMetOp==1)

            Variantas=3;
                                                                    else

            if(IrasaiMetalas[a1][a4]==0 && f1>1)

            Variantas=4;
                                                                    else

            if(IrasaiMetalas[a1][a4]==1 && f1>1)

            Variantas=5;
                                                                    else

            if(IrasaiMetalas[a1][a4]>1 && f1>1)

            Variantas=6;
                                                                    else

            if(Variantas==0)

            System.out.println("Kazkas dar naujo - Variantas = " + Variantas);



                                                        switch (Variantas) {

                                                        case 1: //
(IrasaiMedis[a1][a4]==0 && f1==1) ++++++++++++++++++++++

            //System.out.println("Papuoleme ir case : " + Variantas);

            DarbinisN[r][1][f1]=0;

            break;
                                                        case 2: //
(IrasaiMedis[a1][a4]==1 && f1==1) ++++++++++++++++++++++

            //System.out.println("Papuoleme ir case : " + Variantas);
                                                                                //
Ziuriu ar telpu iki iraso

                                                        //
LaikaiMedis[a1][a4][0][0] - paskutinis [0], nes yra tik vienintelis irasas

            if(DarbinisN[r][0][f1]<LaikaiMetalas[a1][a4][0][0])

            {          DarbinisN[r][1][f1]=0;          }
```

```
          else // jeigu netelpu, pradedu gaminti po iraso

          {          DarbinisN[r][1][f1]=LaikaiMetalas[a1][a4][1][0];      }

          break;
                                                    case 3: //
(IrasaiMedis[a1][a4]>1 && f1==1) ++++++++++++++++++++++

          //System.out.println("Papuoleme ir case : " + Variantas);
                                                                          //
Ziuriu ar mano pirmoji operacija telpa i kuri nors tarpa tarp irasu
                                                              int Radau =0;

          for(int a2=0;a2<IrasaiMetalas[a1][a4];a2++)
                                                                          {

          //if (r==rowNum-1)

          //System.out.println(IrasaiMedis[a1][a4]);



          if (Radau==1)

                    break;

          else

          if (a2==0)

          {

                    if(DarbinisN[r][0][f1]<LaikaiMetalas[a1][a4][0][a2])

                              {DarbinisN[r][1][f1]=0;

                              break;

                              }

                    else {



                              for(int
a3=a2+1;a3<IrasaiMetalas[a1][a4];a3++) // sitas ciklas sukasi tol, kol
tenkinama viena is tu dvieju operaciju - BUTINAI!
                                                  // --------
                              // ieskau ar telpa i bet kuri tarpa


          if(DarbinisN[r][0][f1]<LaikaiMetalas[a1][a4][0][a3]-
LaikaiMetalas[a1][a4][1][a3-1])

                                                  {
          DarbinisN[r][1][f1]=LaikaiMetalas[a1][a4][1][a3-1];   break;     }

                                                  else // jeigu netelpa i tarpa
ziuriu ar paskutine operacija


          if(a3+1==IrasaiMetalas[a1][a4])
```

```
                                        {
          DarbinisN[r][1][f1]=LaikaiMetalas[a1][a4][1][a3];

                                        Radau=1;

                                        break;

                                        }}}

          else

          if(DarbinisN[r][0][f1]<LaikaiMetalas[a1][a4][0][a2]-
LaikaiMetalas[a1][a4][1][a2-1])

          {         DarbinisN[r][1][f1]=LaikaiMetalas[a1][a4][1][a2-1];
          break;    }

          else // teoriskai cia galetu kažkur nubegti sprendinys, taciau
nemanau, nes lygtais nera salygos jam tai padaryti

                    if(a2+1==IrasaiMetalas[a1][a4])

          {         DarbinisN[r][1][f1]=LaikaiMetalas[a1][a4][1][a2];
          break;    }

                                                  }

          break;
                                                  case 4: //
(IrasaiMedis[a1][a4]==0 && f1>1) ++++++++++++++++++++++

          //System.out.println("Papuoleme ir case : " + Variantas);

          DarbinisN[r][1][f1]=DarbinisN[r][2][f1-1];

          break;
                                                  case 5: //
(IrasaiMedis[a1][a4]==1 && f1>1) ++++++++++++++++++++++

          //System.out.println("Papuoleme ir case : " + Variantas);
                                                                        //
jei pataikau IKI vienintelio iraso
                                                                        if
(DarbinisN[r][2][f1-1]<LaikaiMetalas[a1][a4][0][0])
                                                                        {

          // jeigu   tilpau iki vienintelio iraso

          if (DarbinisN[r][2][f1-
1]+DarbinisN[r][0][f1]<LaikaiMetalas[a1][a4][0][0])

          {         DarbinisN[r][1][f1]=DarbinisN[r][2][f1-1]; }

          // jeigu netilpau iki vienintelio iraso

          else

          {         DarbinisN[r][1][f1]=LaikaiMetalas[a1][a4][1][0];      }
                                                                        }
                                                                        //
jei pataikau ANT vienintelio iraso
```

```
          else if (DarbinisN[r][2][f1-1]>=LaikaiMetalas[a1][a4][0][0] &&
DarbinisN[r][2][f1-1]<=LaikaiMetalas[a1][a4][1][0])

          {          DarbinisN[r][1][f1]=LaikaiMetalas[a1][a4][1][0];      }
                                                                          //
jei pataikau PO  vienintelio iraso

          else if (DarbinisN[r][2][f1-1]>LaikaiMetalas[a1][a4][1][0])

          {          DarbinisN[r][1][f1]=DarbinisN[r][2][f1-1]; }

          break;


                                                    case 6: //
(IrasaiMedis[a1][a4]>1 && f1>1)
                                                                          //
Ziuriu ar mano konkreti operacija telpa i kuri nors tarpa tarp irasu

          for(int a2=0;a2<IrasaiMetalas[a1][a4];a2++)
                                                                          //
tikrinu su pirmuoju irasu ant stakliu

          // taip turi buti


          if(a2==0)

          {
//
                    // Jeigu baigiu iki pirmojo iraso

                    if(DarbinisN[r][2][f1-1]<LaikaiMetalas[a1][a4][0][a2])
                                              // [a2]

                    {// Jeigu telpu

                              if( DarbinisN[r][2][f1-
1]+DarbinisN[r][0][f1]<LaikaiMetalas[a1][a4][0][a2])

                              {


          DarbinisN[r][1][f1]=DarbinisN[r][2][f1-1];


          //System.out.println(DarbinisN[r][1][f1]);

                                        break;



                              }

                              else {

                              // Jeigu netelpu

                                                  for(int
a3=a2+1;a3<IrasaiMetalas[a1][a4];a3++) // sitas ciklas sukasi tol, kol
tenkinama viena is tu dvieju operaciju - BUTINAI!
```

```
                                                                    // ieskau ar
telpa i bet kuri tarpa

          if(DarbinisN[r][0][f1]<LaikaiMetalas[a1][a4][0][a3]-
LaikaiMetalas[a1][a4][1][a3-1])

                                                                    {
          DarbinisN[r][1][f1]=LaikaiMetalas[a1][a4][1][a3-1];   break;     }

                                                                    else // jeigu
netelpa i tarpa ziuriu ar paskutine operacija

          if(a3+1==IrasaiMetalas[a1][a4])

                                                                    {
          DarbinisN[r][1][f1]=LaikaiMetalas[a1][a4][1][a3];     break;     }

                                        }

                    }

                    // Jeigu baigiu ant pirmo iraso

                    else if (DarbinisN[r][2][f1-
1]>=LaikaiMetalas[a1][a4][0][a2] && DarbinisN[r][2][f1-
1]<=LaikaiMetalas[a1][a4][1][a2])

                    {

                              if (a2+1==IrasaiMetalas[a1][a4])

                              {
          DarbinisN[r][1][f1]=LaikaiMetalas[a1][a4][1][a2];     break;     }

                              else

                              { // jeigu ne paskutinis irasas ant stakliu,
bet zinau, kad mano praeita operacija jau yra pasibaigus

                                        for(int
a3=a2+1;a3<IrasaiMetalas[a1][a4];a3++) // sitas ciklas sukasi tol, kol
tenkinama viena is tu dvieju operaciju - BUTINAI!

                                                            // ieskau ar telpa i
bet kuri tarpa

          if(DarbinisN[r][0][f1]<LaikaiMetalas[a1][a4][0][a3]-
LaikaiMetalas[a1][a4][1][a3-1])

                                                            {
          DarbinisN[r][1][f1]=LaikaiMetalas[a1][a4][1][a3-1];   break;     }

                                                            else // jeigu netelpa i
tarpa ziuriu ar paskutine operacija

          if(a3+1==IrasaiMetalas[a1][a4])

                                                            {
          DarbinisN[r][1][f1]=LaikaiMetalas[a1][a4][1][a3];     break;     }
```

```
                                }

                        }

                        // Jeigu baigiu po pirmo iraso

                        else if (DarbinisN[r][2][f1-
1]>LaikaiMetalas[a1][a4][1][a2])

                                {          if (a2+1==IrasaiMetalas[a1][a4])

                                                                {
            DarbinisN[r][1][f1]=DarbinisN[r][1][f1-1]; break;     }

                                                                else

                                                                { // jeigu ne
paskutinis irasas ant stakliu, bet zinau, kad mano praeita operacija jau yra
pasibaigus

                                                                                for(int
a3=a2+1;a3<IrasaiMetalas[a1][a4];a3++) // sitas ciklas sukasi tol, kol
tenkinama viena is tu dvieju operaciju - BUTINAI!

                                                                                                //
Irasas turi buti pasibaiges veliau, nei mano operacija

                                                                                                //
Jei mano pabaiga mazesne uz iraso pabaiga - baigiau ant iraso


            if(DarbinisN[r][2][f1]<LaikaiMetalas[a1][a4][1][a3] &&
DarbinisN[r][2][f1]>=LaikaiMetalas[a1][a4][0][a3])

                                                                                {


            if (a3+1==IrasaiMetalas[a1][a4])


            {          DarbinisN[r][1][f1]=LaikaiMetalas[a1][a4][1][a3];
            break;     }


            if(DarbinisN[r][0][f1]<LaikaiMetalas[a1][a4][0][a3+1]-
LaikaiMetalas[a1][a4][1][a3])


            {          DarbinisN[r][1][f1]=LaikaiMetalas[a1][a4][1][a3];
break;     }


            if(DarbinisN[r][0][f1]<LaikaiMetalas[a1][a4][0][a3+1]-
Math.max(DarbinisN[r][2][f1-1],LaikaiMetalas[a1][a4][1][a3]))


            {          DarbinisN[r][1][f1]=Math.max(DarbinisN[r][2][f1-
1],LaikaiMetalas[a1][a4][1][a3]);          break;     }



                                                                                }
```

```
            else       {
            if(a3+1==IrasaiMetalas[a1][a4])


            {          DarbinisN[r][1][f1]=LaikaiMetalas[a1][a4][1][a3];
            break;     }}}}}

            else

            { // irasas ne pirmas



                        // Pataikau i tarpa

                        if (DarbinisN[r][2][f1-1]<LaikaiMetalas[a1][a4][0][a2]
&& DarbinisN[r][2][f1-1]>LaikaiMetalas[a1][a4][1][a2-1])

                        {

                                    // ir jeigu telpu  i tarpa

                                    if(DarbinisN[r][2][f1-
1]+DarbinisN[r][0][f1]<LaikaiMetalas[a1][a4][0][a2])

                                    {
            DarbinisN[r][1][f1]=DarbinisN[r][2][f1-1]; break;     }

                                    else { // jeigu i tarpa netilpau

                                                // jei tai buvo paskutinis irasas
ant takliu

                                                if (a2+1==IrasaiMetalas[a1][a4])

                                                {
            DarbinisN[r][1][f1]=LaikaiMetalas[a1][a4][1][a2];     break;     }

                                                else { // jeigu ne paskutinis
irasas ant stakliu, bet zinau, kad mano praeita operacija jau yra pasibaigus

                                                            for(int
a3=a2+1;a3<IrasaiMetalas[a1][a4];a3++) // sitas ciklas sukasi tol, kol
tenkinama viena is tu dvieju operaciju - BUTINAI!

                                                                        // ieskau ar
telpa i bet kuri tarpa


            if(DarbinisN[r][0][f1]<LaikaiMetalas[a1][a4][0][a3]-
LaikaiMetalas[a1][a4][1][a3-1])

                                                                        {
            DarbinisN[r][1][f1]=LaikaiMetalas[a1][a4][1][a3-1];   break;     }

                                                                        else // jeigu
netelpa i tarpa ziuriu ar paskutine operacija


            if(a3+1==IrasaiMetalas[a1][a4])
```

```
                                                                {
          DarbinisN[r][1][f1]=LaikaiMetalas[a1][a4][1][a3];     break;     }
                                                          }}}
Else
{

                                  // jeigu pataikau ant iraso

                                  if (DarbinisN[r][2][f1-
1]>=LaikaiMetalas[a1][a4][0][a2] && DarbinisN[r][2][f1-
1]<LaikaiMetalas[a1][a4][1][a2])

                                  {

                                            if (a2+1==IrasaiMetalas[a1][a4])

                                            {
          DarbinisN[r][1][f1]=LaikaiMetalas[a1][a4][1][a2];     break;     }

                                            else

                                            { // jeigu ne paskutinis irasas
ant stakliu, bet zinau, kad mano praeita operacija jau yra pasibaigus

                                                      for(int
a3=a2+1;a3<IrasaiMetalas[a1][a4];a3++) // sitas ciklas sukasi tol, kol
tenkinama viena is tu dvieju operaciju - BUTINAI!

