KAUNO TECHNOLOGIJOS UNIVERSITETAS

EVELINA STANEVIČIENĖ

HIBRIDINIŲ GENETINIŲ ALGORITMŲ TAIKYMAS PILKŲJŲ ŠABLONŲ FORMAVIMO UŽDAVINIUI

Daktaro disertacija
Gamtos mokslai, informatika (N 009)

2019, Kaunas

Disertacija rengta 2013–2018 metais Kauno technologijos universiteto Informatikos fakulteto Multimedijos inžinerijos katedroje.

Mokslinis vadovas:

Prof. dr. Alfonsas MISEVIČIUS (Kauno technologijos universitetas, gamtos mokslai, informatika, N 009.

Interneto svetainės, kurioje skelbiama disertacija, adresas:
http://ktu.edu

Redagavo:

Inga Nanartonytė

ISBN 978-609-02-1625-5

Leidinio bibliografinė informacija pateikiama Lietuvos nacionalinės Martyno Mažvydo bibliotekos Nacionalinės bibliografijos duomenų banke (NBDB).

TURINYS

SANTRUMPOS

2D – dvimatė grafika (angl. *2D computer graphics*);
3D – trimatė grafika (angl. *3D computer graphics*);
GA – genetiniai algoritmai (angl. *genetic algorithms*);
GAP – godžioji adaptyvioji procedūra (angl. *greedy adaptive procedure*);
GDU – grafų dalijimo uždavinys (angl. *graph partitioning problem*);
GGAP – greitoji godžioji adaptyvioji procedūra (angl. *fast greedy adaptive procedure*);
GŽR – geriausia žinoma reikšmė (angl. *best known value*);
GŽS – geriausias žinomas sprendinys (angl. *best known solution*);
HGA – hibridiniai genetiniai algoritmai (angl. *hybrid genetic algorithms*);
HITP – hierarchinė iteratyvioji tabu paieška (angl. *hierarchical iterated tabu search*);
ITP – iteratyvioji tabu paieška (angl. *iterated tabu search*);
KP – kvadratinio paskirstymo uždavinys (angl. *quadratic assignment problem*);
KP-1 – modifikacija „DAŽNUMU PAREMTAS GODUSIS KRYŽMINIMAS“;
KP-2 – modifikacija „GODUSIS PRIEŠINGASIS (OPOZICINIS) KRYŽMINIMAS“;
KP-3 – modifikacija „GODUSIS KRYŽMINIMAS“;
KP-4 – modifikacija „EURISTINIS PRIEŠINGASIS (OPOZICINIS) KRYŽMINIMAS“;
KP-5 – modifikacija „EURISTINIS KRYŽMINIMAS“;
KP-6 – modifikacija „DAUGELIO TĖVŲ GENŲ DAŽNUMU PAREMTAS PRIEŠINGASIS (OPOZICINIS) KRYŽMINIMAS“;
KP-7 – modifikacija „DAUGELIO TĖVŲ GENŲ DAŽNUMU PAREMTAS KRYŽMINIMAS“;
KP-8 – modifikacija „PRIDERINTASIS (SPECIFINIS) PRIEŠINGASIS (OPOZICINIS) KRYŽMINIMAS“;
KP-9 – modifikacija „PRIDERINTASIS (SPECIFINIS) KRYŽMINIMAS“;
KP-10 – modifikacija „DAŽNUMU PAREMTAS GODUSIS PRIEŠINGASIS (OPOZICINIS) KRYŽMINIMAS“;
KS-1 – modifikacija „PADIDINTAS KRYŽMINIMŲ SKAIČIUS“;
KS-2 – modifikacija „VIENAS KRYŽMINIMAS“;
KU – komivojažieriaus uždavinys (angl. *traveling salesman problem*);
LP – lokalioji (arba aplinkos) paieška (angl. *local (or neighbourhood) search*);
LP-1 – modifikacija „TABU PAIEŠKOS ITERACIJŲ SKAIČIAUS MAŽINIMAS“;
LP-2 – modifikacija „TABU PAIEŠKOS ITERACIJŲ SKAIČIAUS DIDINIMAS“;
LP-3 – modifikacija „HIERARCHINĖS ITERATYVIOSIOS TABU PAIEŠKOS ITERACIJŲ SKAIČIAUS MAŽINIMAS“;
LP-4 – modifikacija „HIERARCHINĖS ITERATYVIOSIOS TABU PAIEŠKOS ITERACIJŲ SKAIČIAUS DIDINIMAS“;
MSU – maksimalaus skirtingumo uždavinys (angl. *maximum diversity problem*);
PA-1 – modifikacija „BLOGIAUSIO POPULIACIJOS NARIO PAKEITIMAS GERESNIU PALIKUONIU“;
PA-2 – modifikacija „BLOGIAUSIO POPULIACIJOS NARIO PAKEITIMAS“;

PA-3 – modifikacija „BLOGESNIOJO TĖVO PAKEITIMAS GERESNIU PALIKUONIU“;
PA-4 – modifikacija „BLOGESNIOJO TĖVO PAKEITIMAS“;
PA-5 – modifikacija „BLOGIAUSIO POPULIACIJOS NARIO PAKEITIMAS SU PAPILDOMA SĄLYGA“;
PGEA – pagerintasis genetinis evoliucinis algoritmas (angl. *improved genetic-evolutionary algorithm*, *IGEA*);
PP-1 – modifikacija „ATSITIKTINĖ POPULIACIJA“;
PP-2 – modifikacija „PADIDINTA IR PAGERINTA POPULIACIJA“;
PP-3 – modifikacija „PAGERINTA POPULIACIJA“;
PR-1 – modifikacija „POPULIACIJOS PERTVARKYMO NETAIKYMAS“;
PR-2 – modifikacija „MULTIMUTACIJA PAREMTAS POPULIACIJOS PERTVARKYMAS“;
PR-3 – modifikacija „PRIEŠINGŲ (OPOZICINIŲ) SPRENDINIŲ GENERAVIMU PAREMTAS POPULIACIJOS PERTVARKYMAS“;
PR-4 – modifikacija „GENŲ TRANSLOKACIJA PAREMTAS POPULIACIJOS PERTVARKYMAS;
PR-5 – modifikacija „DETERMINUOTUOJU CHAOSU PAGRĮSTAS POPULIACIJOS PERTVARKYMAS“;
PR-6 – modifikacija „DETERMINUOTUOJU CHAOSU PAGRĮSTAS MODIFIKUOTAS POPULIACIJOS PERTVARKYMAS“;
PR-7 – modifikacija „ATSITIKTINĖS POPULIACIJOS GENERAVIMU PAREMTAS POPULIACIJOS PERTVARKYMAS“;
PŠF – pilkųjų šablonų formavimo uždavinys (angl. *grey pattern problem*);
SB – sėkmingų bandymų skaičius (angl. *success rate*);
STA-1 – modifikacija „RULETĖS RATĄ MODELIUOJANTI ATRANKOS PROCEDŪRA“;
STA-2 – modifikacija „SPRENDINIŲ RANGU PAGRĮSTA ATRANKOS PROCEDŪRA“;
TP – tabu paieška (angl. *tabu search*).

ĮVADAS

Gana aktuali informatikos mokslo sritis yra optimizavimo metodai ir algoritmai, jų sudarymas ir tyrimas. Reikšmingą šių metodų grupę sudaro modernūs euristiniai algoritmai, tarp jų ir hibridiniai genetiniai algoritmai.

Modernūs euristiniai algoritmai sėkmingai išvystyti sprendžiant sudėtingus teorinius matematinius ir taikomuosius praktinius uždavinius. Tokio uždavinio pavyzdžiai yra kvadratinio paskirstymo uždavinys (angl. *quadratic assignment problem*) (KP) ir jo atskiras atvejis – pilkųjų šablonų formavimo uždavinys (angl. *grey pattern problem*) (PŠF).

Šiame darbe dėmesys sutelkiamas būtent į efektyvių euristinių algoritmų kūrimą ir taikymą pilkųjų šablonų formavimo uždaviniui spręsti, kai siekiama gauti kuo tikslesnį (galimai optimalų) iš anksto apibrėžto intensyvumo pilką atspalvį. Darbe siekta tiek teorinių (iš pradžių buvo sukurti efektyvių euristinių algoritmų prototipai, skirti sudėtingų teorinių kombinatorinių optimizavimo uždavinių klasei), tiek praktinių tikslų (prototipai adaptuoti konkrečiam praktiniam techniniam uždaviniui).

PŠF uždaviniui gali būti pritaikyti įvairių tipų euristiniai algoritmai, pradedant lokaliosios paieškos algoritmais, baigiant biologinių procesų inspiruotais kolektyvinio intelekto imitavimo algoritmais. Svarbus aspektas yra susijęs su hibridizavimu, kai bandoma pasiekti naudingą sąveiką kombinuojant (derinant) skirtingus komponentus (algoritmus). Geras pavyzdys galėtų būti greitosios vieno sprendinio pagrindu paremtos euristikos ir populiacija paremto (evoliucinio) algoritmo kombinavimas. Kitas svarbus aspektas yra prieš tai sukonstruotų efektyvių euristinių algoritmų daugkartinis taikymas.

Tyrimo objektas. Tyrimo objektas yra hibridiniai genetiniai algoritmai, skirti pilkųjų šablonų formavimo uždaviniui spręsti.

Tyrimo tikslas. Darbo tikslas yra ištirti hibridinio genetinio algoritmo modifikacijų pilkųjų šablonų formavimo uždaviniui veikimo efektyvumą ir nustatyti efektyviausius algoritmo variantus.

Tyrimo uždaviniai. Tikslui pasiekti išsikelti šie darbo uždaviniai:

1. Atlikti pilkųjų šablonų formavimo uždaviniui taikomų algoritminių sprendimo būdų žvalgomąjį tyrimą.
2. Pasiūlyti ir realizuoti naują hibridinį genetinį algoritmą pilkųjų šablonų formavimo uždaviniui.
3. Eksperimentiškai ištirti algoritmo efektyvumą ir išanalizuoti gautus rezultatus.

Tyrimo metodika. Tyrimo metodika paremta euristinių ir metaeuristinių kombinatorinio optimizavimo algoritmų taikymu, kompiuterine algoritmų efektyvumo analize.

Darbo mokslinis naujumas ir praktinė vertė. Mokslinį darbo naujumą leidžia pagrįsti šie darbo rezultatai:

1. Pasiūlytas patobulintos struktūros (architektūros) hibridinis genetinis algoritmas pilkųjų šablonų formavimo uždaviniui spręsti.

2. Realizuotas hibridinio genetinio algoritmo integravimas su naujo tipo iteratyviosios tabu paieškos procedūra, kombinuojama su efektyviu adaptyviu sprendinių pertvarkymu.

Darbas turi praktinę vertę: sudarytas algoritmas sėkmingai išbandytas sprendžiant didelės apimties pilkųjų šablonų formavimo uždavinius. Tai leidžia sukurti geresnės kokybės spalvų atspalvius, skaitmeninius vaizdus. Visa tai yra aktualu pritaikant modernius euristinius algoritmus praplėstojoje (virtualiojoje) realybėje.

Ginamieji teiginiai:

1. Sudarytas hibridinis genetinis algoritmas yra tinkamas ir efektyvus būdas spręsti pilkųjų šablonų formavimo uždavinį.
2. Kompiuteriniais eksperimentais nustatyta, kad sukurtas hibridinis genetinis algoritmas yra pranašesnis už kitus euristinius algoritmus šiam uždaviniui spręsti.

Darbo rezultatų aprobavimas. Paskelbti 6 moksliniai straipsniai disertacijos tema, 2 iš jų – Mokslinės informacijos instituto (ISI) pagrindinio sąrašo leidiniuose, turinčiuose citavimo indeksą. Disertacijos tema atliktų tyrimų rezultatai buvo pristatyti 3 tarptautinėse mokslinėse konferencijose.

Darbo struktūra. Disertaciją sudaro įvadas, trys dalys, išvados, priedai, literatūros šaltinių ir paskelbtų mokslinių publikacijų darbo tema sąrašai. Darbo apimtis yra 88 puslapiai. Tekste panaudotos 26 formulės, pateikti 38 paveikslai ir 30 lentelių. Rengiant disertaciją remtasi 93 literatūros šaltiniais.

1. LITERATŪROS APŽVALGA

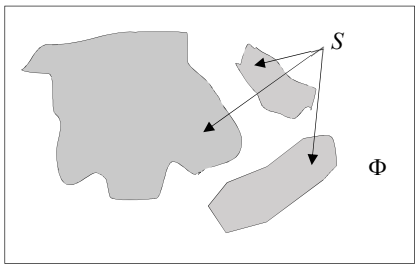
Sprendžiant optimizavimo uždavinius (angl. *optimization problems*), siekiama rasti geriausią kintamųjų konfigūraciją, kitaip tariant, geriausią (galimai optimalų) sprendinį. Optimizavimo uždavinius galima suskirstyti į dvi pagrindines kategorijas pagal kintamųjų prigimtį: naudojami tolydžiojo tipo kintamieji arba diskretinio tipo kintamieji. Kai kintamieji yra tolydžiojo tipo, paprastai ieškoma realiųjų skaičių rinkinio. Sprendžiant kombinatorinio optimizavimo uždavinius (angl. *combinatorial optimization problems*), naudojami diskretinio tipo kintamieji. Tokiu atveju kintamųjų reikšmių aibė yra baigtinė arba bent jau suskaičiuojama [1–3].

Formaliai kombinatorinio optimizavimo uždavinys gali būti aprašytas kaip pora (*S, f*) [4], kur *S* yra leistinų sprendinių aibė (angl. *set of feasible solutions*), *f* yra funkcija, kurios apibrėžimo sritis – aibė *S*. Ši funkcija vadinama optimizavimo uždavinio tikslo funkcija (angl. *objective function*) [1]. Aibę *S* galima apibrėžti kaip bazinės aibės Φ (angl. *ground set*) poaibį (1 pav.). Aibė Φ yra visų galimų reikšmių aibė [5]. Jeigu tikslo funkcija *f* turi būti minimizuojama (kitu atveju gali būti maksimizuojama), tada išspręsti kombinatorinio optimizavimo uždavinį (*S, f*) reiškia rasti tokį sprendinį $S^* \in S$, kur

$$f(S^*) \leq f(s); \tag{1}$$

čia $s \in S$.

Sprendinys S^*, kuris minimizuoja tikslo funkciją *f*, yra uždavinio (*S, f*) optimalus sprendinys.



1 pav. Bazinė ir leistinų sprendinių aibės

Nemažą dalį kombinatorinio optimizavimo uždavinių sudaro uždaviniai, kuriuose kintamųjų reikšmių tipas yra perstatymas (angl. *permutation*). Perstatymą galima apibūdinti kaip kurios nors baigtinės aibės nesikartojančių elementų seką. Aibė *S* formaliai aprašoma taip:

$$S = \{\, s \mid s = (s(1), s(2), ..., s(n)),\ s(i) \in \{\, 1, 2, ..., n \,\},\ i = 1, 2, ..., n,$$
$$s(i) \neq s(j),\ i, j = 1, 2, ..., n,\ i \neq j \,\}; \tag{2}$$

čia *s* – perstatymas, *s*(*i*) – perstatymo *s* *i*-tasis elementas, *n* – uždavinio apimtis.

Algoritmus, skirtus optimizavimo uždaviniams spręsti, galima suskirstyti į tris pagrindines kategorijas: 1) tikslieji algoritmai, kuriuos taikant randamas optimalus uždavinio sprendinys (algoritmų vykdymo laikas ilgėja eksponentiškai, didėjant uždavinio apimčiai); 2) aproksimaciniai algoritmai, kurie užtikrina, kad gauto

sprendinio kokybė skirsis nuo geriausio galimo sprendinio kokybės ne daugiau kaip iš anksto nustatyta paklaida; 3) euristiniai algoritmai, leidžiantys gauti geros kokybės (artimą optimaliam) sprendinį per priimtiną skaičiavimų laiką. Euristinių algoritmų pranašumas prieš tiksliuosius ir aproksimacinius algoritmus yra tai, kad nebelieka eksponentinės paieškos laiko priklausomybės nuo sprendžiamo uždavinio apimties. Euristinius algoritmus sudaro keliolika atskirų algoritmų grupių, tokių kaip: atkaitinimo modeliavimas (angl. *simulated annealing*), genetiniai algoritmai (angl. *genetic algorithms*), tabu paieška (angl. *tabu search*), godžioji randomizuota adaptyvioji paieška (angl. *greedy randomized adaptive search*) ir kt. [5, 6].

Euristiniai algoritmai taikomi spręsti sudėtingiems kombinatorinio optimizavimo uždaviniams, tokiems kaip kvadratinio paskirstymo uždavinys [7–10], pilkųjų šablonų formavimo uždavinys, komivojažieriaus (keliaujančio pirklio) uždavinys (angl. *traveling salesman problem*) (KU) [11–13], maksimalaus skirtingumo uždavinys (angl. *maximum diversity problem*) (MSU) [14–16], grafų dalijimo uždavinys (angl. *graph partitioning problem*) (GDU) [17].