                                                                // ieskau ar
telpa i bet kuri tarpa


          if(DarbinisN[r][0][f1]<LaikaiMetalas[a1][a4][0][a3]-
LaikaiMetalas[a1][a4][1][a3-1])

                                                                {
          DarbinisN[r][1][f1]=LaikaiMetalas[a1][a4][1][a3-1];   break;     }

                                                                else // jeigu
netelpa i tarpa ziuriu ar paskutine operacija


          if(a3+1==IrasaiMetalas[a1][a4])

                                                                {
          DarbinisN[r][1][f1]=LaikaiMetalas[a1][a4][1][a3];     break;     }

                                                      }



                                  }

                                  else

                                  {

                                            if (a2+1==IrasaiMetalas[a1][a4])

                                            {
          DarbinisN[r][1][f1]=Math.max(DarbinisN[r][2][f1-
1],LaikaiMetalas[a1][a4][1][a2]);          break;     }
```

```
                                        //else // prasuka tusciai, nes
irasas nera paskutinis, bet mano operacija baigiasi veliau uz irasa

                                        //        System.out.println("
DAR VIENAS KAZKOKS ATVEJIS, KURIO NEAPGALVOJAU   ");

                                }}}     break;
                                                        }


          Kolkas[a4]=DarbinisN[r][1][f1];
                                                        // baigiasi
a4 ciklas
                                        }

                                                int StakliuNr=0;
                                                StakliuNr =
MinIkiInd(Kolkas,-1,StaklesMetalas[a1]);



          DarbinisN[r][1][f1]=Kolkas[StakliuNr];


          DarbinisN[r][2][f1]=DarbinisN[r][1][f1]+DarbinisN[r][0][f1];

//
          DarbinisMetalasN=DarbinisN;

                                                        // IRASU
MASYVAS TURI ATRODYTI: IrasaiMedis[a1][st] - a1  - operacijos rusis, st -
stakliu numeris
//SITUS PASKUI REIKIA ATKOMENTUOTI IR SUTVARKYTI




                                                        for (int
a=1;a<n3;a++)
                                                                if
(OOO[a].toLowerCase().contains(DarbinisT[r][f1].toLowerCase()))
                                                                {

                                        DarbinisT[r][f1]=OOO[a+StakliuNr];

          DarbinisMetalasT[r][f1]=OOO[a+StakliuNr];

          //DarbinisMetalasT[r][f1]=DarbinisT[r][f1]; NEBANDYK SITO NAUDOT


//
          System.out.println("Iprastas:    "+DarbinisT[r][f1]+"  Kitas:
          "+DarbinisMetalasT[r][f1]);



          DarbinisMetalasN[r][2][f1]=DarbinisN[r][2][f1];

          DarbinisMetalasN[r][1][f1]=DarbinisN[r][1][f1];

          DarbinisMetalasN[r][0][f1]=DarbinisN[r][0][f1];

                                                        break;
```

```
                                                                    }

            LaikaiMetalas[a1][StakliuNr][0][IrasaiMetalas[a1][StakliuNr]]=Darbi
nisN[r][1][f1];

            LaikaiMetalas[a1][StakliuNr][1][IrasaiMetalas[a1][StakliuNr]]=Darbi
nisN[r][2][f1];

            LaikaiMetalas[a1][StakliuNr][2][IrasaiMetalas[a1][StakliuNr]]=Darbi
nisN[r][0][f1];

//
            System.out.println();

//
            System.out.println("Operacija  "+OOMetalas[a1] + "   StakliuNr
"+StakliuNr);
//
            System.out.println("Irasu sk PRIES priskyrimo: "+
IrasaiMetalas[a1][StakliuNr]);

            IrasaiMetalas[a1][StakliuNr] = IrasaiMetalas[a1][StakliuNr] + 1;
//
            System.out.println("Irasu sk PO priskyrimo: "+
IrasaiMetalas[a1][StakliuNr]);


//
            System.out.println();



            if (FinishMetalas < DarbinisN[r][2][f1])

            {FinishMetalas=DarbinisN[r][2][f1];

            //System.out.println("Paskutinis Medzio irasas : "+ a21);

            }

            break;
                                              }}

                                        // baigiasi operaciju ziurejimo ciklas
                        }
                        LaikaiTikrasMedis=LaikaiMedis;
                        LaikaiTikrasMetalas=LaikaiMetalas;
                        IrasaiTikrasMedis=IrasaiMedis;
                        IrasaiTikrasMetalas=IrasaiMetalas;
                        DarbinisTikrasT=DarbinisT;
                        DarbinisTikrasN=DarbinisN;

DarbinisT[r][MedSk]+ "   Last Metalo op: " + DarbinisT[r][MetSk]);
OPERACIJU RUSIES
```

```
                    int ArBandomAbu = 1;

                    if (ArBandomAbu == 1){


                    int IsAbieju = 0;
                    int Pradzia = Math.max(FinishMedis, FinishMetalas);


                    for(int f1=1;f1<1+k[0][nn0[r]];f1++)
                    {


                              for (int a1=1;a1<n23;a1++)
                                                  {

                                        // cia iskarto suku dar viena
cikla ir einu per stakles
                                                                  if
(DarbinisT[r][f1].toLowerCase().contains(OOAbu[a1].toLowerCase()))
                                                                  {

          //System.out.println("Gaminys: " + S1dataT[r][1]);

          //System.out.println("Operacija: " + DarbinisT[r][f1]);

          IsAbieju = IsAbieju + 1;


          int[] Kolkas= new int[StaklesAbu[a1]];
                                                                                //
nunulinu reiksmes

          for (int a4=0;a4<StaklesAbu[a1];a4++)

          Kolkas[a4]=0;


          //if (r==rowNum-1)
                                                                                //
          System.out.println(" PASKUTINIS IRASAS");


          for (int a4=0;a4<StaklesAbu[a1];a4++)

          {

          //System.out.println(OO[a1]+"  "+StaklesMedis[a4]);


          //FinishMedis = FinishMetalas;


          int Variantas = 0;


          if(IrasaiAbu[a1][a4]==0 && IsAbieju==1)

                    Variantas=1;

          else
```

```
            if(IrasaiAbu[a1][a4]==1 && IsAbieju==1)
                        Variantas=2;
            else
            if(IrasaiAbu[a1][a4]>1 && IsAbieju==1)
                        Variantas=3;


            else
            if(IrasaiAbu[a1][a4]==0 && IsAbieju>1)
                        Variantas=4;
            else
            if(IrasaiAbu[a1][a4]==1 && IsAbieju>1)
                        Variantas=5;
            else
            if(IrasaiAbu[a1][a4]>1 && IsAbieju>1)
                        Variantas=6;
            else
            if(Variantas==0)
                        System.out.println("Kazkas dar naujo - Variantas = " +
Variantas);
            switch (Variantas) {


            case 1: // (Irasai[a1][a4]==0 && f1==1) ++++++++++++++++++++++
                                    //System.out.println("Papuoleme ir case : " +
Variantas);
                                    DarbinisN[r][1][f1]=Pradzia;
                                    break;
            case 2: // (Irasai[a1][a4]==1 && f1==1) ++++++++++++++++++++++
                                    //System.out.println("Papuoleme ir case : " +
Variantas);
                                    // Ziuriu ar telpu iki iraso

            // Laikai[a1][a4][0][0] - paskutinis [0], nes yra tik vienintelis
irasas

            if(Pradzia+DarbinisN[r][0][f1]<LaikaiAbu[a1][a4][0][0])
```

```
                                        {
            DarbinisN[r][1][f1]=Pradzia;     }

                                else // jeigu netelpu, pradedu gaminti po
iraso

                                        {
            DarbinisN[r][1][f1]=Math.max(Pradzia,LaikaiAbu[a1][a4][1][0]);   }

                                break;




            case 3: // (Irasai[a1][a4]>1 && f1==1) ++++++++++++++++++++++


                        //System.out.println("Operacija: " + DarbinisT[r][f1]);


                                // Ziuriu ar mano pirmoji operacija telpa i
kuri nors tarpa tarp irasu
//
                        int ArTuriuCia = 0;


                        int JauTuriuKoReikia0 = 0;
//
                        int RastaPradzia0;
//
                        int RastaPabaiga0;

                        int RastaPradziaInd0 = -1;

                        int RastaPabaigaInd0 = -1;


                        for(int k1=0;k1<IrasaiAbu[a1][a4];k1++)

                        {

                                    if(LaikaiAbu[a1][a4][0][k1]>=Pradzia)

                                    {
//
            RastaPradzia0=LaikaiAbu[a1][a4][0][k1];

                                                RastaPradziaInd0=k1;

                                                break;

                                    }

                        }


                        for(int k1=0;k1<IrasaiAbu[a1][a4];k1++)
```

```
                        {
                                    if(LaikaiAbu[a1][a4][1][k1]>Pradzia)
                                    {
//
          RastaPabaiga0=LaikaiAbu[a1][a4][1][k1];
                                                RastaPabaigaInd0=k1;
                                                break; }}
                                                                        //if
(IsAbieju==1)
                        //{        // Jeigu mano pabaiga veliau nei visu irasu
pradzios ir pabaigos
                                    if (RastaPradziaInd0 == -1 &&
RastaPabaigaInd0 == -1)
                                    {
                                                DarbinisN[r][1][f1]=Pradzia;
                                                break;
                                    }
                                    else
                                    // Jeigu mano pabaiga daugiau uz paskutinio
iraso pradzia, bet maziau uz jo pabaiga
                                    if (RastaPradziaInd0 == -1 &&
RastaPabaigaInd0 != -1)
                                    {

          DarbinisN[r][1][f1]=LaikaiAbu[a1][a4][1][RastaPabaigaInd0];
                                                break;
                                    }
                                    else


                                    if (RastaPradziaInd0 == RastaPabaigaInd0) //
PATAIKAU I PRIES IRASA
                                    {
                                                // =
                                                if
(DarbinisN[r][0][f1]<(LaikaiAbu[a1][a4][0][RastaPradziaInd0]-Pradzia))
                                                {
```

```
            DarbinisN[r][1][f1]=Pradzia;
                                                    break;

                                        }
                                        else
                                        {

            if(RastaPradziaInd0+1==IrasaiAbu[a1][a4])

                                                    {


            DarbinisN[r][1][f1]=LaikaiAbu[a1][a4][1][RastaPradziaInd0];

                                                                break;
                                                    }

                                                    else

                                                    {

                                                                for(int
k2=RastaPradziaInd0+1;k2<IrasaiAbu[a1][a4];k2++)

                                                                {

            if(DarbinisN[r][0][f1]<LaikaiAbu[a1][a4][0][k2]-
LaikaiAbu[a1][a4][1][k2-1])

            {

            DarbinisN[r][1][f1]=LaikaiAbu[a1][a4][1][k2-1];

            JauTuriuKoReikia0 = 1;

            break;

            }

            else // jeigu netelpa i tarpa ziuriu ar paskutine operacija


            if(k2+1==IrasaiAbu[a1][a4])


                        {
                        DarbinisN[r][1][f1]=LaikaiAbu[a1][a4][1][k2];
                        JauTuriuKoReikia0 = 1;

            break;
                        }

                                                                }

                                    if (JauTuriuKoReikia0 == 1)


            break;                  }}}

                                    else
```

```
                                // Anas jau DONE :) - realiai visiskai toks
pat kodas, tik kazkur reikia deti tiktais

                                // RastaPabaiga, bet ne RastaPradzia -
pasiziureti ATIDZIAI!!!.

                                if (RastaPradziaInd0 > RastaPabaigaInd0)

                                {

                                        if
(DarbinisN[r][0][f1]<=(LaikaiAbu[a1][a4][0][RastaPradziaInd0] -
LaikaiAbu[a1][a4][1][RastaPabaigaInd0]))

                                        {


          DarbinisN[r][1][f1]=LaikaiAbu[a1][a4][1][RastaPabaigaInd0];

                                                break;

                                        }

                                        else

                                        {


          if(RastaPradziaInd0+1==IrasaiAbu[a1][a4])

                                                {


          DarbinisN[r][1][f1]=LaikaiAbu[a1][a4][1][RastaPradziaInd0];


          //DarbinisN[r][1][f1]=Math.max(DarbinisN[r][2][f1-
1],LaikaiAbu[a1][a4][1][RastaPradziaInd0]);

                                                        break;

                                                }

                                                else

                                                {

                                                        for(int
k2=RastaPradziaInd0+1;k2<IrasaiAbu[a1][a4];k2++)

                                                        {


          if(DarbinisN[r][0][f1]<LaikaiAbu[a1][a4][0][k2]-
LaikaiAbu[a1][a4][1][k2-1])


          {


          DarbinisN[r][1][f1]=LaikaiAbu[a1][a4][1][k2-1];
```

```
            JauTuriuKoReikia0 = 1;

            break;
            }
                                                                    else // jeigu
netelpa i tarpa ziuriu ar paskutine operacija

            if(k2+1==IrasaiAbu[a1][a4])

                        {
                        DarbinisN[r][1][f1]=LaikaiAbu[a1][a4][1][k2];

                        JauTuriuKoReikia0 = 1;

                        break;

                        }

                                                                    }

                                                                    if
(JauTuriuKoReikia0 == 1)

            break; }}}

                                    break;

            case 4: // (Irasai[a1][a4]==0 && f1>1) ++++++++++++++++++++++

                                    //System.out.println("Papuoleme ir case : " +
Variantas);

                                    DarbinisN[r][1][f1]=DarbinisN[r][2][f1-1];

                                    break;

            case 5: // (Irasai[a1][a4]==1 && f1>1) ++++++++++++++++++++++

                                    //System.out.println("Papuoleme ir case : " +
Variantas);

                                    // jei pataikau IKI vienintelio iraso

                                    if (DarbinisN[r][2][f1-
1]<LaikaiAbu[a1][a4][0][0])

                                    {

                                                // jeigu   tilpau iki vienintelio
iraso

                                                if (DarbinisN[r][2][f1-
1]+DarbinisN[r][0][f1]<LaikaiAbu[a1][a4][0][0])
```

```
                                        {
            DarbinisN[r][1][f1]=DarbinisN[r][2][f1-1]; }

                                        // jeigu netilpau iki vienintelio
iraso

                                        else

                                        {
            DarbinisN[r][1][f1]=LaikaiAbu[a1][a4][1][0];          }

                                }

                                // jei pataikau ANT vienintelio iraso

                                else if (DarbinisN[r][2][f1-
1]>=LaikaiAbu[a1][a4][0][0] && DarbinisN[r][2][f1-
1]<=LaikaiAbu[a1][a4][1][0])

                                        {
            DarbinisN[r][1][f1]=LaikaiAbu[a1][a4][1][0];          }

                                // jei pataikau PO  vienintelio iraso

                                else if (DarbinisN[r][2][f1-
1]>LaikaiAbu[a1][a4][1][0])

                                        {
            DarbinisN[r][1][f1]=DarbinisN[r][2][f1-1]; }

                                break;


            case 6: // (Irasai[a1][a4]>1 && f1>1)

                                //System.out.println("Papuoleme ir case : " +
Variantas);

                                // Ziuriu ar mano konkreti operacija telpa i
kuri nors tarpa tarp irasu

                        //DarbinisN[r][1][f1]=600;


                        int Testuotuojam = 1;


                        if (Testuotuojam == 0)

                        {

                                DarbinisN[r][1][f1]=600;

                        }

                        else

                        {
```

```
                        int JauTuriuKoReikia1 = 0;
//
                        int RastaPradzia1;
//
                        int RastaPabaiga1;
                        int RastaPradziaInd1 = -1;
                        int RastaPabaigaInd1 = -1;


                        for(int k1=0;k1<IrasaiAbu[a1][a4];k1++)
                        {

            if(LaikaiAbu[a1][a4][0][k1]>=DarbinisN[r][2][f1-1])
                                    {
//
            RastaPradzia1=LaikaiAbu[a1][a4][0][k1];
                                                RastaPradziaInd1=k1;
                                                break;
                                    }
                        }
                                                                    for(int
k1=0;k1<IrasaiAbu[a1][a4];k1++)
                        {

            if(LaikaiAbu[a1][a4][1][k1]>=DarbinisN[r][2][f1-1])
                                    {
//
            RastaPabaiga1=LaikaiAbu[a1][a4][1][k1];
                                                RastaPabaigaInd1=k1;
                                                break;
                                    }
                        }
                        if (RastaPradziaInd1 == -1 && RastaPabaigaInd1 == -1)
                        {
                                    DarbinisN[r][1][f1]=DarbinisN[r][2][f1-1];
                                    break;
                        }
                        else
```

```
                    // Jeigu mano pabaiga daugiau uz paskutinio iraso
pradzia, bet maziau uz jo pabaiga

                    if (RastaPradziaInd1 == -1 && RastaPabaigaInd1 != -1) //
- iskarto surandu paskutini irasa

                    {

                              // Uztektu tiesiog Laikai[][][][] , nes as ir
taip ieskau iraso pabaigos didesnes uz mano pradzia (kai nera iraso pradzios,
didesnes uz mano pradzia


          DarbinisN[r][1][f1]=Math.max(DarbinisN[r][2][f1-
1],LaikaiAbu[a1][a4][1][RastaPabaigaInd1]);


          //DarbinisN[r][1][f1]=,LaikaiAbu[a1][a4][1][RastaPabaigaInd]; // -
vat va taip turi atrodyti

                              break;

                    }

                    else



                    if (RastaPradziaInd1 == RastaPabaigaInd1) // PATAIKAU
PRIES IRASA

                    {// =

                              if
(DarbinisN[r][0][f1]<(LaikaiAbu[a1][a4][0][RastaPradziaInd1]-
DarbinisN[r][2][f1-1]))

                              {


          DarbinisN[r][1][f1]=DarbinisN[r][2][f1-1];

                                        break;

                              }

                              else

                              {


          if(RastaPabaigaInd1+1==IrasaiAbu[a1][a4])

                                        {


          DarbinisN[r][1][f1]=LaikaiAbu[a1][a4][1][RastaPabaigaInd1];

                                                  break;