1.1. Pilkųjų šablonų formavimo uždavinys

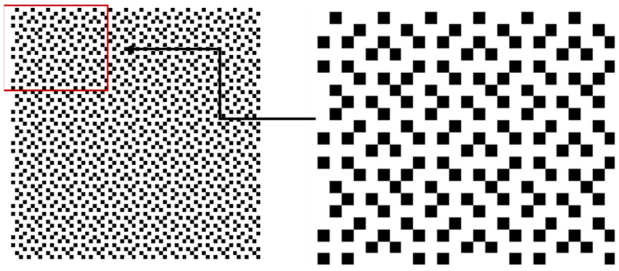
Pilkųjų šablonų formavimo uždavinį galima aprašyti taip: duotas $n_1 \times n_2$ matmenų tinklelis, kurį sudaro $n = n_1 \times n_2$ vienodo dydžio kvadratėlių (taškų) [10]. Tinklelio eilučių skaičius yra lygus n_1, stulpelių skaičius – n_2. Sprendžiant PŠF uždavinį laikoma, kad standartiniu atveju kvadratėliai yra juodos arba baltos spalvos. (Sprendžiant spalvotų šablonų formavimo uždavinį, spalvos gali skirtis.) Baltos spalvos kvadratėliai dažniausiai laikomi fonu (angl. *background*), o juodos – pagrindine spalva (angl. *foreground*). Yra duotas parametras $m < n$, kuris nurodo juodos spalvos kvadratėlių skaičių. Tuomet baltos spalvos kvadratėlių skaičius yra lygus $n - m$.

Turimus juodus ir baltus kvadratėlius tvarkingai išdėsčius tinklelyje, gaunami vadinamieji pilkieji šablonai (angl. *grey patterns*). Šablono spalvos intensyvumas (angl. *grey density*) yra lygus $\frac{m}{n}\left(0 \le \frac{m}{n} \le 1\right)$. Kuo mažesnė parametro *m* reikšmė, tuo spalva „šviesesnė“ ir artimesnė baltai spalvai. Ir, atvirkščiai, kuo *m* didesnis, tuo spalva „tamsesnė“ ir artimesnė juodai spalvai.

Taigi, yra duotas bendras kvadratėlių skaičius *n* ir parametro *m* reikšmė[1]. Tikslas yra išdėstyti tinklelyje baltus ir juodus kvadratėlius kiek įmanoma tolygiau. Kitaip tariant, reikia suformuoti kuo taisyklingesnį pilkųjų atspalvių šabloną, imituojantį pageidaujamo intensyvumo pilką spalvą [18, 19]. Pilkųjų šablonų pavyzdžiai pateikiami 2 pav.

Detali PŠF uždavinio matematinė formuluotė pateikiama 2.2. poskyryje.

[1] Dar galima sakyti, kad yra duota n elementų ir skaičius m ($m < n$), kuris nurodo kiek elementų turi būti atrinkta. Taip pat yra nurodyti atstumai tarp visų elementų. Siekiama surasti elementų poaibį, susidedantį iš m elementų ir tokį, kad visų šio poaibio elementų porų atstumų suma būtų minimali.



2 pav. Pilkųjų šablonų pavyzdžiai

1.2. Pilkųjų šablonų formavimo uždavinio sprendimas: euristiniai algoritmai

Pilkųjų šablonų formavimo uždavinys [10, 20, 21] formuluojamas matematiškai kaip atskiras kvadratinio paskirstymo uždavinio atvejis [10]. Nors šis uždavinys skirtas pilkiesiems atspalviams generuoti (angl. *digital halftoning*) [22, 23], jis gali būti pritaikomas bet kuriai turimos spalvų paletės diskretinei spalvai [18].

Pilkųjų šablonų formavimo uždaviniui spręsti taikomi tokie euristiniai algoritmai, kaip godusis algoritmas (angl. *greedy algorithm*) (GoA) [24], greičiausio nusileidimo algoritmas (angl. *steepest descent algorithm*) (GNA) [24], tabu paieška (TP) [24–27], atkaitinimo modeliavimo (AM) algoritmas [24], skruzdėlių elgseną imituojantis algoritmas (angl. *ant system*) (SEI) [28, 29], hibridinis genetinis algoritmas (angl. *hybrid genetic algorithm*) (HGA) [20, 28, 30].

1.2.1. Godusis algoritmas

Godžiojo algoritmo veikimas yra pagrįstas tokia paieškos strategija, kai kiekviename algoritmo žingsnyje pasirenkamas tuo momentu geriausias sprendinys [31, 32]. Daugeliu atvejų GoA nesuranda globaliai optimalaus sprendinio, tačiau ši euristika gali padėti rasti pakankamai gerą sprendinį per labai trumpą laiką. Taikant GoA yra formuojamas vadinamasis dalinis sprendinys (klasteris) [24]. Sprendinys konstruojamas taikant atitinkamus žingsnius ir kiekviename algoritmo žingsnyje esamas sprendinys yra papildomas (išplečiamas).

Pagrindiniai algoritmo žingsniai gali būti aprašomi taip:

1. Dalinis sprendinys pradedamas pildyti nuo pirmojo atskaitinio elemento (tuo metu geriausio pasirinkimo).
2. Tikrinami visi elementai, kurie nėra įtraukti į dalinį sprendinį, ir skaičiuojama atstumų tarp visų šių elementų ir elementų, kurie yra įtraukti į dalinį sprendinį, suma.
3. Į dalinį sprendinį įtraukiamas elementas, turintis mažiausią atstumų nuo visų sprendinio elementų sumą, t. y. pridedamas elementas iš dar nepasirinktų elementų $\{j: j = 1, \dots, n\} \setminus \{p(i): i = 1, \dots k\}$ su mažiausia atstumų suma ($p(1), \dots, p(k)$ – esamas sprendinys).
4. Kartojama nuo 2 žingsnio tol, kol sprendinį sudarys m taškų ($m < n$).

Žingsniai kartojami n kartų tol, kol kiekvienas elementas bus pasirinktas kaip atskaitinis elementas. Geriausias sprendinys, gautas iš visų n galimų sprendinių, yra godžiojo algoritmo rezultatas.

Literatūroje [24] pateikiamuose eksperimentų rezultatuose GoA vykdymo laikas visais 126 m atvejais (kai $n = 256$) buvo trumpesnis nei 2 minutės [24]. Geriausią žinomą reikšmę (angl. *best known value*) (GŽR) pavyko pasiekti su 11 m reikšmių (m = 3, 4, 8, 16, 112, 120, 124–128).

1.2.2. Greičiausio nusileidimo algoritmas

Greičiausio nusileidimo algoritmas [24] yra labai panašus į euristinį algoritmą, kuris buvo pasiūlytas Teitzo ir Bart [33] p-medianos uždaviniui (angl. *p-median problem*) spręsti.

1. Pasirenkamas pradinis sprendinys M, sudarytas iš m elementų, $m < n$.
2. Vertinamas tikslo funkcijos reikšmės pokytis atliekant visus galimus elementų, esančių sprendinyje, ir elementų, kurie nepriklauso sprendiniui M, sukeitimus.
3. Jeigu randama tokia elementų pora, kai, elementus sukeitus vietomis, tikslo funkcijos reikšmė pagerinama, tada elementai sukeičiami ir kartojama nuo 2 žingsnio. Priešingu atveju vykdomas 4 žingsnis.
4. Gautas sprendinys yra algoritmo rezultatas. Algoritmas baigiamas.

Šiuo atveju tikslo funkcijos reikšmės pokytis (žr. 2 žingsnį) apskaičiuojamas pagreitintu būdu.

Aprašytuose eksperimentuose [24] GNA vykdymo laikas visais 126 m atvejais (kai $n = 256$) buvo apytiksliai 4 minutės [24]. GŽR pavyko pasiekti su 25 m reikšmėmis (m = 3–17, 20, 21, 23, 24, 31, 33, 123, 125, 126, 128).

1.2.3. Atkaitinimo modeliavimo algoritmas

Pagrindinė AM algoritmo idėja pagrįsta statistinės mechanikos dėsniais. Jį taikant imituojami sistemų įkaitinimo ir atkaitinimo procesai [34]. Atkaitinimo modeliavimo algoritmas optimizavimo uždaviniams spręsti buvo pasiūlytas Kirkpatricko, Gelato ir Vecchi [35]. AM algoritmą galima aprašyti taip: (S, f); čia S – sprendinių aibė, f – tikslo funkcija. Vykdyti algoritmą pradedama nuo atsitiktinai sugeneruoto sprendinio ($s \in S$). Sprendinio s aplinkoje S' ($S' \subseteq S$) parenkamas sprendinys s' ir apskaičiuojamas tikslo funkcijos pokytis: $\triangle f = f(s') - f(s)$. Jeigu $\triangle f < 0$, esamas sprendinys s pakeičiamas sprendiniu s'. Kitu atveju sprendinys keičiamas su tikimybe $P(\triangle f) = e^{-\triangle f/t}$; čia t – temperatūra, kuri palaipsniui mažinama iki nustatytos apatinės ribos.