                                        }

                                        else
```

```
                                        {
                                                for(int
k2=RastaPabaigaInd1+1;k2<IrasaiAbu[a1][a4];k2++)
                                                {

          if(DarbinisN[r][0][f1]<LaikaiAbu[a1][a4][0][k2]-
LaikaiAbu[a1][a4][1][k2-1])
                                                                        {
          DarbinisN[r][1][f1]=LaikaiAbu[a1][a4][1][k2-1];

          JauTuriuKoReikia1 = 1;

          break;
                                                                        }
                                                        else // jeigu
netelpa i tarpa ziuriu ar paskutine operacija

          if(k2+1==IrasaiAbu[a1][a4])

          {

          DarbinisN[r][1][f1]=LaikaiAbu[a1][a4][1][k2];

          JauTuriuKoReikia1 = 1;

          break;

          }}
          if (JauTuriuKoReikia1 == 1)
                              break;
                                        }}}
                    else

                    // Anas jau DONE :) - realiai visiskai toks pat kodas,
tik kazkur reikia deti tiktais

                    // RastaPabaiga, bet ne RastaPradzia - pasiziureti
ATIDZIAI!!!.

                    if (RastaPradziaInd1 > RastaPabaigaInd1)

                    {

                              if
(DarbinisN[r][0][f1]<=(LaikaiAbu[a1][a4][0][RastaPradziaInd1] -
LaikaiAbu[a1][a4][1][RastaPabaigaInd1]))
```

```
                        {

            DarbinisN[r][1][f1]=LaikaiAbu[a1][a4][1][RastaPabaigaInd1];

                                    break;

                        }

                        else

                        {

            if(RastaPradziaInd1+1==IrasaiAbu[a1][a4])

                                    {

            DarbinisN[r][1][f1]=LaikaiAbu[a1][a4][1][RastaPradziaInd1];

                                            break;

                                    }

                                    else

                                    {

                                            for(int
k2=RastaPradziaInd1+1;k2<IrasaiAbu[a1][a4];k2++)

                                            {

            if(DarbinisN[r][0][f1]<LaikaiAbu[a1][a4][0][k2]-
LaikaiAbu[a1][a4][1][k2-1])

                                                                    {

            DarbinisN[r][1][f1]=LaikaiAbu[a1][a4][1][k2-1];

            JauTuriuKoReikia1 = 1;

            break;

                                                                    }

                                                            else // jeigu
netelpa i tarpa ziuriu ar paskutine operacija

            if(k2+1==IrasaiAbu[a1][a4])

            {

            DarbinisN[r][1][f1]=LaikaiAbu[a1][a4][1][k2];
```

```
            JauTuriuKoReikia1 = 1;

            break;

            }}
                        if (JauTuriuKoReikia1 == 1)
                                                break;

                                                }}}}

                                    break;
                        }

            //if (Variantas == 3)

            //System.out.println("Siuloma pradzia: " + DarbinisN[r][1][f1]);


            Kolkas[a4]=DarbinisN[r][1][f1];

            // baigiasi a4 ciklas
                                                                                }


            int StakliuNr=0;

            StakliuNr = MinIkiInd(Kolkas,-1,StaklesAbu[a1]);


            DarbinisN[r][1][f1]=Kolkas[StakliuNr];

            DarbinisN[r][2][f1]=DarbinisN[r][1][f1]+DarbinisN[r][0][f1];

//
            DarbinisMedisN=DarbinisN;

            // IRASU MASYVAS TURI ATRODYTI: IrasaiAbu[a1][st] - a1  -
operacijos rusis, st - stakliu numeris

            // SITUS PASKUI REIKIA ATKOMENTUOTI IR SUTVARKYTI

                                                        for (int a=1;a<n3;a++)

                        if
(OOO[a].toLowerCase().contains(DarbinisT[r][f1].toLowerCase()))

                        {

            DarbinisT[r][f1]=OOO[a+StakliuNr];

                                    break;
```

```
                    }


          LaikaiAbu[a1][StakliuNr][0][IrasaiAbu[a1][StakliuNr]]=DarbinisN[r][
1][f1];

          LaikaiAbu[a1][StakliuNr][1][IrasaiAbu[a1][StakliuNr]]=DarbinisN[r][
2][f1];

          LaikaiAbu[a1][StakliuNr][2][IrasaiAbu[a1][StakliuNr]]=DarbinisN[r][
0][f1];


                              IrasaiAbu[a1][StakliuNr] =
IrasaiAbu[a1][StakliuNr] + 1;



          break;     }}}}

                    // SURIKIUOJA DUOMENYS LAIKO MASYVE, KAD VISOS PRADZIOS
IR PABAIGOS BUTU GRAZIAI NUOSEKLIOS

//                  if(r>1)
//                  {
                              for(int a1=1;a1<n2;a1++)
                              {
                                        for (int
a4=0;a4<StaklesMedis[a1];a4++)
                                        {
                                                  // Sitoj vietoj
patikrinti, kaip atsiranda 8 sitame IrasaiTikrasMedis[a1][a4], nes tiek
neturi cia buti :)

          Arrays.sort(LaikaiMedis[a1][a4][0], 0, (IrasaiMedis[a1][a4])); //
Sitoj vietoj patikrinti, kaip atsiranda 8, nes tiek neturi cia buti :)

          Arrays.sort(LaikaiMedis[a1][a4][1], 0, (IrasaiMedis[a1][a4]));
                                        for (int
a2=0;a2<IrasaiMedis[a1][a4];a2++)

          LaikaiMedis[a1][a4][2][a2]=LaikaiMedis[a1][a4][1][a2]-
LaikaiMedis[a1][a4][0][a2];
                                        }
                              }

                              for(int a1=1;a1<n22;a1++)
                              {
                                        for (int
a4=0;a4<StaklesMetalas[a1];a4++)
                                        {

          Arrays.sort(LaikaiMetalas[a1][a4][0], 0, (IrasaiMetalas[a1][a4]));

          Arrays.sort(LaikaiMetalas[a1][a4][1], 0, (IrasaiMetalas[a1][a4]));
                                        for (int
a2=0;a2<IrasaiMetalas[a1][a4];a2++)

          LaikaiMetalas[a1][a4][2][a2]=LaikaiMetalas[a1][a4][1][a2]-
LaikaiMetalas[a1][a4][0][a2];
                                        }
                              }
```

```
                                for(int a1=1;a1<n23;a1++)
                                {
                                          for (int
a4=0;a4<StaklesAbu[a1];a4++)
                                          {
                                          Arrays.sort(LaikaiAbu[a1][a4][0],
0, (IrasaiAbu[a1][a4]));
                                          Arrays.sort(LaikaiAbu[a1][a4][1],
0, (IrasaiAbu[a1][a4]));
                                          for (int
a2=0;a2<IrasaiAbu[a1][a4];a2++)

          LaikaiAbu[a1][a4][2][a2]=LaikaiAbu[a1][a4][1][a2]-
LaikaiAbu[a1][a4][0][a2];
                                          }
                                }
                     }


int[] GamybosTrukme=new int [3]; // [0] - bus medis, [1] - bus metalas, [2] -
bus Abu (nors nezinau ar reikia)

                     // MEDIS
                     //PrastEa[1]=0;
                     for(int a1=1;a1<n2;a1++)
                     {
                                for (int a4=0;a4<StaklesMedis[a1];a4++)
                                {
                                for (int a2=0;a2<IrasaiMedis[a1][a4];a2++)
                                {
                                          if
(GamybosTrukme[0]<LaikaiMedis[a1][a4][1][a2])
                                                    {

          GamybosTrukme[0]=LaikaiMedis[a1][a4][1][a2];
                                                    }
                                          if
(VisosGamybosTrukme<LaikaiMedis[a1][a4][1][a2])
                                                    {

          VisosGamybosTrukme=LaikaiMedis[a1][a4][1][a2];
                                                    }
                                          if(a2==0)  // kitais atvejais, cia
bus tiesiog "else" dalis ir viskas
                                          {
          PrastEaMedis[a1][a4]=LaikaiMedis[a1][a4][0][a2];      }
                                          else
                                          {
          PrastEaMedis[a1][a4]=PrastEaMedis[a1][a4]+LaikaiMedis[a1][a4][0][a2
]-LaikaiMedis[a1][a4][1][a2-1];  }
                                }
                     }
                     }

                     // METALAS
                                for(int a1=1;a1<n22;a1++)
                                {
                                          for (int
a4=0;a4<StaklesMetalas[a1];a4++)
                                          {
                                          for (int
a2=0;a2<IrasaiMetalas[a1][a4];a2++)
                                          {
```

```
                                                    if
(GamybosTrukme[1]<LaikaiMetalas[a1][a4][1][a2])
                                                    {

            GamybosTrukme[1]=LaikaiMetalas[a1][a4][1][a2];
                                                    }
                                                    if
(VisosGamybosTrukme<LaikaiMetalas[a1][a4][1][a2])
                                                    {

            VisosGamybosTrukme=LaikaiMetalas[a1][a4][1][a2];
                                                    }
                                                    if(a2==0)  // kitais
atvejais, cia bus tiesiog "else" dalis ir viskas
                                                    {
            PrastEaMetalas[a1][a4]=LaikaiMetalas[a1][a4][0][a2];  }
                                                    else
                                                    {
            PrastEaMetalas[a1][a4]=PrastEaMetalas[a1][a4]+LaikaiMetalas[a1][a4]
[0][a2]-LaikaiMetalas[a1][a4][1][a2-1];    }
                                                    }
                                    }
                                    }



                                    // ABU
                                    for(int a1=1;a1<n23;a1++)
                                    {
                                              for (int
a4=0;a4<StaklesAbu[a1];a4++)
                                              {
                                              for (int
a2=0;a2<IrasaiAbu[a1][a4];a2++)
                                              {
                                                    if
(GamybosTrukme[2]<LaikaiAbu[a1][a4][1][a2])
                                                    {

            GamybosTrukme[2]=LaikaiAbu[a1][a4][1][a2];
//
            System.out.println(GamybosTrukme[2]);
                                                    }
                                                    if
(VisosGamybosTrukme<LaikaiAbu[a1][a4][1][a2])
                                                    {

            VisosGamybosTrukme=LaikaiAbu[a1][a4][1][a2];
//
            System.out.println(VisosGamybosTrukme);
                                                    }
                                                    if(a2==0)  // kitais
atvejais, cia bus tiesiog "else" dalis ir viskas
                                                    {
            PrastEaAbu[a1][a4]=LaikaiAbu[a1][a4][0][a2];          }
                                                    else
                                                    {
            PrastEaAbu[a1][a4]=PrastEaAbu[a1][a4]+LaikaiAbu[a1][a4][0][a2]-
LaikaiAbu[a1][a4][1][a2-1];      }
                                              }
                                    }
                                    }
```

```
//                                for (int i=21;i<29;i++)
//                                {
//
//                                System.out.println((i+1)+"   Operacija :
"+OOO[i]);
//
//                                }

                        for ( int r=1; r<rowNum; r++)
                        { //       VAIZDAVIMO PRADZIA

                                  //System.out.println();

//                                System.out.println("   GAMINYS:   "+r);

                                  for(int f1=1;f1<1+k[0][nn0[r]];f1++)
                                  {
                                            //System.out.println("
          Operacija:   "+DarbinisTikrasT[r][f1]);
                                            for(int p=1;p<n3;p++)
                                                      {

          if(OOO[p].equals(DarbinisTikrasT[r][f1]))   // TIKRINA AR NORIMA
ATLIKTI OPERACIJA SUTAMPA SU ATLIEKAMOMIS OPERACIJOMIS (SHEET 3)
                                                                        {


          //System.out.println((p+1)+"    Operacija :   "+OOO[p]);
//
          System.out.println((p+1)+"      "+OOO[p]);

                                                                                  //
VISI GAMINIU LAIKAI, VIENO GAMINIO VISU OPERACIJU LAIKAI, TADA KITO GAMINIO
VISU OPERACIJU LAIKAI

          OO2[p][r][0]=DarbinisTikrasN[r][0][f1];

          OO2[p][r][1]=DarbinisTikrasN[r][1][f1];

          OO2[p][r][2]=DarbinisTikrasN[r][2][f1];

          //System.out.println("Operacija: "+OO[p]+"  Pradzia =
"+OO2[p][r][1]+"  Pabaiga = "+OO2[p][r][2]+"  Trukme = "+OO2[p][r][0]);

          break;
                                                                        }

                                                      }
                        }

                        for(int i=1;i<n3;i++)
                        {

                                  Pab[0][i][0]=OOO[i];
                                  Pab[r][i][0]=String.valueOf(OO2[i][r][0]);
                                  Pab[r][i][1]=String.valueOf(OO2[i][r][1]);
                                  Pab[r][i][2]=String.valueOf(OO2[i][r][2]);


                        }
                        }


                        // Skaiciuoja prastovas
```

```
                    for(int a1=1;a1<n2;a1++)
                              for (int a4=0;a4<StaklesMedis[a1];a4++)

          Prastovos=Prastovos+PrastEaMedis[a1][a4];

                    for(int a1=1;a1<n22;a1++)
                              for (int a4=0;a4<StaklesMetalas[a1];a4++)

          Prastovos=Prastovos+PrastEaMetalas[a1][a4];

                    for(int a1=1;a1<n23;a1++)
                              for (int a4=0;a4<StaklesAbu[a1];a4++)

          Prastovos=Prastovos+PrastEaAbu[a1][a4];




//                  System.out.println("Visos gamybos prastovos:
"+Prastovos);
//                  System.out.println("Visa gamybos trukme:
"+VisosGamybosTrukme);

//                  TiksloFunkcija = 1;


                    Pab[1][0][0]=String.valueOf(Prastovos);
                    Pab[2][0][0]=String.valueOf(VisosGamybosTrukme);

//                  System.out.println(Pab[2][0][0]);
//                  System.out.println(VisosGamybosTrukme);

                    switch (TiksloFunkcija)
                    {
                    case 1:
                              Pab[0][0][0]=String.valueOf(Prastovos);
                              break;
                    case 2:

          Pab[0][0][0]=String.valueOf(VisosGamybosTrukme);
                              break;
                    default:

Pab[0][0][0]=String.valueOf(0);
                              break;
                    }




                    return Pab[nn1][nn2][nn3];
          }



          public static void Mutacija_2(int[] A)
          {
                    int e=A[1];
                    A[1]=A[rowNum-1];
                    A[rowNum-1]=e;
          }

          public static void Mutacija_3(int[] A)
          {
```

```
                int e=A[2];
                A[2]=A[rowNum-2];
                A[rowNum-2]=e;
        }

        public static void Mutacija_4(int[] A)
        {
                int e=A[1];
                A[1]=A[rowNum-2];
                A[rowNum-2]=A[rowNum-1];
                A[rowNum-1]=A[2];
                A[2]=e;
        }

        public static void Mutacija_5(int[] A)
        {
                for (int i=1; i<10;i++)
            {
                 int randomPosition = i + (int) (Math.random() * (4-i));
                    int temp = A[i];
                    A[i] = A[randomPosition];
                    A[randomPosition] = temp;
            }
        }

        public static void Mutacijos(int[] A, int mutac)
        {
                switch (mutac) {
                case 1: MAISYMAS(A);
                           break;
                case 2: Mutacija_2(A);
                break;
                case 3: Mutacija_3(A);
                break;
                case 4: Mutacija_4(A);
                break;
                case 5: Mutacija_5(A);
                break;
                }
        }

        public static void Genetinis(int[][] se, int geruChromNr, int
bloguChromNr,int mutac, int MutavProc, int Iterac) throws IOException
        {


                double start1 = System.currentTimeMillis();

                double start = System.currentTimeMillis();

                File file = new File("Platus pavyzdys.txt");
                PrintWriter out = new PrintWriter(new FileWriter(file));

                int[][] ge= new int[999999][rowNum];
                for (int j=0;j<chromsk;j++)
                for (int i=1;i<rowNum;i++)
                ge[j][i]=se[j][i];
                mtc=mutac;
                iteracijos =1+Iterac;
                int mm=geruChromNr+bloguChromNr;
                MaxKryzmNr=0;