Literatūroje [24] aprašomas AM algoritmas, pritaikytas PŠF uždaviniui spręsti, turi šiuos pagrindinius žingsnius:

1. Generuojamas pradinis sprendinys M ($|M| < n$). Nustatoma temperatūra $t = t_0$; čia t_0 – pradinė temperatūra.
2. Atsitiktiniu būdu parenkamas elementas, kuris bus pašalintas iš sprendinio M. Į šio elemento vietą įdedamas elementas, kuris nepriklauso sprendiniui M. Taip suformuojamas naujas sprendinys M'.
3. Skaičiuojamas tikslo funkcijos pokytis: $\triangle f = f(M') - f(M)$.
4. Jeigu $\triangle f < 0$, M pakeičiamas M' ir atliekamas 7 žingsnis.
5. Sprendiniai sukeičiami (M pakeičiamas M') su tikimybe $P(\triangle f) = e^{-\triangle f/t}$. Kitu atveju paliekamas esamas sprendinys.

6. Jeigu nepasiektas iteracijų skaičius, kartojama nuo 2 žingsnio.
7. Temperatūros parametras t dauginamas iš aušinimo faktoriaus α.
8. Gautas sprendinys yra algoritmo rezultatas. Algoritmas baigiamas.

Eksperimentuose [24] taikant AM algoritmą PŠF uždaviniui, kai uždavinio dydis $n = 256$, net ir išbandžius skirtingas pradines parametrų reikšmes, gerų rezultatų gauti nepavyko.

1.2.4. Skruzdėlių elgseną imituojantis algoritmas

Šeštojo dešimtmečio pabaigoje imta domėtis vabzdžių kolektyviniu elgesiu. Ieškodamos maisto gamtoje skruzdėlės pirmiausia paiešką atlieka atsitiktinai pasirinkdamos kelią netoli skruzdėlyno. Judėdamos jos išskiria feromonų ir palieka jų pėdsaką (tokiu būdu randa kelią atgal). Radusios maisto, skruzdėlės neša jį į skruzdėlyną, judėdamos tuo pačiu keliu, kuriuo atėjo. Po kurio laiko keliai iki maisto šaltinio bus pažymėti skirtingais kiekiais feromonų, t. y. artimesnis kelias bus pažymėtas didesniu kiekiu feromonų, nes juo bus dažniau einama. Taip skruzdėlės gali optimizuoti maršrutus ieškodamos maisto. 1991 m. pagal skruzdėlių elgseną buvo sukurtas skruzdėlių sistemos algoritmas [36].

Taillardo ir Gambardellos [28] aprašoma greitoji skruzdėlių elgseną imituojanti sistema (angl. *fast ant system*, *FANT*) (GSEI) turi integruotą lokaliąją paiešką, kuri taikoma sprendiniui, gautam pritaikius GSEI algoritmą, pagerinti. Algoritmas taip pat turi pakoreguotą statistinių duomenų rinkimo atmintį, kuri realizuota kaip $n \times n$ dydžio matrica, – feromonų matricą. Pagrindinė algoritmo idėja yra įvertinti feromonų matricos elementų f_{ij} įtaką sprendiniui (perstatymui) p. Feromonų (atminties) matricos F laukelis f_{ij} saugo $p_i = j$ duomenis iš prieš tai paieškos proceso metu sugeneruoto sprendinio. Matrica F saugo skruzdėlių paliktus feromonų pėdsakus. Algoritmas buvo panaudotas bendresniam už PŠF uždavinį KP uždaviniui spręsti.

Pagrindiniai GSEI algoritmo žingsniai:

1. Sugeneruojama PD dydžio pradinė skruzdėlių populiacija (kolonija); čia PD – pradinis parametras, nurodantis populiacijos dydį. Įvertinus atstumą tarp populiacijos narių, apskaičiuojamos pradinės feromonų matricos F reikšmės.
2. Atsitiktinai (arba taikant papildomas sąlygas) atrenkamas populiacijos sprendinys s.
3. Įvertinus matricos F reikšmes, iš atrinkto sprendinio s gaunamas naujas sprendinys s'.
4. Sprendinys s' pagerinamas.
5. Populiacija atnaujinama įtraukiant į ją sprendinį s'.
6. Perskaičiuojamos feromonų matricos F reikšmės.
7. Pereinama į kitą iteraciją, t. y. vykdomas 2 žingsnis. Jeigu atliktos visos iteracijos, algoritmas baigiamas. Algoritmo rezultatas yra geriausias gautas populiacijos sprendinys.

Gambardella, Taillardas ir Dorigo [29] aprašė panašų hibridinį skruzdėlių algoritmą, pagal kurį atmintis yra naudojama modifikuoti esamam sprendiniui taikant lokaliosios paieškos algoritmą.

1.2.5. Tabu paieška

Tabu paieškos metodas buvo pasiūlytas 1987 metais Hanseno ir Jaumard [37]. Prie metodo pradininkų galima priskirti ir Gloverį, kurio straipsniuose šis metodas buvo išsamiai aprašytas [38, 39].

Tabu paieška iki šiol yra gana sėkmingai taikoma kombinatorinio optimizavimo uždaviniams, tokiems kaip kvadratinio paskirstymo, tvarkaraščių sudarymo [40, 41] ir kt., spręsti.

Taikant tabu paieškos metodą, paieška pradedama nuo pradinio sprendinio *s*. Toliau vykdoma iteracinė paieška keliaujant nuo sprendinio prie sprendinio. Paieškos metu tikrinama sprendinio *s* aplinka $N(s)$ siekiant surasti sprendinį, turintį pagerintą (mažesnę, kai funkcija minimizuojama, ir didesnę priešingu atveju) tikslo funkcijos *f* reikšmę. Tabu paieškoje galimi perėjimai, ne tik pagerinantys tikslo funkcijos reikšmę, bet ir pabloginantys ją (tai esminis skirtumas nuo lokaliosios paieškos algoritmų, kuriuos taikant draudžiami perėjimai prie blogesnio (turinčio blogesnę tikslo funkcijos reikšmę) sprendinio). Neradus geresnio sprendinio yra galimas perėjimas prie sprendinio, kuris mažiausiai pablogina tikslo funkcijos reikšmę [42, 43]. Tabu paieška paremta draudimais, t. y. siekiama išvengti grįžimo prie jau buvusių sprendinių (priešingu atveju būtų tikimybė, kad paieška bus pradedama nuo to paties sprendinio) [44]. Kitaip tariant, tokie sprendiniai tampa tabu ir įtraukiami į tabu sąrašą *T*. Svarbus parametras yra šio sąrašo ilgis (paskutinių nagrinėtų sprendinių, įtrauktų į tabu sąrašą, skaičius). Jeigu jis yra per mažas, gali būti gaunamas paieškos ciklinimas ar chaotiškumas. Jeigu šis parametras labai didelis – gali būti užblokuota paieška potencialiai gerose aplinkose.

Kartais (praėjus tam tikram laikui) gali būti leidžiama grįžti prie sprendinio, kuris yra tabu sąraše, tam, kad paieška galėtų būti tęsiama kitomis kryptimis. Šiuo atveju taikomas aspiracijos kriterijus [45, 46], kuriuo nusakomos tam tikros išimtinės taisyklės, kada jis gali būti taikomas, pvz., tada, kai sprendinio, prie kurio norima grįžti, tikslo funkcijos reikšmė yra mažesnė už iki šiol rastą geriausią reikšmę. Tabu sąrašas ir aspiracijos kriterijus yra dinamiškai atnaujinami vykdant algoritmą.

Tabu paieškos pagrindinio ciklo metu nustatoma: 1) ar sprendinys yra tabu; 2) ar tenkinamas aspiracijos kriterijus. Kitaip tariant, ieškoma sprendinio, tenkinančio sąlygą: $s^{\bullet} \in N^{\bullet}(s) \subseteq N(s)$; čia $N^{\bullet}(s)$ – sprendinio *s* aplinkos poaibis, kuris nepriklauso tabu sąrašui *T* ir tenkina aspiracijos sąlygą. Tabu paieška stabdoma pagal nustatytus kriterijus, tokius kaip: pasiektas fiksuotas iteracijų skaičius; tikslo funkcijos reikšmė nepagerinama tam tikrą nustatytą skaičių kartų; tikslo funkcija pasiekia iš anksto nustatytą slenksčio reikšmę [47, 48].

Dreznerio [24] pateikiamo tabu paieškos algoritmo PŠF uždaviniui spręsti pagrindinius žingsnius galima aprašyti taip:

1. Pasirenkamas pradinis sprendinys. Jis laikinai tampa geriausiu rastu sprendiniu.
2. Išvalomas tabu sąrašas.
3. Jeigu pasiekiamas nustatytas iteracijų skaičius, algoritmas stabdomas. Geriausias rastas sprendinys yra TP algoritmo rezultatas.
4. Pagal iš anksto nustatytas ribas $[T_{min}, T_{max}]$ atsitiktinai generuojamas uždraudimų valdymo kriterijus TT.