//                  VisosGamybosLaikasVisos = new int[iteracijos];
```

```
//                  GamybosPrastovosVisos = new int[iteracijos];


            for (int i=mm-1;i>0;i--)
                        MaxKryzmNr=MaxKryzmNr+i;

            int[] sitas =new int[rowNum];
            int sitasChNr;

            int[] sitasPrast =new int[rowNum];
            int[] sitasTrukm =new int[rowNum];

            int[] Atrinkti = new int[MaxKryzmNr+1];
            int[] geri = new int[geruChromNr];
            int[] koefg= new int[geruChromNr];
            int[] blogi= new int[bloguChromNr];
            int[] koefb= new int[bloguChromNr];
            int Totalmin =1000000;
            int[] MinEil = new int[rowNum];
            double Vidurkis =0;

            int[] Pras = new int [MaxKryzmNr+chromsk];
            double[] Prastov=new double [MaxKryzmNr+chromsk];

            int[] GamPras = new int [MaxKryzmNr+chromsk];
            int[] GamTruk = new int [MaxKryzmNr+chromsk];


            double[] GamPrasDoub = new double [MaxKryzmNr+chromsk];
            double[] GamTrukDoub = new double [MaxKryzmNr+chromsk];


            for(int i=0;i<chromsk;i++) {


// cia bandymas

                        String Prastovos=Gaminam(ge[i],0,0,0);
            Pras[i]=Integer.valueOf(Prastovos);
            Prastov[i] = Double.valueOf(Prastovos);


                        String GamPrastovos=Gaminam(ge[i],1,0,0);
            GamPras[i]=Integer.valueOf(GamPrastovos);
            GamPrasDoub[i]=Integer.valueOf(GamPrastovos);

                        String GamTrukmes=Gaminam(ge[i],2,0,0);
            GamTruk[i]=Integer.valueOf(GamTrukmes);
            GamTrukDoub[i]=Integer.valueOf(GamTrukmes);

            }
            //
            int MinChrom=MinChrom(Pras);
            int MinChromPrast=MinChrom(GamPras);
            int MinChromTrumk=MinChrom(GamTruk);



            int aa1 = MinIkiInd(Pras,0,chromsk);


            out.println();
```

```
            out.println("         Sugeneruotu chromosomu, ju eiliskumo ir
prastovu(min) sarasas: ");
            out.println();
            for (int j=0;j<chromsk;j++){
                        out.print("Chromosoma "+j+":     ");
                        for (int i=1;i<rowNum;i++){
                                    out.print(ge[j][i]+" ");
                                    }
                        out.println("      Prastovos:  "+Pras[j]);
            }
            out.println();
            out.println("         Maziausios prastovos: "+MinChrom+"
            Chromosomos nr: "+MinIkiInd(Pras,0,chromsk));

            sitas[0]=MinChrom;
            sitasPrast[0]=MinChromPrast;
            sitasTrukm[0]=MinChromTrumk;

            genvid=MinChrom;

            GamybosPrastovos[0]=MinChrom(GamPras);
            //VisosGamybosLaikas[0]=MinIkiInd(GamPras,0,chromsk);
            VisosGamybosLaikas[0]=GamTruk[MinIkiInd(GamPras,0,chromsk)];

            VisosGamybosLaikas[1]=MinChrom(GamTruk);
            //GamybosPrastovos[1]=MinIkiInd(GamTruk,0,chromsk);
            GamybosPrastovos[1]=GamPras[MinIkiInd(GamTruk,0,chromsk)];


            //System.out.println(GamybosPrastovos); // Prastovos
            //System.out.println(VisosGamybosLaikas); // Gamybos Trukme




            out.print("           Uzsakymu gamybos eiliskumas: ");
            for(int i=1;i<rowNum;i++){
                        sitas[i]=ge[MinIkiInd(Pras,0,chromsk)][i];

                        sitasPrast[i]=ge[MinIkiInd(GamPras,0,chromsk)][i];
                        sitasTrukm[i]=ge[MinIkiInd(GamTruk,0,chromsk)][i];

            out.print(ge[MinIkiInd(Pras,0,chromsk)][i]+" ");}
            out.println();

            out.print("           Bendras prastovu vidurkis: ");
            Vidurkis=VidurkisDouble(Prastov,chromsk);
            out.printf("%.2f",Vidurkis);
            out.println();
            out.println();
            out.println("||||||||||||||||||||||||||||||||||||||||||||||||||||||||||
||||||||||||||||||||\n");
            //System.out.println();
            //                                                          PRASIDEDA
CIKLAS

            for ( int gais = 1; gais < iteracijos; gais++)
            {
            out.println();
            out.println("                                    ITERACIJOS NR: "+gais);
            out.println();
            out.println("                           Geru chromosomu:
"+geruChromNr+"            Gaminiu eiliskumas");
            int bla=0;
```

```
            for (int j=0;j<geruChromNr;j++){
                                    koefg[j]=MinIkiInd(Pras,bla,MaxKryzmNr);
                                    geri[j]=Pras[koefg[j]];



                                    out.print("Prastova "+geri[j]+"
"+"Chromosomos indeksas "+koefg[j]+"                    ");
                                    bla=Pras[koefg[j]];

                                    for(int i=1;i<rowNum;i++){
                                    out.print(ge[koefg[j]][i]+" ");
                                    }
                                    out.println();
                        }

            out.println();
            out.println("                        Blogu chromosomu:
"+bloguChromNr);
            int bla2=10000;
            for (int j=0;j<bloguChromNr;j++){
                        koefb[j]=MaxIkiInd(Pras,bla2,MaxKryzmNr);
                        blogi[j]=Pras[koefb[j]];
                        out.print("Prastova "+blogi[j]+" "+"Chromosomos indeksas
"+koefb[j]+"                        ");
                        bla2=Pras[koefb[j]];

                        for(int i=1;i<rowNum;i++)
                        out.print(ge[koefb[j]][i]+" ");
                        out.println();
            }



            for(int i=0;i<geruChromNr;i++)
                        Atrinkti[i]=koefg[i];
            for(int i=0;i<bloguChromNr;i++)
                        Atrinkti[geruChromNr+i]=koefb[i];




            out.println();
            out.println("-----------------------------------------------------------
---------------------\n");

int[][] PirmasVaikas = new int[999999][rowNum];
int NaujaPop=0;




//   -------------   ATKOMENTUOTI KAI BUS DAROMI NORMALUS
SKAICIAVIMAI!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!!!!!!!!!




for(int i1=0;i1<mm-1;i1++)
{
```

```
            for(int j1=i1+1;j1<mm;j1++)
            {
                        int[] Laikinas = new int[rowNum+1];

                        for(int k1=1;k1<KryzminimoTaskas;k1++)
                        {
            PirmasVaikas[NaujaPop][k1]=ge[Atrinkti[i1]][k1];      }

                        int a1=KryzminimoTaskas;
                        for(int VK=KryzminimoTaskas;VK<rowNum;VK++) {
                        Laikinas[VK-a1]=ge[Atrinkti[i1]][VK];      }
                        int a=0;
                        for(int v=1;v<rowNum;v++) {
                                    for(int opa=0;opa<rowNum-
KryzminimoTaskas;opa++){
                                                if(ge[Atrinkti[j1]][v]==0)
                                                            break;
                                    if(ge[Atrinkti[j1]][v]==Laikinas[opa]){

            PirmasVaikas[NaujaPop][KryzminimoTaskas+a]=Laikinas[opa];
                                    Laikinas[opa]=0;
                                                a=a+1;
                                                            }          }          }
                        NaujaPop=NaujaPop+1;
            }
}

out.println();



out.println("         Po kryzminimo sugeneruotu chromosomu skaicius:
"+MaxKryzmNr);
for(int j=0;j<MaxKryzmNr+1;j++){
            out.println();
            if(PirmasVaikas[j][1]==0 && PirmasVaikas[j][2]==0 &&
PirmasVaikas[j][3]==0)
                        break;

            for(int i=1;i<rowNum;i++)
                        out.print(PirmasVaikas[j][i]+" ");
            }




out.println();
out.println();

out.println("-----------------------------------------------------------------
----------");
out.println();
out.println("         Pasirinktos mutacijos tikimybe: "+MutavProc+" %");
out.println();

// NAUJAS VAIKAS //


outerloop:
for(int j=0;j<MaxKryzmNr+1;j++){
            if(PirmasVaikas[j][1]==0 && PirmasVaikas[j][2]==0 &&
PirmasVaikas[j][3]==0)
                                    break outerloop;
```

```
            int a=randInt(1,100);

            if (MutavProc!=0)
            if(a<=MutavProc)
            { out.print("Mutavo");
                        Mutacijos(PirmasVaikas[j],mutac);
}
            out.print("          ");
            for(int i=1;i<rowNum;i++){
                        ge[j][i]=PirmasVaikas[j][i];
                        out.print(ge[j][i]+" ");
            }


                        String Prastovos=Gaminam(PirmasVaikas[j],0,0,0);

                        Pras[j]=Integer.valueOf(Prastovos);
                        Prastov[j] = Double.valueOf(Prastovos);


                        String GamPrastovos=Gaminam(PirmasVaikas[j],1,0,0);
                        GamPras[j]=Integer.valueOf(GamPrastovos);
                        GamPrasDoub[j]=Integer.valueOf(GamPrastovos);

                        String GamTrukmes=Gaminam(PirmasVaikas[j],2,0,0);
                        GamTruk[j]=Integer.valueOf(GamTrukmes);
                        GamTrukDoub[j]=Integer.valueOf(GamTrukmes);


out.println("         Prastovos:     "+Pras[j]);
}



//aa

int MinPrast=Min(Pras);

//int MinFiksPrast=Min(GamPras);
//int MinFiksTrukm=Min(GamTruk);


int Ieskomas = MinIkiInd(Pras,0,MaxKryzmNr+1);

//System.out.println(MinPrast);
//System.out.println(" Turimas " + Pras[MinIkiInd(Pras,0,MaxKryzmNr+1)]);



//System.out.println("          Iteracijos Nr:     "+gais);
// Cia ieskau minimalias prastovas (arba gamybos trukme - pagal tikslo
funkcija)


// CIA TURI BUTI LYGINIMAS SU ISORINEMIS, VISOMIS PRASTOVOMIS, IMAMAS
ITERACIJOS NUMERIS


if(sitas[0]>MinPrast)
{           sitas[0]=MinPrast;
            sitasChNr=Ieskomas;
//          System.out.println(" Atnaujintas " + Pras[sitasChNr]);
```

```
// Saugau chromosoma, kuri duoda minimalias prastovas (arba gamybos trukme -
pagal tikslo funkcija)
           //System.out.println("          Prastovos:     "+sitas[0]);

           for(int i=1;i<rowNum;i++)
           {
//                    sitas[i]
=PirmasVaikas[MinIkiInd(Pras,0,MaxKryzmNr)][i];
//
           sitasPrast[i]=PirmasVaikas[MinIkiInd(GamPras,0,MaxKryzmNr)][i];
//
           sitasTrukm[i]=PirmasVaikas[MinIkiInd(GamTruk,0,MaxKryzmNr)][i];

                     sitas[i]    =PirmasVaikas[sitasChNr][i];
                     sitasPrast[i]=PirmasVaikas[sitasChNr][i];
                     sitasTrukm[i]=PirmasVaikas[sitasChNr][i];
           }



           Totalm[1]=gais;


}
else
{
           // ATRODO, KAD SITOJE VIETOJE YRA SAUGOMA GERIAUSIA CHROMOSOMA ANT
NAUJOJO SARASO!!!
           if(sitas[0]<MinPrast){
                     for(int i=1;i<rowNum;i++){
                     PirmasVaikas[0][i]=sitas[i];
                     ge[0][i]=sitas[i];
                     }

           Pras[0]=sitas[0];
           //System.out.println("          Prastovos:     "+sitas[0]);
           }


}

//System.out.println("Iteracija "+gais+"   Gamybos prastovos
"+GamybosPrastovosVisos[gais]+" Gamybos trukme
"+VisosGamybosLaikasVisos[gais]);




Totalm[0]=Totalmin=sitas[0];

//Totalm[0]=Totalmin=MinPrast;

for(int i=1;i<rowNum;i++){
           //MinEil[i]=ge[MinIkiInd(Pras,0,MaxKryzmNr)][i];  // TIKRINAME KAD
BUTENT KEICIA SITAS PRISKYRIMAS
           MinEil[i]=sitas[i];
//         BestEil[i]=ge[0][i]=MinEil[i];
           BestEil[i]=MinEil[i];
}

GamybosPrastovos[2]=Integer.valueOf(Gaminam(BestEil,1,0,0));
VisosGamybosLaikas[2]=Integer.valueOf(Gaminam(BestEil,2,0,0));
```

```
//Totalm[1]=IteracNr*Zingsnis+gais;

laikas=BestCr[0][0] = (System.currentTimeMillis() - start)/1000;




out.println();
out.println("         Maziausios prastovos: "+sitas[0]);

// Siek tiek keiciu pacia salyga
//System.out.println("          Prastovos:     "+sitas[0]);


//MIN1[gais-1]=BestCr[gais][1]=MinPrast;
out.print("           Bendras prastovu vidurkis: ");


MIN1[gais-1]=BestCr[gais][1]=sitas[0];
VID1[gais-1]=BestCr[gais][2]=Vidurkis=VidurkisDouble(Prastov,MaxKryzmNr);
MAX1[gais-1]=BestCr[gais][7]=MAXDouble(Prastov,MaxKryzmNr);
//VisuIteracijuChromosomos[gais-1]=BestEil;

//System.out.println(Vidurkis

//VID1Prast[gais-1]=BestCr[gais][4]=VidurkisDouble(GamPrasDoub,MaxKryzmNr);
//VID1Trukm[gais-1]=BestCr[gais][6]=VidurkisDouble(GamTrukDoub,MaxKryzmNr);



out.printf("%.2f",Vidurkis);
out.println();
out.println();
out.println("=================================================================
==========");



if(Totalmin>MinPrast) {
           Totalm[0]=Totalmin=MinPrast;

           for(int i=1;i<rowNum;i++){
                     MinEil[i]=ge[MinIkiInd(Pras,0,MaxKryzmNr)][i];  //
TIKRINAME KAD BUTENT KEICIA SITAS PRISKYRIMAS


//                   MinEil[i]=ge[0][i];
                     BestEil[i]=ge[0][i]=MinEil[i];
//                   BestEil[i]=MinEil[i];
           }

           GamybosPrastovos[2]=Integer.valueOf(Gaminam(BestEil,1,0,0));
           VisosGamybosLaikas[2]=Integer.valueOf(Gaminam(BestEil,2,0,0));
           Totalm[1]=gais;
           laikas=BestCr[0][0] = (System.currentTimeMillis() - start)/1000;


}

           }
           out.println();
           out.println(" Is visu genetinio algoritmo iteraciju maziausios
prastovos: "+Totalmin+",");
```

```
//          System.out.println(" Is visu genetinio algoritmo iteraciju
maziausios prastovos: "+Totalmin+",");
            out.print("          gaminiu isdestymo eiliskumas:  ");
            for(int i=1;i<rowNum;i++)
            out.print(MinEil[i]+" ");
            out.println("");

            out.close();

            }

            public static void GenetinisDuom(int[][] se, int geruChromNr, int
bloguChromNr,int mutac, int MutavProc, int Iterac) throws IOException
            {

                        double start = System.currentTimeMillis();

                        int[][] ge= new int[999999][rowNum];
                        for (int j=0;j<chromsk;j++)
                        for (int i=1;i<rowNum;i++)
                        ge[j][i]=se[j][i];
                        iteracijos =1+Iterac;
                        int mm=geruChromNr+bloguChromNr;
                        MaxKryzmNr=0;


                        for (int i=mm-1;i>0;i--)
                        MaxKryzmNr=MaxKryzmNr+i;


                        int[] sitas=new int[rowNum];

            int[] Atrinkti = new int[MaxKryzmNr+1];
            int[] geri = new int[geruChromNr];
            int[] koefg= new int[geruChromNr];
            int[] blogi= new int[bloguChromNr];
            int[] koefb= new int[bloguChromNr];

            int sitasChNr;

            int Totalmin =1000000;
            int[] MinEil = new int[rowNum];
            double Vidurkis =0;
            double[] Vidurk = new double[iteracijos+1];

            int[] Pras = new int [MaxKryzmNr+chromsk];
            double[] Prastov=new double [MaxKryzmNr+chromsk];

            for(int i=0;i<chromsk;i++) {

                        String Prastovos=Gaminam(ge[i],0,0,0);
//                      System.out.println(Prastovos);
                        Pras[i]=Integer.valueOf(Prastovos);
                        Prastov[i] = Double.valueOf(Prastovos);

            }
//          System.out.println();

                        //Vidurkis=VidurkisDouble(Prastov,chromsk);

                        sitas[0]=MinChrom(Pras);

                        for(int i=1;i<rowNum;i++)
                        sitas[i]=ge[MinIkiInd(Pras,0,chromsk)][i];
```

```
            //                    PRASIDEDA CIKLAS

            for ( int gais = 1; gais < iteracijos; gais++)
            {
                        int bla=0;
            for (int j=0;j<geruChromNr;j++){
                                    koefg[j]=MinIkiInd(Pras,bla,MaxKryzmNr);
                                    geri[j]=Pras[koefg[j]];
                                    bla=Pras[koefg[j]];
                                                }

            int bla2=10000;
            for (int j=0;j<bloguChromNr;j++){
                        koefb[j]=MaxIkiInd(Pras,bla2,MaxKryzmNr);
                        blogi[j]=Pras[koefb[j]];
                        bla2=Pras[koefb[j]];

                        }

            for(int i=0;i<geruChromNr;i++)
                        Atrinkti[i]=koefg[i];
            for(int i=0;i<bloguChromNr;i++)
                        Atrinkti[geruChromNr+i]=koefb[i];

int[][] PirmasVaikas = new int[999999][rowNum];
int NaujaPop=0;

for(int i1=0;i1<mm-1;i1++)
{
            for(int j1=i1+1;j1<mm;j1++)
            {
                        int[] Laikinas = new int[rowNum+1];

                        for(int k1=1;k1<KryzminimoTaskas;k1++)
                        {
            PirmasVaikas[NaujaPop][k1]=ge[Atrinkti[i1]][k1];      }
                        int a1=KryzminimoTaskas;
                        for(int VK=KryzminimoTaskas;VK<rowNum;VK++) {
                        Laikinas[VK-a1]=ge[Atrinkti[i1]][VK];
                        }

                        int a=0;
                        for(int v=1;v<rowNum;v++) {
                                    for(int opa=0;opa<rowNum-
KryzminimoTaskas;opa++){
                                                if(ge[Atrinkti[j1]][v]==0)
                                                            break;
                                    if(ge[Atrinkti[j1]][v]==Laikinas[opa]){

            PirmasVaikas[NaujaPop][KryzminimoTaskas+a]=Laikinas[opa];
                                    Laikinas[opa]=0;
                                                a=a+1;
                                                }          }          }
                        NaujaPop=NaujaPop+1;
            }          }


for(int j=MaxKryzmNr;j>0;j--)
            for(int i=1;i<rowNum;i++)
                        PirmasVaikas[j][i]=PirmasVaikas[j-1][i];

for(int i=0;i<rowNum;i++)
```

```
            PirmasVaikas[0][i]=sitas[i];



// NAUJAS VAIKAS
outerloop:
for(int j=0;j<MaxKryzmNr;j++){
            if(PirmasVaikas[j][1]==0 && PirmasVaikas[j][2]==0 &&
PirmasVaikas[j][3]==0)
                                    break outerloop;

            int a=randInt(1,100);

            if (MutavProc!=0)
            if(a<=MutavProc)
            Mutacijos(PirmasVaikas[j],mutac);

            for(int i=1;i<rowNum;i++)
                        ge[j][i]=PirmasVaikas[j][i];


            String Prastovos=Gaminam(PirmasVaikas[j],0,0,0);

            Pras[j]=Integer.valueOf(Prastovos);
            Prastov[j] = Double.valueOf(Prastovos);

}

int MinPrast=Min(Pras);


int Ieskomas = MinIkiInd(Pras,0,MaxKryzmNr+1);




if(sitas[0]>MinPrast){
            sitas[0]=MinPrast;
            sitasChNr=Ieskomas;

for(int i=1;i<rowNum;i++){
            sitas[i]=PirmasVaikas[sitasChNr][i];
            }
}
else
{
            if(sitas[0]<MinPrast){
                        for(int i=1;i<rowNum;i++){
                        PirmasVaikas[0][i]=sitas[i];
                        ge[0][i]=sitas[i];
                        }

            Pras[0]=sitas[0];
            }
}

// Cia keiciu visiems kitiems grafikams, kad butu vaizduojami gerai


MIN1[gais-1]=BestCr[gais][1]=sitas[0];
VID1[gais-1]=BestCr[gais][2]=Vidurkis=VidurkisDouble(Prastov,MaxKryzmNr);
MAX1[gais-1]=BestCr[gais][7]=MAXDouble(Prastov,MaxKryzmNr);
//VisuIteracijuChromosomos[gais-1]=BestEil;
```

```
//BestCr[gais][1]=sitas[0];
////BestCr[gais][1]=MinPrast;
//
//BestCr[gais][2]/*=Vidurkis*/=VidurkisDouble(Prastov,MaxKryzmNr);
//
////MAX1[gais-1]=BestCr[gais][7]=MAXDouble(Prastov,MaxKryzmNr);
//BestCr[gais][7]=MAXDouble(Prastov,MaxKryzmNr);

if(Totalmin>MinPrast) {
           BestCr[0][0]=Totalmin=MinPrast;

           for(int i=1;i<rowNum;i++){
           MinEil[i]=ge[MinIkiInd(Pras,0,MaxKryzmNr)][i];
           BestCr[0][i]=ge[0][i]=MinEil[i];
           }
           Best=gais;
           TT[vnt] = (System.currentTimeMillis() - start)/1000;
           }
           //Vidurk[gais]=Vidurkis;

           }
           }



           public static void MutacLygin(int[][] ge) throws Exception
           {
                                GASXChart re = new GASXChart();
                     for(int i=1;i<6;i++) {
                     //         double start = System.currentTimeMillis();
                     vnt=MutacRus=i;

                     try {
                                if(mtc!=MutacRus)



                                                     GenetinisDuom(ge, gerC, bloC,
MutacRus, MutacProc, IteracNr);
                                                     //Genetinis(ge, gerC, bloC,
MutacRus, MutacProc, IteracNr);




                                } catch (IOException e) {
                                // TODO Auto-generated catch block
                                e.printStackTrace();
                     }

                     re.NaujaClass();


           }

           }


           public static void main(String[] args)  throws Exception {
```

```
                    File excel = new File(Failas());

                    //   --- REALIAI TAI TAS PATS ---
                    //File excel = new
File("C:/Users/edvinas.duoba/Desktop/eclipse/Duomenys.xls");
                    FileInputStream fis = new FileInputStream(excel);


                    // IS DARBO
                    //FileInputStream fis = new
FileInputStream("C:/Users/edvinas.duoba/Desktop/eclipse/Duomenys.xls");
                    // IS NAMU
                    //FileInputStream fis = new
FileInputStream("C:/Users/Edvinas/Desktop/eclipse/Duomenys.xls");



                    HSSFWorkbook wb = new HSSFWorkbook(fis);

                    HSSFSheet ws1 = wb.getSheet("Uzsakymai");
                    HSSFSheet ws2 = wb.getSheet("Technologija");

                    HSSFSheet ws3 = wb.getSheet("OperMedis");
                    HSSFSheet ws4 = wb.getSheet("OperMetalas");
                    HSSFSheet ws5 = wb.getSheet("OperAbu");
                    HSSFSheet ws7 = wb.getSheet("Stakles");

                    rowNum = ws1.getLastRowNum() + 1;
                    colNum = ws1.getRow(0).getLastCellNum();

                    rowNum1 =ws2.getLastRowNum();
                    colNum1 = ws2.getRow(0).getLastCellNum();

                    S1dataT = new String[rowNum][colNum];  // Tekstu
nuskaitymui
                    S1dataN = new Double[rowNum][colNum];  // Skaiciu
nuskaitymui

                    S2dataT = new String[rowNum1][colNum1];
                    S2dataN = new Double[rowNum1][colNum1];


                    // kadangi kiekvienu stakliu kiekis bus kitoks, taigi
reikes atskirai apsirasyti kiekvienu stakliu kiekius
                    // [sheet3 irasu sk] [kiek tos pacios rusies stakliu]
[0,1,2 - pradzia, pabaiga, trukme] [kiek gmainiu gaminama - turi keistis ne
pagal cikla, bet rankiniu budu kiekvienas]



          int n=115;
          da = new int[rowNum];
          ch=new int[n][rowNum];
          for (int i=1; i<da.length;i++)
                    ch[0][i]=da[i]=i;

          // NUSKAITOMAS SHEET1
          for ( int i = 1; i<rowNum; i++)
          {  HSSFRow row = ws1.getRow(i);
                              for ( int j = 0; j < colNum; j++)
                              {  HSSFCell cell = row.getCell(j);
                                        if (
cell.getCellType()==cell.CELL_TYPE_NUMERIC )
```

```
S1dataN[ch[0][i]][j]=cell.getNumericCellValue();
                                        else

           S1dataT[ch[0][i]][j]=cell.getStringCellValue();  }

                                S1dataT[ch[0][i]][0] += indent.substring(0,
indent.length() - S1dataT[ch[0][i]][0].length());
                                S1dataT[ch[0][i]][0] =
S1dataT[ch[0][i]][0].substring(0,75);
                                S1dataT[ch[0][i]][1] += indent.substring(0,
indent.length() - S1dataT[ch[0][i]][1].length());
                                S1dataT[ch[0][i]][1] =
S1dataT[ch[0][i]][1].substring(0,28);
           }

           //NUSKAITOMAS SHEET2
           for ( int i = 1; i<rowNum1; i++)
           {  HSSFRow row = ws2.getRow(i);

           //System.out.println(row.getCell(0)+"      "+row.getCell(1)+"
           "+row.getCell(2));
                                for ( int j = 0; j < colNum1; j++)
                                {  HSSFCell cell = row.getCell(j);
                                //if ( i>85)
                                //         System.out.println(cell);

                                 if (
cell.getCellType()==cell.CELL_TYPE_NUMERIC )

S2dataN[i][j]=cell.getNumericCellValue();
                                 else

           S2dataT[i][j]=cell.getStringCellValue();  }

                                S2dataT[i][0] += indent.substring(0,
indent.length() - S2dataT[i][0].length());
                                S2dataT[i][0] =
S2dataT[i][0].substring(0,28);
                                //S2dataT[i][1] += indent.substring(0,
indent.length() - S2dataT[i][1].length());
                                //S2dataT[i][1] =
S2dataT[i][1].substring(0,15);
                                }


           n2 = ws3.getLastRowNum()+1;
           n22 = ws4.getLastRowNum()+1;
           n23 = ws5.getLastRowNum()+1;



           StaklesMedis = new int[n2];
           StaklesMetalas = new int[n22];
           StaklesAbu = new int[n23];



           //String[][] S3data=new String[n2][none];
           String[] S3data=new String[n2];

           OOMedis = new String[n2];
```

```
                    for ( int i = 1; i<n2; i++)
                    {  HSSFRow row = ws3.getRow(i);
                                        //for ( int j = 0; j < none; j++)
                                        //{
                                                HSSFCell cell =
row.getCell(0);

          S3data[i]=cell.getStringCellValue();
                                                //}
                                        //S3data[i] += indent.substring(0,
indent.length() - S3data[i].length());
                                        //S3data[i] =
S3data[i].substring(0,15);
                                        OOMedis[i]=S3data[i]; }


                    String[] S4data=new String[n22];

                    OOMetalas = new String[n22];


                              for ( int i = 1; i<n22; i++)
                              {  HSSFRow row = ws4.getRow(i);
                                                //for ( int j = 0; j <
none; j++)
                                                //{
                                                          HSSFCell cell
= row.getCell(0);

          S4data[i]=cell.getStringCellValue();
                                                          //}
                                                //S3data[i] +=
indent.substring(0, indent.length() - S3data[i].length());
                                                //S3data[i] =
S3data[i].substring(0,15);
                                                OOMetalas[i]=S4data[i];
          }



                              String[] S5data=new String[n23];

                              OOAbu = new String[n23];


                                        for ( int i = 1; i<n23; i++)
                                        {  HSSFRow row = ws5.getRow(i);
                                                          //for ( int j
= 0; j < none; j++)
                                                          //{

          HSSFCell cell = row.getCell(0);

          S5data[i]=cell.getStringCellValue();

          //}
                                                          //S3data[i]
+= indent.substring(0, indent.length() - S3data[i].length());
                                                          //S3data[i] =
S3data[i].substring(0,15);

          OOAbu[i]=S5data[i];   }
```

```
                    // Nuskaitau visas esamas stakles

                    n3 = ws7.getLastRowNum()+1;
                    String[] S7data=new String[n3];
                    OOO = new String[n3];

                    for ( int i = 1; i<n3; i++)
                    {  HSSFRow row = ws7.getRow(i);
                                        //for ( int j = 0; j < none; j++)
                                        //{
                                                HSSFCell cell =
row.getCell(0);

          S7data[i]=cell.getStringCellValue();
                                                //}
                                                //S7data[i] +=
indent.substring(0, indent.length() - S7data[i].length());
                                                //S7data[i] =
S7data[i].substring(0,15);
                                                OOO[i]=S7data[i];

                                                for ( int k = 1; k<n2;
k++)
                                                {

          //System.out.println(S7data[i] + " "+ S7data[i].toLowerCase()+"  su
"+OO[k]+" "+OO[k].toLowerCase());
                                                                if
(S7data[i].toLowerCase().contains(OOMedis[k].toLowerCase()))
                                                                {

          StaklesMedis[k]=StaklesMedis[k]+1;

          //System.out.println(OO[k]+" "+StaklesMedis[k]);

          break;
                                                                }
                                                }

                                                for ( int k = 1; k<n22;
k++)
                                                {

          //System.out.println(S7data[i] + " "+ S7data[i].toLowerCase()+"  su
"+OO[k]+" "+OO[k].toLowerCase());
                                                                if
(S7data[i].toLowerCase().contains(OOMetalas[k].toLowerCase()))
                                                                {

          StaklesMetalas[k]=StaklesMetalas[k]+1;

          //System.out.println(OO[k]+" "+StaklesMedis[k]);

          break;
                                                                }
                                                }

                                                for ( int k = 1; k<n23;
k++)
                                                {

          //System.out.println(S7data[i] + " "+ S7data[i].toLowerCase()+"  su
"+OO[k]+" "+OO[k].toLowerCase());
```

```
                                                                    if
(S7data[i].toLowerCase().contains(OOAbu[k].toLowerCase()))
                                                                    {

            StaklesAbu[k]=StaklesAbu[k]+1;

            //System.out.println(OO[k]+" "+StaklesMedis[k]);

            break;
                                                                    }}
}

                        MaxStaklMedis = MAX(StaklesMedis,n2);
                        MaxStaklMetalas = MAX(StaklesMetalas,n22);
                        MaxStaklAbu = MAX(StaklesAbu,n23);


                        LaikaiMedis = new int[n2][MaxStaklMedis][3][rowNum];
                        LaikaiMetalas = new
int[n22][MaxStaklMetalas][3][rowNum];
                        LaikaiAbu = new int[n23][MaxStaklAbu][3][rowNum];

                        LaikaiTikrasMedis = new
int[n2][MaxStaklMedis][3][rowNum];
                        LaikaiTikrasMetalas = new
int[n22][MaxStaklMetalas][3][rowNum];
                        LaikaiTikrasAbu = new int[n23][MaxStaklAbu][3][rowNum];