5. Vertinami visi galimi elementų sukeitimai, kurių skaičius yra lygus $m(n-m)$; čia n – elementų skaičius ($m < n$).
6. Jeigu sukeitus elementus gaunamas geresnis sprendinys už iki tol geriausią rastą sprendinį, šie elementai sukeičiami, įsimenamas naujas geriausias sprendinys ir vykdomas 3 žingsnis. Priešingu atveju vykdomas 7 žingsnis.
7. Atliekamas geriausias elementų sukeitimas (nesvarbu, ar toks sprendinys bus geresnis už iki šiol rastą geriausią sprendinį), išskyrus tuos atvejus, kai įtraukiamo elemento uždraudimų valdymo parametro reikšmė tabu sąraše yra mažesnė už nustatytą kriterijų TT. Į tabu sąrašą įtraukiamas elementas, kuris buvo pašalintas iš sprendinio. Atliekamas 3 žingsnis.

Aprašytas algoritmas buvo taikomas sprendžiant $n = 256$ dydžio uždavinį [24]. Eksperimentais buvo nustatyta, kad geresnius rezultatus galima gauti tada, kai parametrai $T_{min} = 0{,}2(n-m)$ ir $T_{max} = 0{,}2(n-m)$. Eksperimentai buvo atlikti taikant $N_{iter} = 2000n$ ir $N_{iter} = 10000n$; čia N_{iter} – iteracijų skaičius. Pastebėta, kad didinant iteracijų skaičių gaunami geresnės kokybės rezultatai. Tais atvejais, kai $m \leq 64$ ir $m \geq 95$, uždavinį išspręsti pavyko nesunkiai. Tačiau kai $65 \leq m \leq 94$, TP algoritmas neparodė gerų rezultatų. Kai $22 \leq m \leq 112$, GŽR pavyko pasiekti 55,1 proc. kartų, esant $N_{iter} = 2000n$, ir 62,6 proc. kartų, kai $N_{iter} = 10000n$. Vidutinis TP algoritmo vykdymo laikas, kai $N_{iter} = 2000n$ ir $N_{iter} = 10000n$, buvo atitinkamai 0,46 min ir 2,12 min. GŽR buvo rasta su 41 m reikšme.

Taillardas [25] aprašo stabiliojo veikimo tabu paiešką (angl. *robust taboo search*). Šiuo atveju vykdoma paprasta lokalioji paieška (angl. *local (or neighbourhood) search*), tačiau gautas sprendinys turi tenkinti papildomas sąlygas. Taip pat nustatomi du parametrai t (ilgalaikė atmintis) ir u (trumpalaikė atmintis), kurie padeda algoritmui nenagrinėti kas kartą jau nagrinėtų sprendinių. Algoritmas buvo išbandytas su $n = 64$ dydžio PŠF uždaviniu, kai $m = 13$. Per eksperimentą gautas santykinis nuokrypis nuo GŽR buvo 0,439 proc.

Kitas literatūroje randamas metodas yra reaktyvioji (reaguojanti) tabu paieška (angl. *reactive tabu search*), aprašyta Battiti ir Tecchiolli [26]. Šis metodas paremtas paprastosios tabu paieškos principais. Pradinis sprendinys, kaip ir prieš tai aprašytu atveju, pasirenkamas atsitiktinai, tačiau diversifikavimo mechanizmas skiriasi: jeigu duotieji sprendiniai yra dažnai pasirenkami (aplankomi), tada atsitiktinai pašalinama keletas narių, atmintis išvaloma. Šiuo atveju santykinis nuokrypis nuo GŽR sudarė 0,335 proc., kai $n = 64, m = 13$.

Apribotoji tabu paieška (angl. *strict tabu search*) [26] labiausiai atitinka standartinę tabu paiešką, kurią vykdant visi sprendiniai, aplankyti iteracijos k metu, yra leidžiami. Taikant apribotąją tabu paiešką santykinis nuokrypis nuo GŽR sudarė 0,703 proc. ($n = 64, m = 13$).

Talbi, Hafidi ir Geibo [27] aprašomas lygiagrečiosios adaptyviosios tabu paieškos algoritmas (angl. *parallel adaptive tabu search algorithm*) paremtas pagrindine idėja, kad keletas nepriklausomų (savarankiškai vykdomų) tabu paieškos algoritmų veikia lygiagrečiai. Šiuo atveju nereikalingas komunikavimas tarp nuoseklių užduočių. Publikacijoje [27] aprašyti eksperimentai su $n = 64$ ir $n = 256$ dydžio uždaviniais. GŽR pavyko pagerinti tris kartus, kai $m = 64, 86, 92$, $n = 256$.

Kitais atvejais, kai $m = 13$ ($n = 64$) ir kai $m = 98,128$ (n = 256), buvo pasiektos GŽR.

1.2.6. Hibridiniai genetiniai algoritmai

Pagrindiniai genetinių algoritmų (GA) veikimo principai pirmą kartą buvo aprašyti mokslininko Hollando [48]. Genetiniai algoritmai priskiriami prie evoliucinių skaičiavimo metodų klasės (angl. *evolutionary computation*). GA veikimas pagrįstas procesų, vykstančių natūraliojoje gamtoje, imitavimu – natūraliąja atranka ir evoliucija [49–54]. GA sprendimo būdas nepriklauso nuo sprendžiamo uždavinio, todėl šie algoritmai gali būti taikomi pradedant elementariomis matematinėmis funkcijomis ir baigiant sudėtingomis sistemomis.

Genetiniai algoritmai yra labai lankstūs. Tai leidžia gana nesunkiai juos modifikuoti ir taip sukurti naujus variantus. Vis dėlto šių algoritmų efektyvumas priklauso nuo sprendžiamo uždavinio. Galima išskirti pagrindinius GA trūkumus: priešlaikinį konvergavimą (angl. *premature convergence*) ir paieškos stagnacijos fenomeną (angl. *stalled evolution* (*stagnation*). Tačiau šiuos trūkumus gali atsverti teigiamos šio algoritmo savybės: gebėjimas atkartoti natūralius gamtoje vykstančius procesus ir operavimas sprendinių rinkiniu (populiacijomis). Dėl pastarosios savybės yra galimas savotiškas bendras individų būrio judėjimas hipotetinėje paieškos erdvėje [55].

Modifikuoti genetiniai algoritmai su papildomomis integruotomis optimizavimo procedūromis vadinami hibridiniais genetiniais algoritmais[2] (angl. *hybrid genetic algorithms*) (HGA).

Sudarant tokio tipo algoritmus galima remtis šiomis strategijomis:

- nuoseklioji hibridizacija (angl. *sequential (relay) hybridization*) – algoritmai vykdomi vienas po kito;
- įdėtinė hibridizacija (angl. *embedded (teamwork) hybridization*) – algoritmai veikia kaip kooperuojantys agentai (angl. *cooperating agents*) [58].

Priežastys, skatinančios kurti hibridinius algoritmus, gali būti šios: siekiama pagerinti algoritmo našumą ir sprendinių kokybę; evoliucinis algoritmas prijungiamas kaip didesnės sistemos dalis [59].

Vykdant HGA sprendiniai gali būti optimizuojami taikant įvairias optimizavimo procedūras (lokaliojo sprendinių pagerinimo procedūras). Optimizavimas gali būti atliekamas įvairiuose GA etapuose, pvz., optimizavimo procedūros taikomos jau pirminės populiacijos nariams; sprendiniai pagerinami po kryžminimo procedūros (tai gana dažnai taikoma strategija); optimizavimas gali būti atliekamas ir po mutacijos procedūros. Jeigu iš anksto optimizuojama pradinė populiacija, tada genetinis algoritmas gali operuoti jau su pagerinta populiacija. Dar vienas būdas yra sprendinių optimizavimas po atrankos operacijos, t. y. atrenkami du individai (tėvai), ir individas su geresne tikslo funkcijos reikšme optimizuojamas; toliau genetinis algoritmas tęsiamas įprasta eiga [60].

[2] Hibridiniai genetiniai algoritmai kartais dar yra vadinami memetiniais algoritmais (angl. *memetic algorithms*) [56, 57].

Fleurentas ir Fernandas [30] aprašė genetinį tabu paieškos hibridinį algoritmą (angl. *genetic-taboo search hybrid*, *GTSH*) (GTPH). Šis algoritmas – tai genetinis algoritmas, naudojantis specialų kryžminimo operatorių [61] ir nenaudojantis standartinio mutacijos operatoriaus. Vietoj sprendinių mutacijos procedūros taikomos dvi euristikos – lokalioji paieška ir tabu paieška. Siekiant pagerinti sprendinių kokybę, kiekvienam pradinės populiacijos nariui taikomas euristinis algoritmas. Taikant lokaliosios paieškos algoritmą, vykdomas vienas pagrindinis ciklas tol, kol randamas lokaliai optimalus sprendinys. Ieškomas sprendinys. GTPH buvo taikomas sprendžiant PŠF uždavinį, kai $n = 64$, $m = 13$. Algoritmas vykdytas 30 kartų ir visais atvejais pavyko pasiekti GŽR.

Modifikuotą GTPH versiją, genetinį nusileidimo hibridinį metodą (angl. *genetic-descent hybrid method*, *GDH*) (GNH) aprašė Taillardas ir Gambardella [28]. Pagrindinis skirtumas tarp šių dviejų algoritmų yra tas, kad GNH algoritme vietoj tabu paieškos buvo taikomas GSEI algoritmas (žr. 1.1.4 skyrelį).

Misevičius[3] [20] pasiūlė pagerintąjį genetinį evoliucinį algoritmą (angl. *improved genetic-evolutionary algorithm*, *IGEA*) (PGEA). Šis algoritmas turi standartinius genetinius operatorius, tačiau kryžminimo (rekombinavimo) operatorius yra specifinis. Algoritmas paremtas dviem susietais procesais: interevoliucija, t. y. globaliuoju genetiniu evoliuciniu procesu, ir intraevoliucija – lokaliuoju paieškos procesu. Interevoliucija leidžia išlaikyti paieškos krypčių įvairumą, o intraevoliucija atlieka sprendinio pagerinimo funkciją.

PGEA pagrindiniai žingsniai:

1. Atsitiktinai sugeneruojama PD dydžio pradinė populiacija; čia PD – pradinis parametras, nurodantis populiacijos dydį.
2. Taikant intraevoliucinį algoritmą, optimizuojami pradinės populiacijos P nariai.
3. Randamas ir išsaugomas geriausias populiacijos sprendinys $s^* = \arg\min_{s \in P} f(s)$; čia P – pradinė populiacija, f – tikslo funkcija.
4. Vykdoma sprendinių-tėvų atranka $s', s'' \in P$.
5. Atlikus rekombinavimo procedūrą, gaunamas palikuonis s'''.
6. Taikant intraevoliucinį algoritmą, optimizuojamas sprendinys s'''.
7. Atnaujinama populiacija P, įtraukiant į ją palikuonį s'''.
8. Tikrinama sąlyga $f(s''') < f(s^*)$; jeigu sąlyga tenkinama, tada $s^* = s'''$ (įsimenamas geriausias sprendinys).
9. Jeigu sprendiniai nėra pagerinami ET kartų (ET – nustatytas tuščiosios eigos generacijų skaičius), visiems populiacijos nariams taikoma mutacijos procedūra.
10. Jeigu pasiektas nustatytas generacijų skaičius, algoritmas baigiamas. Kitu atveju žingsniai kartojami, pradedant 4 žingsniu.

PGEA Misevičiaus [20] buvo išbandytas sprendžiant $n = 256$ dydžio PŠF uždavinį. GŽR pavyko surasti visais m atvejais.

Literatūroje taip pat aprašomi hibridiniai genetiniai algoritmai, kuriuose taikomos skirtingos rekombinavimo procedūros [52, 62–69]. Misevičius ir

[3] Išplėstinius eksperimentus su PGEA galima rasti straipsniuose [18, 19].

Rubliauskas [70] aprašo daugelio tėvų kryžminimą (angl. *multiple parent crossover*), kai atrenkami ne du (kaip standartiniame genetiniame algoritme), o nustatytas skaičius tėvų. Pritaikius šį kryžminimo operatorių gaunamas daugiau nei vienas palikuonis. Visi jie įtraukiami į populiaciją. Atitinkamai iš populiacijos pašalinami blogiausi nariai.

Kitas kryžminimo operatorius, vadinamasis tolygusis kryžminimas (angl. *uniform like crossover*, *ULX*), yra aprašytas Tate'o ir Smith [61]. Atsitiktinai pasirenkama vieta (taškas) π, nuo kurios bus atliekamas kryžminimas (sukeitimas). Pirma, visi elementai, kurie yra tose pačiose pozicijose abiejuose tėvuose, nukopijuojami į palikuonį. Antra, nepriskirtos perstatymo pozicijos nagrinėjamos iš kairės į dešinę: į nepriskirtą poziciją (nuo 1 iki π) atsitiktinai renkamas elementas iš tėvų (jeigu jis dar nėra palikuonio eilutėje). Likę nepriskirti elementai užpildomi atsitiktiniu būdu.

Viena iš tolygiojo kryžminimo operatoriaus modifikacijų yra bloko kryžminimas (angl. *block crossover, BX*) [63]. Šiam kryžminimui vietoj atskirų elementų naudojami elementų blokai (segmentai). Bloko dydis pasirenkamas intervale $[1, \left\lfloor \frac{n}{2} \right\rfloor]$. Kopijuojant blokus, turi būti atsižvelgiama į tai, jog perstatymo elementai negali kartotis.

1.3. Tikslieji algoritmai

Nagrinėjamam uždaviniui spręsti Drezneris [24] pasiūlė šakų ir ribų algoritmą (angl. *branch and bound algorithm*) (ŠRA). Turima simetriška atstumų d_{ij} tarp elementų i ir j $(d_{ij} = d_{ji})$ $n \times n$ dydžio matrica su nuliais užpildyta įstrižaine. Tarkime, jog pasirenkama m indeksų $p_1, p_2, \dots, p_m$ susietajam branduoliui (klasteriui) formuoti. Pusė tikslo funkcijos reikšmių gali būti apskaičiuojamos susumavus atstumus stulpeliuose $p_1, p_2, \dots, p_m$, kurių eilutės priklauso žemesniems p_j. Visi šie atstumai bus viršutiniame matricos trikampyje. Visos galimos m stulpelių iš n kombinacijos yra netiesiogai patikrinamos (įvertinamos). Apatinė riba nustatoma kiekvienam daliniam stulpelio pasirinkimui (sprendinių poaibiui). Kitaip tariant, formuojamas paieškos medis, kurio viršūnės yra sprendinių poaibiai. Renkamas pirmas stulpelis ir skaičiuojama apatinė riba, tada pasirenkamas antras stulpelis, skaičiuojama apatinė riba ir t. t. Skaičiavimai atliekami tol, kol apatinė riba yra didesnė už geriausią rastą sprendinį arba jam lygi. Tie stulpeliai, kurių apatinis rėžis patenka į nustatytą intervalą, nagrinėjami toliau, o kiti stulpeliai pašalinami. Publikacijoje [24] pateikiami eksperimentų rezultatai su $n = 256$ dydžio uždaviniu. Kai $m = 3, \dots, 8$ GŽR pavyko pasiekti, tačiau algoritmo vykdymo laikas lyginant su euristiniais algoritmais buvo gana didelis.

Drezneris, Misevičius ir Palubeckis [71] išbandė dar vieną šakų ir ribų algoritmą. Šiuo atveju jie pasiūlė kitokios (lyginant su prieš tai aprašytu algoritmu) konstrukcijos paieškos medį. Sprendinys sudaromas iš pasirinktų $a_1, a_2, \dots, a_m$ elementų. Tikslo funkcijos reikšmė apskaičiuojama susumavus atstumų reikšmes pasirinktų eilučių ir stulpelių susikirtimo taškuose. Pagrindinė algoritmo idėja ta, kad eliminuojamos (pašalinamos) paieškos medžio šakos neturinčios optimalaus sprendinio. Atlikus eksperimentus su $n = 256, 576, 1024$ dydžio uždaviniais buvo

pastebėta, kad šis šakų ir ribų algoritmas randa optimalius sprendinius per priimtiną skaičiavimų laiką tuomet, kai m reikšmės yra nedidelės.

1.4. Pirmojo skyriaus išvados

Atlikus algoritmų, taikomų PŠF uždaviniui spręsti, apžvalgą galima daryti tokias išvadas:

1. Nėra efektyvių tikslių ar apytikslių algoritmų sudėtingiems kombinatorinio optimizavimo uždaviniams (tarp jų ir šiame darbe sprendžiamam PŠF uždaviniui) spręsti, todėl tokie uždaviniai sprendžiami taikant euristinius algoritmus. Taigi tikslinga kurti (tobulinti) įvairius euristinius algoritmus (pvz., HGA) siekiant pagerinti sprendinių kokybę (rasti sprendinius, kuo artimesnius globaliajam optimumui) arba kiek įmanoma sutrumpinti vykdymo laiką.
2. Euristinių algoritmų lyginamoji analizė parodė, kad taikant šiuos algoritmus galima per priimtiną vykdymo laiką gauti geros kokybės (galimai optimalius) sprendinius, kiek galima artimesnius globaliajam optimumui.

2. HIBRIDINIS GENETINIS ALGORITMAS IR JO VARIANTAI PILKŲJŲ ŠABLONŲ FORMAVIMO UŽDAVINIUI SPRĘSTI

2.1. Kvadratinio paskirstymo uždavinys

Kvadratinio paskirstymo uždavinys 1957 metais buvo pristatytas Koopmanso ir Beckmanno kaip matematinis modelis ekonominių uždavinių kontekste [72]. Nuo to laiko jis buvo daugybės matematikos ir kompiuterių mokslo sričių tyrimų objektas. Dabar KP uždavinys traktuojamas kaip klasikinis kombinatorinio optimizavimo uždavinys, kuris vis dar įvairiapusiškai patrauklus tyrėjams. Galima išskirti keletą priežasčių, kodėl jis yra populiarus kombinatorinio optimizavimo kontekste: daugėja realaus gyvenimo uždavinių, matematiškai suformuluotų kaip KP uždaviniai; yra nemažai kitų gerai žinomų kombinatorinio optimizavimo uždavinių, kurie gali būti formuluojami kaip atskiri KP uždavinio atvejai (pvz., pilkųjų šablonų formavimo uždavinys, keliaujančio pirklio uždavinys, grafų dalijimo uždavinys ir kt.) [7].