                        IrasaiMedis = new int[n2][MaxStaklMedis];
                        IrasaiMetalas = new int[n2][MaxStaklMetalas];
                        IrasaiAbu = new int[n2][MaxStaklAbu];

                        IrasaiTikrasMedis = new int[n2][MaxStaklMedis];
                        IrasaiTikrasMetalas = new int[n2][MaxStaklMetalas];
                        IrasaiTikrasAbu = new int[n2][MaxStaklAbu];


                        //for ( int i = 1; i<n2; i++)
                        //         System.out.println(OO[i]+"
"+StaklesMedis[i]);
                        //



            //OO2= new int[n2][rowNum][4];
            OO2= new int[n3][rowNum][4];


            OOO2= new int[n3][rowNum][4];

                        EventQueue.invokeLater(new Runnable() {
                                    public void run() {
                                                try {
                                                            GANau frame = new
GANau();
                                                            frame.setVisible(true);
                                                } catch (Exception e) {
                                                            e.printStackTrace();
                                                }
                                    }});}

            public GANau() throws IOException {
```

```
                    setTitle("Gantt programa");
                    setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
                    setBounds(100, 60, 1300, 900);

                    JScrollPane scrollPane = new JScrollPane();

          scrollPane.setHorizontalScrollBarPolicy(ScrollPaneConstants.HORIZON
TAL_SCROLLBAR_ALWAYS);

          scrollPane.setVerticalScrollBarPolicy(ScrollPaneConstants.VERTICAL_
SCROLLBAR_ALWAYS);
                    getContentPane().add(scrollPane, BorderLayout.CENTER);

                    final JPanel panel = new JPanel();
                    panel.setPreferredSize(new Dimension(2200, 2200));
                    panel.setSize(new Dimension(1000, 1000));
                    panel.setAlignmentY(Component.TOP_ALIGNMENT);
                    panel.setAlignmentX(Component.LEFT_ALIGNMENT);
                    scrollPane.setViewportView(panel);
                    panel.setLayout(null);

                    final JLabel label = new JLabel(".");
                    final JLabel lblNewLabel = new JLabel();
                    final JLabel lblNewLabel1 = new JLabel();
                    final JLabel lblNewLabel2 = new JLabel();
                    label.setBounds(272, 309, 1000, 2512);
                    lblNewLabel.setBounds(139, 263, 100, 20);
                    lblNewLabel1.setBounds(51, 62, 250, 20);
                    lblNewLabel2.setBounds(350, 62, 460, 20);
                    lblNewLabel1.setText("\u012Eveskite populiacijos
dyd\u012F");
                    lblNewLabel2.setText("Pasirinkite norim\u0105
atvaizduoti atsitiktinai sugeneruot\u0105 chromosom\u0105");

                    final TextField textField_1 = new TextField();
                    final TextField textField_2 = new TextField();
                    final TextField textField_3 = new TextField();
                    final TextField textField_4 = new TextField();
                    final TextField textField_5 = new TextField();

                    final JButton btnAisk = new JButton("?");
                    btnAisk.setBounds(225, 263, 44, 20);
                    panel.add(btnAisk);


                    final JButton btnAisk2 = new JButton("?");
                    btnAisk2.setBounds(245, 212, 44, 20);
                    panel.add(btnAisk2);

                    btnAisk2.addActionListener(new ActionListener() {
                              public void actionPerformed(ActionEvent e) {

          JOptionPane.showMessageDialog(panel.getComponent(0)," Elementai -
duomen\u0173 failo \u012Fra\u0161ai (u\u017Esakymai).\n"
                                                                        +
"Kry\u017Eminimai gali b\u016Bti daromi nuo 1 ir iki prie\u0161paskutinio
elemento. \n"
                                                                        +
"Pavyzd\u017Eiui jeigu turime 12 element\u0173, tai galima kry\u017Eminti nuo
1 elemento iki 10.\n"
                                                                        +
"Kry\u017Eminant po 11 elemento, gausime tas pa\u010Dias chromosomas.\n",
                                                                          "Elementai
", JOptionPane.INFORMATION_MESSAGE );
```

```
                                    }});

                        btnAisk.addActionListener(new ActionListener() {
                                    public void actionPerformed(ActionEvent e) {

            JOptionPane.showMessageDialog(panel.getComponent(0),

                                                                    "Mutacija
1: Sumai\u0161omi chromosomos elementai\n"
                                                                  + "Mutacija
2: Sukei\u010Diami pirmas ir paskutinis chromosomos elementai\n"
                                                                  + "Mutacija
3: Sukei\u010Diami antras ir prie\u0161 paskutinis chromosomos elementai\n"
                                                                  + "Mutacija
4: Pirmas elementas = prie\u0161paskutinis, prie\u0161paskutinis =
paskutinis,\n"
                                                                  + "paskutinis
= antras, antras = pirmas\n"
                                                                  + "Mutacija
5: Sumai\u0161omi pirmi devyni chromosomos elementai\n",
                                                                    "Mutacijos:
", JOptionPane.INFORMATION_MESSAGE );
                                    }});

                        //Sukuriu gaminiu Masyva reikalinga ComboBox
                                                final String[] Bo=new
String[rowNum];
                                                for (int i=0; i<Bo.length;i++)
                                                { Bo[i]=Gaminiai(S2dataT,i); }

                                                final JComboBox comboBox = new
JComboBox(Bo);
                                                comboBox.setBounds(780, 95, 170,
20);


                                                final String[] mut= new String[5];
// 1 tuscias ir 5 todel, kad as esu sukures 5 skirtingas mutacijas
                                                for(int i=0;i<5;i++)
                                                {
                                                            String ba="  ";

            ba=Integer.toString(i+1);
                                                            mut[i]=ba;
                                                }

                                                //Mutacju ComboBox
                                                final JComboBox comboBox_1 = new
JComboBox(mut);
                                                comboBox_1.setBounds(175, 263, 44,
20);
                                                panel.add(comboBox_1);




                                                String[] TiksloF_jos= new
String[2];

                                                TiksloF_jos[0] = " Minimizuojamos
prastovas ";
                                                TiksloF_jos[1] = " Minimizuojama
gamybos trukme ";
```

```
                                                final JComboBox comboBox_2 = new
JComboBox(TiksloF_jos);
                                                //comboBox_2.setBounds(175, 263,
44, 20);
                                                comboBox_2.setBounds(55, 360, 200,
20);
                                                panel.add(comboBox_2);


                        panel.add(label);
                        panel.add(lblNewLabel);
                        panel.add(lblNewLabel1);

                        textField_1.setBounds(65, 88, 72, 20);
                        panel.add(textField_1);
                        textField_2.setBounds(115, 165, 44, 20);
                        panel.add(textField_2);
                        textField_3.setBounds(239, 165, 44, 20);
                        panel.add(textField_3);
                        textField_4.setBounds(190, 237, 44, 20);
                        panel.add(textField_4);
                        textField_5.setBounds(92, 310, 44, 20);
                        panel.add(textField_5);


                        JLabel label5 = new JLabel("Pasirinkite
skai\u010Di\u0173, kiek bus naudojama");
                        label5.setBounds(51, 119, 285, 20);
                        panel.add(label5);

                        JLabel label6 = new JLabel("Geriausi\u0173");
                        label6.setBounds(51, 165, 54, 20);
                        panel.add(label6);

                        JLabel label7 = new JLabel("Blogiausi\u0173");
                        label7.setBounds(167, 165, 72, 20);
                        panel.add(label7);

                        JLabel label8 = new JLabel("Pasirinkite mutacij\u0105");
                        label8.setBounds(51, 263, 138, 20);
                        panel.add(label8);

                        JLabel label9 = new JLabel("Mutacijos tikimyb\u0117 %");
                        label9.setBounds(51, 237, 138, 20);
                        panel.add(label9);

                        JLabel label10 = new JLabel("\u012Eveskite genetinio
algoritmo atliekam\u0173 iteracij\u0173");
                        label10.setBounds(51, 285, 272, 20);
                        panel.add(label10);

                        JLabel label11 = new JLabel("kiek\u012F");
                        label11.setBounds(51, 310, 35, 20);
                        panel.add(label11);

                        JLabel label12 = new JLabel("Pasirinkite pagrindin\u0119
tikslo funkcij\u0105:");
                        label12.setBounds(51, 335, 300, 20);
                        panel.add(label12);


                        JLabel lblGeriausiuIrBlogiausiu = new
JLabel("geriausi\u0173 ir blogiausi\u0173 chromosom\u0173");
                        lblGeriausiuIrBlogiausiu.setBounds(51, 139, 285, 20);
```

```
                    panel.add(lblGeriausiuIrBlogiausiu);

                    final JLabel Label11 = new JLabel("Pasirinkite, kokius
optimalios chromosomos elementus norite matyti");
                    Label11.setBounds(350, 93, 430, 20);


                    final JLabel label2 = new JLabel();
                    label2.setBounds(350, 40, 750, 200);
                    panel.add(label2);

                    final JLabel lblNewLabel_1 = new JLabel();
                    lblNewLabel_1.setBounds(593, 62, 64, 20);
                    panel.add(lblNewLabel_1);

                    final JLabel label_1 = new JLabel();
                    label_1.setBounds(566, 93, 154, 20);
                    panel.add(label_1);

                    final JLabel lblNewLabel_2 = new JLabel();
                    lblNewLabel_2.setBounds(90, 430, 200, 20);
                    panel.add(lblNewLabel_2);

                    final JLabel lblNewLabel_3 = new JLabel();
                    lblNewLabel_3.setBounds(110, 460, 200, 20);
                    panel.add(lblNewLabel_3);

                    JLabel lblles = new JLabel("Pasirinkite, po kurio
elemento norite");
                    lblles.setBounds(51, 191, 227, 20);
                    panel.add(lblles);

                    JLabel lblNewLabel_4 = new JLabel("prad\u0117ti
kry\u017Eminim\u0105");
                    lblNewLabel_4.setBounds(51, 211, 129, 20);
                    panel.add(lblNewLabel_4);

                    final TextField textField_6 = new TextField();
                    textField_6.setBounds(191, 211, 44, 20);
                    panel.add(textField_6);

                    JLabel lblKk = new JLabel("U\u017Efiksuoti duomenis,
naudokite \"Enter\" klavi\u0161\u0105");
                    lblKk.setBounds(51, 29, 400, 20);
                    panel.add(lblKk);



                    textField_1.addKeyListener(new KeyAdapter() {
                              @Override
                              public void keyTyped(KeyEvent arg0) {
                                        char c=arg0.getKeyChar();
                                        if(!(Character.isDigit(c) ||
(c==KeyEvent.VK_BACK_SPACE) || (c==KeyEvent.VK_DELETE) ||
(c==KeyEvent.VK_ENTER))){
                                                  arg0.consume();

                                        }
                              }
                    });

                    textField_2.addKeyListener(new KeyAdapter() {
                              @Override
                              public void keyTyped(KeyEvent arg0) {
```

```
                                        char c=arg0.getKeyChar();
                                        if(!(Character.isDigit(c) ||
(c==KeyEvent.VK_BACK_SPACE) || (c==KeyEvent.VK_DELETE) ||
(c==KeyEvent.VK_ENTER))){
                                                arg0.consume();

                                        }
                                }
                        });

                        textField_3.addKeyListener(new KeyAdapter() {
                                @Override
                                public void keyTyped(KeyEvent arg0) {
                                        char c=arg0.getKeyChar();
                                        if(!(Character.isDigit(c) ||
(c==KeyEvent.VK_BACK_SPACE) || (c==KeyEvent.VK_DELETE) ||
(c==KeyEvent.VK_ENTER))){
                                                arg0.consume();

                                        }
                                }
                        });

                        textField_4.addKeyListener(new KeyAdapter() {
                                @Override
                                public void keyTyped(KeyEvent arg0) {
                                        char c=arg0.getKeyChar();
                                        if(!(Character.isDigit(c) ||
(c==KeyEvent.VK_BACK_SPACE) || (c==KeyEvent.VK_DELETE) ||
(c==KeyEvent.VK_ENTER))){
                                                arg0.consume();

                                        }
                                }
                        });

                        textField_5.addKeyListener(new KeyAdapter() {
                                @Override
                                public void keyTyped(KeyEvent arg0) {
                                        char c=arg0.getKeyChar();
                                        if(!(Character.isDigit(c) ||
(c==KeyEvent.VK_BACK_SPACE) || (c==KeyEvent.VK_DELETE) ||
(c==KeyEvent.VK_ENTER))){
                                                arg0.consume();

                                        }
                                }
                        });


                        textField_6.addKeyListener(new KeyAdapter() {
                                @Override
                                public void keyTyped(KeyEvent arg0) {
                                        char c=arg0.getKeyChar();
                                        if(!(Character.isDigit(c) ||
(c==KeyEvent.VK_BACK_SPACE) || (c==KeyEvent.VK_DELETE) ||
(c==KeyEvent.VK_ENTER))){
                                                arg0.consume();

                                        }
                                }
                        });
```

```
                    // Patikslinti, ka grafike rodo ne tikslo funkcijos
taskai, nes jeigu chromosoma nesikeicia - tikslo f-ja grazina
                    // toki pati rezultata, kodel keiciasi prastovos.

                    // Kaip analizuoti koreliacija? - pasikalbeti su
destytoju.




                    // Generuojamu chromosomu skaicius
                    textField_1.addActionListener(new ActionListener() {
                              public void actionPerformed(ActionEvent e) {
                                        final String input1 =
textField_1.getText();
                                        //chromsk = 20;
                                                  chromsk =
Integer.valueOf(input1);

          textField_1.enable(false);




                                                  // Geriausiu iteraciju
skaicius
                                        textField_2.addActionListener(new
ActionListener() {
                                                  public void
actionPerformed(ActionEvent e) {
                                                            final String
input2 = textField_2.getText();
                                                            //gerC = 6;
                                                            gerC =
Integer.valueOf(input2);

          textField_2.enable(false);


                                                                      //
BLogiausiu iteraciju skaicius

          textField_3.addActionListener(new ActionListener() {

          public void actionPerformed(ActionEvent e) {

          final String input3 = textField_3.getText();

          //bloC = 4;

          bloC = Integer.valueOf(input3);

          textField_3.enable(false);


                                                                      //
Kryzminimas

          textField_6.addActionListener(new ActionListener() {

          public void actionPerformed(ActionEvent e) {

          final String input6 = textField_6.getText();
```

```
            //KryzminimoTaskas = 1 + 2;
            KryzminimoTaskas = 1 + Integer.valueOf(input6);
            textField_6.enable(false);


            // Mutacijos tikimybe
            textField_4.addActionListener(new ActionListener() {
            public void actionPerformed(ActionEvent e) {
            final String input4 = textField_4.getText();
            //MutacProc = 30;
            MutacProc = Integer.valueOf(input4);
            lblNewLabel.setText("    ");
            textField_4.enable(false);

                        comboBox_1.addActionListener(new ActionListener() {
                                    public void actionPerformed(ActionEvent e3) {
                                                JComboBox ee1 =
(JComboBox)e3.getSource();
                                                String qw =
(String)ee1.getSelectedItem();
                                                //MutacRus = 2;
                                                MutacRus = Integer.valueOf(qw);
                                                comboBox_1.enable(false);



                                                // Genetinio algoritmo iteraciju
skaicius
                                                textField_5.addActionListener(new
ActionListener() {
                                                            public void
actionPerformed(ActionEvent e) {
                                                                        final String
input5 = textField_5.getText();
                                                                        //IteracNr =
5;
                                                                        IteracNr =
Integer.valueOf(input5);
            textField_5.enable(false);
            panel.add(Label11);
            panel.add(lblNewLabel2);


                                                 // Tikslo funkcijos pasirinkimas
                                                 comboBox_2.addActionListener(new
ActionListener() {
```