KP uždavinys yra vienas iš sudėtingiausių kombinatorinio optimizavimo uždavinių. Apskritai, kai uždavinio dydis $n > 30$, jis neišsprendžiamas per priimtiną laiką (pvz., kvadratinio paskirstymo uždavinys išsprendžiamas tiksliai tada, kai jo apimtis (n) neviršija 36), todėl reikia taikyti įvairius euristinius algoritmus vidutinio dydžio ir dideliems KP uždaviniams spręsti [7, 9, 73–80].

Nustatytos atstumų matricos su sveikosiomis reikšmėmis $A = (a_{ij})_{n\times n}$, $B = (b_{kl})_{n\times n}$ ir perstatymų aibė Π_n, sudaryta iš sveikųjų skaičių nuo 1 iki *n*. Reikia rasti perstatymą $p = (p(1), p(2), \dots, p(n)) \in \Pi_n$ minimizuojant funkciją:

$$z(p) = \sum_{i=1}^{n} \sum_{j=1}^{n} a_{ij} b_{p(i)p(j)}. \quad (3)$$

Sprendžiant KP uždavinį, perstatymą $p = (p(1), p(2), \dots, p(n))$ galima traktuoti kaip objektų išdėstymą į tam tikras pozicijas. Matrica $A = (a_{ij})_{n\times n}$ gali būti interpretuojama kaip sujungimų tarp komponentų matrica. Matrica $B = (b_{kl})_{n\times n}$ yra atstumų matrica, kur b_{kl} – atstumai tarp *k* ir *l* pozicijų. Kiekviena paskirties vietų konfigūracija atitinka tam tikrą perstatymą $p = (p(1), p(2), \dots, p(n))$, kur $p(i)$ nurodo vietą, į kurią padedamas *i*-tasis komponentas. Funkcija *z* gali būti traktuojama kaip apskaičiuota visų susijungimų ilgių, padaugintų iš reikšmių a_{ij}, suma.

2.2. PŠF uždavinio matematinė formuluotė

PŠF uždavinio atveju matricos $A = (a_{ij})_{n\times n}$ elementai gali įgyti tik dvi reikšmes – 0 ir 1, t. y.:

$$a_{ij} = \begin{cases} 1, & 1 \le i \le m \; ir \; 1 \le j \le m \\ 0, & priešingu \; atveju \end{cases}; \quad (4)$$

čia $i, j = 1, 2, \dots, n, m \le n$.

Matrica *A* sprendžiant PŠF uždavinį tampa nereikalinga, užtenka žinoti tik parametro *m* tikslią reikšmę. Matricos *B* reikšmės apskaičiuojamos visoms esamoms taškų poroms pagal formulę:

$$b_{kl} = b_{(r-1)n_2+t\,(u-1)n_2+v} = \omega_{rtuv}, \quad (5)$$

$$\omega_{rtuv} = \min_{w_1,w_2\in\{-1,0,1\}} \frac{1}{(r-u+w_1n_1)^2+(t-v+w_2n_2)^2}; \quad (6)$$

čia $k,l = 1,2,\dots,n, r,u = 1,2,\dots,n_1, t,v = 1,2,\dots,n_2, n_1 \times n_2 = n$.

Fizikinė (dydžio) ω_{rtuv} interpretacija būtų dydis, proporcingas stiprumui atstūmimo jėgos, veikiančios tarp dalelių *i* ir *j*, esančių pozicijose $p(i)$ ir $p(j)$. Pozicijų k ir l koordinatės yra atitinkamai (r,t) ir (u,v). PŠF uždavinio rezultatu laikomas rastas perstatymas *p*. Norint rasti galimai optimalų perstatymą (perstatymo elementus $p(1),p(2),\dots,p(m)$, kur $1 \le p(i) \le n$), reikia rasti minimalią tikslo funkcijos reikšmę $z(p)$ ((7) formulė).

Tokiu būdu, sprendžiant PŠF uždavinį tikslas yra surasti perstatymo elementus $p(1),\dots,p(m)$ $(1 \le p(i) \le n, i = 1,\dots,m)$ ir tokius, kad būtų minimizuojama ši tikslo funkcija:

$$z(p) = \sum_{i=1}^{m}\sum_{j=1}^{m} b_{p(i)p(j)}. \quad (7)$$

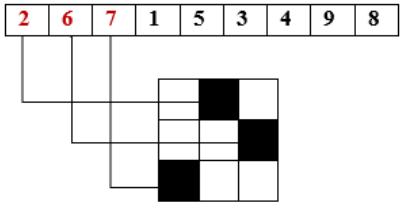
Rasto perstatymo *p* *i*-tasis elementas $p(i) = (r-1)n_2 + t$ rodo, kurioje vietoje turi būti padėtas juodas (spalvotas) kvadratėlis. Eilutės ir stulpelio indeksai *r* ir *t* apskaičiuojami pagal formules:

$$r = \lfloor (p(i)-1)/n_2 \rfloor + 1, \quad (8)$$

$$t = \big((p(i)-1) mod\ n_2\big) + 1; \quad (9)$$

čia $i \le m$, *mod* žymi liekaną, gautą padalijus vieną skaičių iš kito.

Pavyzdžiui, jeigu turime galimai optimalų sprendinį $p = (2,6,7,1,5,3,4,9,8)$, $n_1 = 3, n_2 = 3, n = 9, m = 3$, tada sudėliotos dalelės (taškai) atrodytų taip (3 pav.):



3 pav. Perstatymo ir grafinio tinklelio pozicijų sąsaja

Paprastai n_1 ir n_2 reikšmės būna fiksuotos ir vienodo dydžio, o parametro *m* reikšmė yra kintanti. Esant fiksuotoms parametrų n_1 ir n_2 reikšmėms, PŠF uždavinį pakanka spręsti tik $n/2$ kartų, nes sprendiniai nuo $\frac{n}{2}+1$ iki *n* gali būti gaunami invertuojant šablono fono ir pagrindinę spalvas.

2.3. Baziniai apibrėžimai

Šiame poskyryje pateikiami pagrindiniai apibrėžimai.

1 apibrėžimas. Aplinkos funkcija Θ: $\Pi_n \to 2^{\Pi_n}$ priskiria kiekvienam $p \in \Pi_n$ rinkinį $\Theta(p) \subseteq \Pi_n$, kur $\Theta(p)$ yra p sprendinio kaimyninių sprendinių rinkinys. Sprendžiant perstatymu grįstus uždavinius įprasta taikyti aplinkos funkciją Θ_2, kurią

galima apibrėžti taip: $\Theta_2(p) = \{p' : p' \in \Pi_n, \delta_H(p, p') = 2\}$; čia $\delta_H(p, p')$ yra Hamingo atstumas[4] (angl. *Hamming distance*) tarp perstatymų p ir p'.

Pažymėtina, kad $p(1), \dots, p(m)$ elementų išdėstymo tvarka nėra svarbi PŠF uždaviniui, todėl reikia suformuluoti aplinkos nagrinėjimo (paieškos) funkciją tinkamu būdu.

2 apibrėžimas. Funkcija Θ_1 suformuluota taip, kad kiekvienas naujas sprendinys $p' \in \Theta_1(p)$ yra gaunamas iš esamo sprendinio p tiesiog sukeičiant vieną elementą iš $\{p(i) : i = 1, \dots, m\}$ su kitu elementu iš $\{p(j) : j = m+1, \dots, n\}$.

Akivaizdu, kad ši funkcija užtikrina sprendinių tinkamumą, t. y. $\forall p \in \Pi_n$: $p' \in \Theta_1(p) \Longrightarrow p' \in \Pi_n$. Formaliai:

$$\Theta_1(p) = \{p' : p' \in \Pi_n, \delta(p, p') = 1\}; \tag{10}$$

čia δ – atstumas tarp dviejų sprendinių.

3 apibrėžimas. Atstumas tarp dviejų sprendinių p_1 ir p_2 apskaičiuojamas:

$$\delta(p_1, p_2) = m - |\{p_1(i) : i = 1, \dots, m\} \cap \{p_2(i) : i = 1, \dots, m\}|; \tag{11}$$

čia $0 \le \delta \le m$, $\delta(p, p) = 0$, $\delta(p_1, p_2) = \delta(p_2, p_1)$.