```
                                                    public void
actionPerformed(ActionEvent e5) {
                                                              JComboBox ee2
= (JComboBox)e5.getSource();
                                                              String TF =
(String)ee2.getSelectedItem();
                                                              //MutacRus =
2;
            //System.out.println(TF);

                                                              if
(TF.equals(TiksloF_jos[0]))
            TiksloFunkcija=1;

                                                              if
(TF.equals(TiksloF_jos[1]))
            TiksloFunkcija=2;

            //TiksloFunkcija = Integer.valueOf(TF);
            comboBox_2.enable(false);

                                              lblNewLabel_2.setText("Detalus
pavyzdys atspausdinamas ");
                                              lblNewLabel_3.setText(" faile
'Detalus pavyzdys.txt'");

                                              wee= new int[999999][rowNum];
                                              final int[][] we= new
int[999999][rowNum];




                                              BestEil = new int[rowNum];
                                              BestCr = new
double[IteracNr+1][rowNum+7];
                                              Totalm= new double[5];
                                              Totalm[1]=1;



                                              for (int j=0;j<chromsk;j++)
                                              {
                                                        MAISYMAS(da);
                                              for (int i=1;i<rowNum;i++)
                                              {         we[j][i]=da[i];
            //System.out.println(da[i]);
                                                        wee[j][i]=da[i];
                                              }
                                              }

                                              BestEil = new int[rowNum];
```

```
                                        BestCr = new
double[IteracNr+1][rowNum+7];
                                        //BestCrAbu = new
double[4][IteracNr+1][rowNum];
                                        MIN1= new double[IteracNr];
                                        VID1= new double[IteracNr];
                                        MAX1= new double[IteracNr];

                                        Totalm= new double[5];


                                        try {


//                                                  for ( int Zingsnis = 0;
Zingsnis < Riba; Zingsnis++)
//                                                  {
                                                    Genetinis(we, gerC,
bloC, MutacRus, MutacProc, IteracNr);

//                                                  }


                                        } catch (IOException e2) {
                                                    // TODO Auto-generated
catch block
                                                    e2.printStackTrace();
                                        }


          Totalm[1]=MinIkiIndDouble(MIN1,0,IteracNr)+1;


//
          BestEil=VisuIteracijuChromosomos[(int) Totalm[1]-1];




                                        /*
                                        for (int i =1;i<n2;i++)
                                        {
                                                    System.out.println("\n
Operacija: "+OO[i]+"  buvo atlikta: "+ IrasaiMedis[i]+" kartus \n");
                                                                for (int
j=0;j<IrasaiMedis[i];j++)
                                                    {

          System.out.println("  Pradzia = "+LaikaiMedis[i][0][0][j]+"
Pabaiga = "+LaikaiMedis[i][0][1][j]+"  Trukme = "+LaikaiMedis[i][0][2][j]);
                                                    }
                                        }

                                        */



                                        final int[][] we1 = (int[][])we;

                                        JComboBox comboBox1= new
JComboBox();

                                                                final
String[] Chr=new String[chromsk];
```

```
                                                                    for (int i=0;
i<chromsk;i++)
                                                                    {
            String as= "  ";
            as=Integer.toString(i);
            Chr[i]=as; }
                                                                    comboBox1 =
new JComboBox(Chr);
            comboBox1.setBounds(750, 62, 70, 20);
            panel.add(comboBox1);
            panel.add(comboBox);
            label2.setText("    ");
                                                                    //
Chromosomos pasirinkimas
            comboBox1.addActionListener(new ActionListener() {
            public void actionPerformed(ActionEvent e3) {
            JComboBox CB2 = (JComboBox)e3.getSource();
            String MSG3 = (String)CB2.getSelectedItem();
            final String MSG4 = (String)MSG3;

            final int MSG12 = Integer.valueOf(MSG4);
                                                                    MSGG = MSG12;
                                                                    GAGantt rw =
new GAGantt();
            rw.Newclass();

            }});
                                                                    final JButton
bttw = new JButton("Mutacij\u0173 lyginimas");
            panel.add(bttw);
            bttw.addActionListener(new ActionListener() {
            public void actionPerformed(ActionEvent e) {


            try {
                        double start1 = System.currentTimeMillis();
```

```
                MutacLygin(we);




        } catch (Exception e1) {
                // TODO Auto-generated catch block
                e1.printStackTrace();
        }

        bttw.setEnabled(false);
                        }});

                int Veliausias=Veliausias(BestEil);
                                                        final JButton
btw = new JButton("Grafikas");
        panel.add(btw);
        btw.addActionListener(new ActionListener() {
        public void actionPerformed(ActionEvent e) {

        bttw.setBounds(VK+149+Veliausias+150,570,140,20);
        GASXChart re = new GASXChart();
        try {
                re.NaujaClass();
        } catch (Exception e12) {
                // TODO Auto-generated catch block
                e12.printStackTrace();
        }
        btw.setEnabled(false);
        }});

                final JScrollPane scrollPane_1 = new JScrollPane();
                final JTextArea txtrS = new JTextArea();


                // Pasirinkimas gaminio arba visu gaminiu
                comboBox.addActionListener(new ActionListener() {
```

```
                                public void actionPerformed(ActionEvent e) {
                                        JComboBox cb =
(JComboBox)e.getSource();

                                // --- CIA SUFEIKINU KAD VISADA IMTU VISI
GAMINIAI REIKSME ---


                                                        //if (Gaminsim==1)
                                                        //msg = "Visi gaminiai
";
                                                        //final String msg1 =
"Visi gaminiai               ";
                                                        //else
                                                        // --- SITUS NAUDOTI,
KAI NORMALIAI JAU DARYSIU ---
                                                        String msg =
(String)cb.getSelectedItem();
                                                        //final String msg1 =
(String)msg;




                                                label_1.setText("    ");
                                                lblNewLabel_1.setText("      ");

                                                // Mygtukas "Grafikas"

          btw.setBounds(VK+desinen+149+Veliausias+20,570,90,20);

                                                if(sand==0)
                                                {
                                                        JLabel ButLab = new
JLabel();
                                                        int k2=0;
                                                        int k=0;
                                                        for(int i=1; i<n3;i++)
                                                        {
                                                                // Paslenka
zodzius
                                                        ButLab
=DefaultComponentFactory.getInstance().createLabel(OOO[i]);
                                                        ButLab.setBounds(VK+80,
VV+320+k, 80+desinen, 15);
                                                        panel.add(ButLab);

                                                                // Isilgines
linijos
                                                        JSeparator isilgai =
new JSeparator();

          isilgai.setBounds(VK+78, VV+317+k, 310-238+Veliausias+desinen, 2);
                                                        panel.add(isilgai);
                                                        k=k+20;
                                                        k2=i;
                                                        }
```

```
                                                    // Apatine grafiko
linija
                                                    JSeparator isilgai =
new JSeparator();
          isilgai.setBounds(VK+78, VV+317+k, 310-238+Veliausias+desinen, 2);
                                                    panel.add(isilgai);

                                                    // Vertikali linija is
desines zodziams
                                                    JSeparator separator_1
= new JSeparator();
          separator_1.setOrientation(SwingConstants.VERTICAL);
                                                    separator_1.setSize(new
Dimension(2, 25));
          separator_1.setBounds(VK+desinen+149, VV+317, 3, k);
                                                    panel.add(separator_1);

                                                    // Vertikali linija is
kaires zodziams
                                                    JSeparator separator_2
= new JSeparator();
          separator_2.setOrientation(SwingConstants.VERTICAL);
                                                    separator_2.setSize(new
Dimension(2, 25));
          separator_2.setBounds(VK+78, VV+317, 3, k);
                                                    panel.add(separator_2);

                                                    // Vertikali linija
desineje - paskutine
                                                    JSeparator separator_3
= new JSeparator();
          separator_3.setOrientation(SwingConstants.VERTICAL);
                                                    separator_3.setSize(new
Dimension(2, 25));
          separator_3.setBounds(VK+desinen+150+Veliausias, VV+317, 3, k);
                                                    panel.add(separator_3);

                                                    //         LAIKO JUOSTA

                                                    int Format=Format(k2);

                                                    JLabel laikas =
DefaultComponentFactory.getInstance().createLabel("Laikas, val");
          laikas.setBounds(VK+desinen+77, VV+144+Format, 310-238+Veliausias,
20);
                                                    panel.add(laikas);

                                                    int[] y = new int[24];
                                                    int gap = 0;
                                                    int gap1;
                                                    y[0]=-59;
                                                    for (int i=0;
i<Veliausias; i++)
                                                    {
                                                              gap1=0;
```

```
                                                                    for (int j=1;
j<4;j++)
                                                                    {
            gap1=gap1+15;
            //if(i+gap1>GANau.Veliausias(BestEil))
            if(i+gap1>Veliausias)

            break;
            JSeparator separator_5 = new JSeparator();
            separator_5.setOrientation(SwingConstants.VERTICAL);
            separator_5.setSize(new Dimension(2, 50));
            separator_5.setBounds(VK+desinen+149+gap+gap1, VV+138+Format, 3,
5);
            panel.add(separator_5);
                                                                    }

            JSeparator separator_4 = new JSeparator();
            separator_4.setOrientation(SwingConstants.VERTICAL);
            separator_4.setSize(new Dimension(2, 25));
            separator_4.setBounds(VK+desinen+149+gap, VV+137+Format, 3, 10);
            panel.add(separator_4);
            gap = gap + 60;
            i=i+59;
                                                        }

                                                        gap=0;
                                                        int sk=Veliausias+60;
                                                        for (int p=0;p<24;p++)
                                                        {
                                                                    if (sk-60>0)
                                                                    {
                                                                    JLabel Val =
DefaultComponentFactory.getInstance().createLabel(String.valueOf(p));
            Val.setBounds(VK+desinen+145+gap, VV+145+Format, 310-
238+Veliausias, 20);
            panel.add(Val);
                                                                    gap = gap +
60;
                                                                    }
                                                                    sk=sk-60;
                                                        }
```

```
                                        JLabel pra1 =
DefaultComponentFactory.getInstance().createLabel(" Atsitiktinai
sugeneruot\u0173 chromosom\u0173");

          pra1.setBounds(VK+desinen+149+Veliausias+20, 410, 250, 25);
                                        panel.add(pra1);

                                        if (GANau.TiksloFunkcija==1)
                                        {
                                                JLabel pra2 =
DefaultComponentFactory.getInstance().createLabel(" ma\u017Eiausios
prastovos: "+genvid+" min ");

          pra2.setBounds(VK+desinen+149+Veliausias+20, 435, 250, 25);
                                                panel.add(pra2);

                                                JLabel pra6 =
DefaultComponentFactory.getInstance().createLabel(" gamybos trukm\u0117:
"+VisosGamybosLaikas[0]+" min ");

          pra6.setBounds(VK+desinen+149+Veliausias+20, 460, 250, 25);
                                                panel.add(pra6);

                                        }
                                        if (GANau.TiksloFunkcija==2)
                                        {
                                                JLabel pra2 =
DefaultComponentFactory.getInstance().createLabel(" trumpiausia gamybos
trukm\u0117: "+genvid+" min ");

          pra2.setBounds(VK+desinen+149+Veliausias+20, 435, 250, 25);
                                                panel.add(pra2);

                                                // SKIRIASI INDEKSAI
                                                JLabel pra6 =
DefaultComponentFactory.getInstance().createLabel(" prastovos:
"+GamybosPrastovos[1]+" min ");

          pra6.setBounds(VK+desinen+149+Veliausias+20, 460, 250, 25);
                                                panel.add(pra6);
                                        }



//                                      System.out.println("Sugeneruotu
chrom +++Prastovos: "+GamybosPrastovos[0]+" Trukme: "+VisosGamybosLaikas[0]);
//                                      System.out.println("Sugeneruotu
chrom Prastovos: "+GamybosPrastovos[1]+" +++Trukme: "+VisosGamybosLaikas[1]);
//                                      System.out.println("Prastovos:
"+GamybosPrastovos[2]+" Trukme: "+VisosGamybosLaikas[2]);
//                                      System.out.println("Prastovos:
"+GamybosPrastovos[3]+" Trukme: "+VisosGamybosLaikas[3]);


                                        JLabel pra3 =
DefaultComponentFactory.getInstance().createLabel(" Po genetinio algoritmo
iteracij\u0173 ");

          pra3.setBounds(VK+desinen+149+Veliausias+20, 490, 250, 25);
                                        panel.add(pra3);


                                        if (GANau.TiksloFunkcija==1)
                                        {
```

```
                                        JLabel pra4 =
DefaultComponentFactory.getInstance().createLabel(" surastos ma\u017Eiausios
prastovos: "+String.valueOf(Totalm[0])+ " min");

          pra4.setBounds(VK+desinen+149+Veliausias+20, 515, 290, 25);
                                        panel.add(pra4);

                                        JLabel pra5 =
DefaultComponentFactory.getInstance().createLabel(" \u0161ios chromosomos
gamybos trukm\u0117: "+String.valueOf(VisosGamybosLaikas[2])+ " min");

          pra5.setBounds(VK+desinen+149+Veliausias+20, 540, 290, 25);
                                        panel.add(pra5);
                                        }
                                        if (GANau.TiksloFunkcija==2)
                                        {
                                        JLabel pra4 =
DefaultComponentFactory.getInstance().createLabel(" surasta ma\u017Eiausia
gamybos trukm\u0117: "+String.valueOf(Totalm[0])+ " min");

          pra4.setBounds(VK+desinen+149+Veliausias+20, 515, 290, 25);
                                        panel.add(pra4);

                                        JLabel pra5 =
DefaultComponentFactory.getInstance().createLabel(" \u0161ios chromosomos
prastovos: "+String.valueOf(GamybosPrastovos[2])+ " min");

          pra5.setBounds(VK+desinen+149+Veliausias+20, 540, 290, 25);
                                        panel.add(pra5);
                                        }









                                        //System.out.println("
          \n\n    TEST \n\n");

                                        sand=sand+1;
                                        }

                                        scrollPane_1.setBounds(350, 130,
650, 250);
                                        panel.add(scrollPane_1);

          scrollPane_1.setViewportView(txtrS);

                                        txtrS.setText("       Gaminio
kodas                  Kiekis
Gaminio pavadinimas\n");
                                        txtrS.append("-------------------
-----------------------------------"+
                                                     "-----------------
-------------------------------------------------------------\n");
```

```
                                        // --- ATKOMENTUOTI SIUOS KAI JAU
NORMALIAI PROGRAMUOSIU ---

                                        int[][] we= new
int[chromsk][rowNum];
                                        we=we1;

                                        for (int i =1; i<rowNum; i++)
                                {
                                if(msg.equals("Visi gaminiai
"))
                                        {
                                        txtrS.append(i+")
"+S1dataT[BestEil[i]][1]);
                                        txtrS.append("
           "+String.valueOf(S1dataN[BestEil[i]][2]));
                                        txtrS.append("
"+S1dataT[BestEil[i]][3]+"         "+S1dataT[BestEil[i]][0]+"\n");

           txtrS.setEditable(false);
                                        }
                                else
                                {

           if(msg.equals(S1dataT[BestEil[i]][1]))
                                        {

           txtrS.append(S1dataT[BestEil[i]][1]);
                                                txtrS.append("
           "+String.valueOf(S1dataN[BestEil[i]][2]));
                                                txtrS.append("
"+S1dataT[BestEil[i]][3]+"         "+S1dataT[BestEil[i]][0]+"\n\n");

           txtrS.setEditable(false);
                                        }

                                        }
                                }


                                        // SPAUSDINA GAMINIO OPERACIJU
SARASA I BALTA FONA
                                        if(!msg.equals("Visi gaminiai
"))
                                        {
                                        txtrS.append("    Gaminio kodas
Operacija          Gamyba(min)    \n");
                                        txtrS.append("-----------------
-------------------------------------------------------------\n");

                                                for (int i =1;
i<rowNum1; i++)
                                        {
                                                        // Nesuprasi
ka cia sitas veike... viskas veike, kai main() funkcijoje uzduodavau ilgi

           //S2dataT[i][0] += indent.substring(0, indent.length() -
S2dataT[i][0].length());

           //S2dataT[i][0] = S2dataT[i][0].substring(0,28);

           //S2dataT[i][1] += indent.substring(0, indent.length() -
S2dataT[i][1].length());
```

```
            //S2dataT[i][1] = S2dataT[i][1].substring(0,15);


            if(msg.equals(S2dataT[i][0]))
                                                    {

            txtrS.append(S2dataT[i][0]+"    "+S2dataT[i][1]+"
");

            txtrS.append(String.valueOf(S2dataN[i][2])+"\n");
                                                    }
                                                }
                                                }
                                                    txtrS.append("\n");


                                                    /*
                                                    for (int
j=1;j<rowNum;j++)


            System.out.println(BestEil[j]);
                                                    */


                                                    for (int
j=1;j<rowNum;j++)
                                                                {
                                                                int a =
randInt(0,255);
                                                                int b =
randInt(0,255);
                                                                int c =
randInt(0,255);

                                                    for (int i=1;i<n3;i++)
                                                    {


            //System.out.println(Gaminam(BestEil,j,i,1));

                                                                if
(Integer.valueOf(Gaminam(BestEil,j,i,0))>0)
                                                                {
                                                                 JButton
btnNewButton1 = new JButton("");


            btnNewButton1.setBounds(VK+desinen+150+Integer.valueOf(Gaminam(Best
Eil,j,i,1)), VV+EachOp(i,Gaminam(BestEil,0,i,0)),

                        Integer.valueOf(Gaminam(BestEil,j,i,0)), 15);

            btnNewButton1.setBackground(Color.BLUE);

            btnNewButton1.setBackground(new Color(a, b, c));

            panel.add(btnNewButton1);



            //asd=asd+1;
```

```
            //System.out.println(asd);

            //System.out.println((VK+desinen+150+Integer.valueOf(Gaminam(BestEi
l,j,i,1)))+"   "+(VV+EachOp(i,Gaminam(BestEil,0,i,0)))+"   ");


                                                        final int qqe=(int)j;
                                                        final int e1 = (int)i;


            btnNewButton1.addActionListener(new ActionListener() {
                                                                  public void
actionPerformed(ActionEvent e) {


            JOptionPane.showMessageDialog(panel.getComponent(0)," Gaminio kodas
                        "

                        +"        Operacija                          Gamyba(min)
             \n "+

                        S1dataT[BestEil[qqe]][1]+ Gaminam(BestEil,0,e1,0)+"
"+Gaminam(BestEil,qqe,e1,0)      );
                                                                  }});


                                                                  }
                                                        }
                                                        label.setText(indent);
                                                        }

                                                        Gaminsim=Gaminsim+1;



                                        }});

                                          }});
                                          }});
                                          }});
                                          }});
                                          }});
                                          }});
                                          }});
                                          }});



            }
}
```

2) GAGantt.java

```
import java.awt.Color;
import java.awt.Dimension;
import java.awt.EventQueue;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
```

```
import javax.swing.JButton;
import javax.swing.JFrame;
import javax.swing.JLabel;
import javax.swing.JOptionPane;
import javax.swing.JSeparator;
import javax.swing.SwingConstants;
import com.jgoodies.forms.factories.DefaultComponentFactory;

public class GAGantt {

		public static int desinen=GANau.desinen;
		public static int KK=0;
		public static int VV=-300;
		public static int n3=GANau.n3;
		public static Object window;
		private JFrame frame;

		public void Newclass() {

				EventQueue.invokeLater(new Runnable() {
					public void run() {
						try {
							GAGantt window = new
GAGantt();

		window.frame.setVisible(true);
						} catch (Exception e) {
							e.printStackTrace();

						}}});}

		public GAGantt() {
				initialize();
		}

		private void initialize() {



				int Veliausias=GANau.Veliausias(GANau.wee[GANau.MSGG]);

				frame = new JFrame();
				frame.setTitle("Chromosoma: "+GANau.MSGG);
				frame.setBounds(100, 100, KK+350+Veliausias, 820);
```

```
                        JLabel ButLab = new JLabel();
                        int k2=0;
                        int k=0;
                        for(int i=1; i<n3;i++)
                        {

                                // Paslenka zodzius
                        ButLab
=DefaultComponentFactory.getInstance().createLabel(GANau.OOO[i]);
                        ButLab.setBounds(KK+30, VV+320+k, 80+desinen, 15);
                        frame.add(ButLab);

                                // Horizontaliu liniju ilgis
                        JSeparator isilgai = new JSeparator();
                        isilgai.setBounds(KK+28, VV+317+k, 310-238+Veliausias+desinen,
2);
                        frame.add(isilgai);
                        k=k+20;
                        k2=i;           }

                                // Paslenka antraste
                        JLabel he = DefaultComponentFactory.getInstance().createLabel("
Gamybos eiliskumas ");
                        he.setBounds(KK+desinen+145+Veliausias, 2, 400, 40);
                        frame.add(he);

                                // Paslenka visus kodus
                        int yo=17;
                        for(int i=1;i<GANau.rowNum;i++) {
                                JLabel pav =
DefaultComponentFactory.getInstance().createLabel(i+")
"+GANau.S1dataT[GANau.wee[GANau.MSGG][i]][1]);
                                pav.setBounds(KK+desinen+145+Veliausias, yo, 400,
50);
                                frame.add(pav);
                                yo=yo+17;
                        }

                        JSeparator isilgai = new JSeparator();
                        isilgai.setBounds(KK+28, VV+317+k, 310-238+Veliausias+desinen,
2);
                        frame.add(isilgai);

                        // Paslenka antra vertikalia juosta
```

```
JSeparator separator_1 = new JSeparator();
separator_1.setOrientation(SwingConstants.VERTICAL);
separator_1.setSize(new Dimension(2, 25));
separator_1.setBounds(KK+desinen+99, VV+317, 3, k);
frame.add(separator_1);


JSeparator separator_2 = new JSeparator();
separator_2.setOrientation(SwingConstants.VERTICAL);
separator_2.setSize(new Dimension(2, 25));
separator_2.setBounds(KK+28, VV+317, 3, k);
frame.add(separator_2);


JSeparator separator_3 = new JSeparator();
separator_3.setOrientation(SwingConstants.VERTICAL);
separator_3.setSize(new Dimension(2, 25));
separator_3.setBounds(KK+desinen+100+Veliausias, VV+317, 3, k);
frame.add(separator_3);

//                              LAIKO JUOSTA

int Format=GANau.Format(k2);
JLabel laikas =
DefaultComponentFactory.getInstance().createLabel("Laikas, val");
laikas.setBounds(KK+desinen+27, VV+144+Format, 310-
238+Veliausias, 20);
frame.add(laikas);

JLabel Prast = DefaultComponentFactory.getInstance().createLabel("
Prastovos: "+GANau.Gaminam(GANau.wee[GANau.MSGG],0,0,0)+" min");
Prast.setBounds(KK+desinen+245, k+45, 200, 20);
frame.add(Prast);

int gap = 0;
int gap1;
        for (int i=0; i<Veliausias; i++)
{
                gap1=0;
        for (int j=1; j<4;j++)
        {
                gap1=gap1+15;
                if(i+gap1>Veliausias)
                        break;
```

```
                                                JSeparator separator_5 = new JSeparator();

            separator_5.setOrientation(SwingConstants.VERTICAL);
                                                separator_5.setSize(new Dimension(2,
50));

            separator_5.setBounds(KK+desinen+99+gap+gap1, VV+138+Format, 3, 5);
                                                frame.add(separator_5);
                                                }

                                                JSeparator separator_4 = new JSeparator();

            separator_4.setOrientation(SwingConstants.VERTICAL);
                                                separator_4.setSize(new Dimension(2,
25));

            separator_4.setBounds(KK+desinen+99+gap, VV+137+Format, 3, 10);
                                                frame.add(separator_4);
                                                gap = gap + 60;
                                                i=i+59;
                                    }

                                    gap=0;
                                    int sk=Veliausias+60;
                                    for (int p=0;p<24;p++)
                                    {
                                                if (sk-60>0)
                                                {
                                                JLabel Val =
DefaultComponentFactory.getInstance().createLabel(String.valueOf(p));
                                                Val.setBounds(KK+desinen+95+gap, VV+145+Format,
310-238+Veliausias, 20);
                                                frame.add(Val);
                                                gap = gap + 60;
                                                }
                                                sk=sk-60;
                                    }

                                    for (int j=1;j<GANau.rowNum;j++)
                                                {
                                                int a = GANau.randInt(0,255);
                                                int b = GANau.randInt(0,255);
                                                int c = GANau.randInt(0,255);
                                    for (int i=1;i<n3;i++)
```

```
{
                final JButton btnNewButton1 = new JButton("");


        btnNewButton1.setBounds(KK+desinen+100+Integer.valueOf(GANau.Gaminam(
GANau.wee[GANau.MSGG],j,i,1)),
VV+GANau.EachOp(i,GANau.Gaminam(GANau.wee[GANau.MSGG],0,i,0)),

        Integer.valueOf(GANau.Gaminam(GANau.wee[GANau.MSGG],j,i,0)), 15);

        btnNewButton1.setBackground(Color.BLUE);

                                        btnNewButton1.setBackground(new
Color(a, b, c));
                                        frame.add(btnNewButton1);
                frame.add(btnNewButton1);

        final int qqe=(int)j;
        final int e1 = (int)i;

        btnNewButton1.addActionListener(new ActionListener() {
                public void actionPerformed(ActionEvent e) {


        JOptionPane.showMessageDialog(frame.getComponent(0)," Gaminio kodas
          "
                                                                +"
Operacija          Gamyba(min)          \n "+

        GANau.S1dataT[GANau.wee[GANau.MSGG][qqe]][1]+
GANau.Gaminam(GANau.wee[GANau.MSGG],0,e1,0)+GANau.Gaminam(GANau.wee[GANa
u.MSGG],qqe,e1,0)       );
                                        }});}}

        JLabel dasd = DefaultComponentFactory.getInstance().createLabel("");
        dasd.setBounds(KK+27, VV+144+Format, 310-238+Veliausias, 20);
        frame.add(dasd);
        }}
```

3) GASXChart.java

```
import com.xeiam.xchart.Chart;
import com.xeiam.xchart.ChartBuilder;
import com.xeiam.xchart.SwingWrapper;
import com.xeiam.xchart.StyleManager.ChartType;
```

```
import com.xeiam.xchart.StyleManager.LegendPosition;

public class GASXChart {

	public static void NaujaClass() throws Exception {

		int vat =GANau.IteracNr;
		double[] X   = new double[vat];

		double[] VID = new double[vat];
		double[] MIN = new double[vat];
		double[] MAX = new double[vat];

//		double[] VIDPrast = new double[vat];
//		double[] MINPrast = new double[vat];
//		double[] VIDTrukm = new double[vat];
//		double[] MINTrukm = new double[vat];



		Chart chart;

		for(int i=0;i<vat;i++){
			X[i]=i+1;
//			MIN[i]=GANau.BestCr[i+1][1]; // Padaryti, kad butu su MinIkiInd - rodytu tikrai maziausia visada (kaip kad turetu buti pagal teorija)
//			VID[i]=GANau.BestCr[i+1][2];
//			MAX[i]=GANau.BestCr[i+1][7];

			MIN[i]=GANau.MIN1[i]; // Padaryti, kad butu su MinIkiInd - rodytu tikrai maziausia visada (kaip kad turetu buti pagal teorija)
			VID[i]=GANau.VID1[i];
			MAX[i]=GANau.MAX1[i];



			}

		String Tekstas = "";

		if (GANau.TiksloFunkcija==1)
			Tekstas="Prastov\u0173 "+"minimizavimas. Mutacija: "+GANau.mtc+".	"+" Po "+GANau.Totalm[1]+" iter. "+"MIN : "+(int)GANau.Totalm[0]+". Uztruko "+GANau.BestCr[0][0]+ " sec";
		if (GANau.TiksloFunkcija==2)
```

```
                                    Tekstas="Gamybos trukm\u0117s "+"minimizavimas.
Mutacija: "+GANau.mtc+".            "+" Po "+GANau.Totalm[1]+" iter. "+"MIN :
"+(int)GANau.Totalm[0]+". Uztruko "+GANau.BestCr[0][0]+ " sec";



                        if(GANau.vienasKartas==0){


                        chart = new
ChartBuilder().chartType(ChartType.Scatter).width(650).height(420).
                        //title("Prastov\u0173 VID ir MIN.  Mutacija: "+GANau.mtc+".
                "+" Po "+GANau.Totalm[1]+" iter. "+"MIN : "+(int)GANau.Totalm[0]+". Uztruko
"+GANau.BestCr[0][0]+ " sec").xAxisTitle("Iteracij\u0173 sk ").yAxisTitle("Prastovos, min
").build();
                        title(Tekstas).xAxisTitle("Iteracij\u0173 sk ").yAxisTitle("Prastovos /
Gamybos trukm\u0117, min ").build();

                        if (GANau.TiksloFunkcija==1)
                        {
                        chart.addSeries("Prastov\u0173 minimumai ", X, MIN);
                        chart.addSeries("Prastov\u0173 vidurkiai ", X, VID);
                        chart.addSeries("Atrinktos naujos imties maksimumai ", X, MAX);
                        }

                        if (GANau.TiksloFunkcija==2)
                        {
                        chart.addSeries("Gamybos trukm\u0117s minimumai ", X, MIN);
                        chart.addSeries("Gamybos trukm\u0117s vidurkiai ", X, VID);
                        chart.addSeries("Atrinktos naujos imties maksimumai ", X, MAX);
                        }

//                      if (GANau.TiksloFunkcija==2)
//                      chart.addSeries("Gamybos prastovos ", X, MINPrast);
//                      chart.addSeries("Vid Prastovos ", X, VIDPrast);

//                      if (GANau.TiksloFunkcija==1)
//                      chart.addSeries("Gamybos trukm\u0117 ", X, MINTrukm);

//                      chart.addSeries("Vid Gamybos trukme ", X, VIDTrukm);
                        // Sitoje vietoje padaryti kad butu dar dvi eilutes -

                        GANau.vienasKartas=GANau.vienasKartas+1;
                        }
```

```
                else
                        {
                        if(GANau.mtc==GANau.MutacRus){
                        chart = new
ChartBuilder().chartType(ChartType.Scatter).width(450).height(250).
                        title("Mutacija: "+GANau.mtc+".          "+" Po
"+GANau.Totalm[1]+" iter. "+"MIN: "+(int)GANau.Totalm[0]+"\n"+". Uztruko
"+GANau.laikas+ " sec").xAxisTitle("Iteracij\u0173 sk ").yAxisTitle("Prastovos / Gamybos
trukm\u0117, min ").build();
                        chart.addSeries("Minimumai ", X, GANau.MIN1);
                        chart.addSeries("Vidurkiai ", X, GANau.VID1);
                        chart.addSeries("Maksimumai ", X, GANau.MAX1);
                        }
                        else
                        {
                                chart = new
ChartBuilder().chartType(ChartType.Scatter).width(450).height(250).
                                title("Mutacija: "+GANau.MutacRus+".
          "+" Po "+GANau.Best+" iter. "+"MIN: "+(int)GANau.BestCr[0][0]+"\n"+".
Uztruko "+GANau.TT[GANau.MutacRus]+ " sec").xAxisTitle("Iteracij\u0173 sk
").yAxisTitle("Prastovos / Gamybos trukm\u0117, min ").build();
                                chart.addSeries("Minimumai ", X, MIN);
                                chart.addSeries("Vidurkiai ", X, VID);
                                chart.addSeries("Maksimumai ", X,
MAX);
                        }
                        }
                // Customize Chart

        chart.getStyleManager().setLegendPosition(LegendPosition.OutsideE);
                chart.getStyleManager().setAxisTitlesVisible(true);
           // Show it
                  new SwingWrapper(chart).displayChart();
                }}
```