Detaliau, tarkime, $p(v)$ $(u = 1, \dots, m)$ ir $p(w)$ $(v = m+1, \dots, n)$ yra du elementai, kurie bus sukeičiami. Tada $p^{v,w}$ galima apibrėžti taip: $p^{v,w}(i) = \begin{cases} p(i), & i \neq v, w \\ p(v), & i = w \\ p(w), & i = v \end{cases}$. Vadinasi, $p^{v,w}$ gaunamas iš sprendinio p sukeičiant $p(v)$ ir $p(w)$ elementus vietomis. Žinoma, kad $\delta(p, p^{v,w}) = 1$, $p^{v,v} = p$, $(p^{v,w})^{v,w} = p$.

4 apibrėžimas. Tikslo funkcijos pokytis sukeitus $p(v)$ ir $p(w)$ reikšmes apskaičiuojamas taip:

$$\Delta(p^{v,w}, p) = z(p^{v,w}) - z(p) = 2\Big(c\big(p(w)\big) - c\big(p(v)\big) - B\big(p(v), p(w)\big)\Big); \tag{12}$$

čia $c(x)$ yra elemento x susijusių atstumų suma ((13) formulė).

5 apibrėžimas. Susijusių atstumų sumų masyvo c reikšmės apskaičiuojamos pagal formulę:

$$c(x) = \sum_{y=1}^{m} b_{xp(y)}; \tag{13}$$

čia $x = 1, \dots, n$.

6 apibrėžimas. Masyvo c reikšmės perskaičiuojamos sukeitus elementus pagal formulę:

$$c(x) = \begin{cases} c(x) + B\big(x, p(v)\big), & x = p(w) \\ c(x) - B\big(x, p(w)\big), & x = p(v) \\ c(x) + B\big(x, p(v)\big) - B\big(x, p(w)\big), & x \neq p(v), x \neq p(w) \end{cases} \tag{14}$$

čia $x = 1, \dots, n$.

[4] Pažymėtina, kad Hamingo atstumas tarp dviejų perstatymų p_1 ir p_2 gali būti apibrėžiamas kaip $\delta_H(p_1, p_2) = |\{i : p_1(i) \neq p_2(i)\}|$.

Rasto sprendinio (perstatymo) elementai „sudėliojami“ remiantis (8) ir (9) formulėmis (žr. 2.2.1 skyrelį).

Galima keisti ir po kelis elementus, tada funkcija $\Theta_{2\oplus 2}$ (keičiami du elementai su dviem) bus apskaičiuojama [81]:

$$\Theta_{2\oplus 2}(p) = \left\{ p^{\nabla\nabla} \middle| \begin{matrix} p^{\nabla\nabla} = p^{ijrt}, p^{\nabla\nabla} \in \Pi_n, i = 1, \dots, m, j = m+1, \dots, n, \\ r = (i \bmod m) + 1, t = \max((j \bmod n) + 1, m+1) \end{matrix} \right\}; \quad (15)$$

čia $p^{ijrt}(k) = \begin{cases} p(k), k \neq i, j, r, t \\ p(i), k = j \\ p(j), k = i \\ p(r), k = t \\ p(t), k = r \end{cases}$, $k = 1, \dots, n$.

Tada $\Theta_{2\oplus 2\oplus 2}$:

$$\Theta_{2\oplus 2\oplus 2}(p) = \left\{ p^{\nabla\nabla\nabla} \middle| \begin{matrix} p^{\nabla\nabla\nabla} = p^{ijrtuv}, p^{\nabla\nabla\nabla} \in \Pi_n, i = 1, \dots, m, j = m+1, \dots, n, \\ r = (i \bmod m) + 1, t = \max((j \bmod n) + 1, m+1), \\ u = (r \bmod m) + 1, v = \max((t \bmod n) + 1, m+1) \end{matrix} \right\}; \quad (16)$$

čia $p^{ijrtuv}(k) = \begin{cases} p(k), k \neq i, j, r, t, u, v \\ p(i), k = j \\ p(j), k = i \\ p(r), k = t \\ p(t), k = r \\ p(u), k = v \\ p(v), k = u \end{cases}$, $k = 1, \dots, n$.

Kai taikoma $\Theta_{2\oplus 2}$ funkcija, tikslo funkcijos reikšmės pokytis apskaičiuojamas:

$$\Delta z\left(p, p^{ijrt}\right) = 2 \begin{pmatrix} c_{p(j)} - c_{p(i)} + c_{p(t)} - c_{p(r)} + b_{p(i)p(r)} + b_{p(j)p(t)} - \\ b_{p(i)p(j)} - b_{p(i)p(t)} - b_{p(r)p(j)} - b_{p(r)p(t)} \end{pmatrix}. \quad (17)$$

Atitinkamai, taikant $\Theta_{2\oplus 2\oplus 2}$ funkciją, $\Delta z(p, p^{ijrtuv})$ apskaičiuojamas:

$$\Delta z\left(p, p^{ijrtuv}\right) = 2 \begin{pmatrix} c_{p(j)} - c_{p(i)} + c_{p(t)} - c_{p(r)} + c_{p(v)} - c_{p(u)} + \\ b_{p(i)p(r)} + b_{p(i)p(u)} + b_{p(r)p(u)} + b_{p(j)p(t)} + b_{p(j)p(v)} + \\ b_{p(t)p(v)} - b_{p(i)p(j)} - b_{p(i)p(t)} - b_{p(i)p(v)} - b_{p(r)p(j)} - \\ b_{p(r)p(t)} - b_{p(r)p(v)} - b_{p(u)p(j)} - b_{p(u)p(t)} - b_{p(u)p(v)} \end{pmatrix}. \quad (18)$$

PŠF uždaviniui spręsti bus naudojamas ir priešingumu pagrįstas sprendinys (kitaip tariant, esamo sprendinio papildinys) (angl. *opposition-based solution*), kuris siejamas su priešingumu paremtu mokymu (angl. *opposition-based learning*) [82]. Šio metodo loginis pagrindas yra prielaida, kad naudingiau apsvarstyti papildytą sprendinį atsižvelgiant į esamą sprendinį paieškos erdvėje nei pasirinkti naują atsitiktinai

sugeneruotą sprendinį. Priešingo (opozicinio) sprendinio naudingumas buvo nustatytas sprendžiant maksimalaus įvairumo uždavinį [15], kurį būtų galima pavadinti PŠF uždavinio „dvyniu“.

7 apibrėžimas. PŠF uždavinio sprendinys $\overline{p} \in \Pi_n$ yra sprendiniui p priešingas (opozicinis) sprendinys, jeigu $(p, \overline{p}) = m$.

Galiausiai apibrėžiamas pagrindo atrinkimu paremtas sprendinys (angl. *backbone solution*) [15].

8 apibrėžimas. PŠF uždavinio sprendinys $p^{*} \in \Pi_n$ yra pagrindo atrinkimu paremtas sprendinys (atsižvelgiant į du pagrindinius sprendinius p_1, p_2), jeigu tuo pačiu metu $\delta(p^{*}, p_1) \leq \lceil m/2 \rceil$ ir $\delta(p^{*}, p_2) \leq \lceil m/2 \rceil$.

Galima sakyti, kad pagrindo atrinkimu paremtas sprendinys dalijasi informacija su abiem sprendiniais (p_1 ir p_2) ir kartu išlieka gana artimas jiems abiem.

2.4. Hibridinis genetinis algoritmas PŠF uždaviniui

Pristatomas algoritmas [21] sukonstruotas genetinio algoritmo struktūros pagrindu: populiacija operuojanti evoliucinė paieška yra sujungta su lokaliu palikuonių pagerinimu siekiant padidinti paieškos efektyvumą ir kokybę.

Šį hibridinį genetinį algoritmą (HGA) sudaro šešios pagrindinės dalys: a) pradinės populiacijos konstravimas; b) tėvų atranka; c) kryžminimo operatorius; d) gautų palikuonių pagerinimas; e) populiacijos atnaujinimas; f) naujos populiacijos generavimas (jeigu to reikia). HGA pseudokodas[5] pateikiamas 4 pav.

[5] Čia pseudokodas (angl. *pseudocode*) – algoritmo atvaizdavimo forma, kai algoritmas pateikiamas taikant aukšto abstrakcijos lygio sintaksės formalizuotas taisykles.

```
procedure Hibridinis_genetinis_algoritmas
// duomenys: n, m, B
// rezultatas: s* – galimai optimalus sprendinys
// parametrai[6]: PS – populiacijos dydis, N_gen – generacijų skaičius, L_idle_gen – tuščiosios eigos generacijų skaičius,
// DT – atstumo slenkstis
begin
  nustatyti pirminius parametrus; inicijuoti algoritmo kintamuosius;
  sukonstruoti pradinę populiaciją P (PS dydžio); // žiūrėti 2.4.1 skyrelį
  p* := argmin_{p∈P}{z(p)}; // / išsaugomas geriausias sprendinys
  for gen_index := 1 to N_gen do begin
    atsitiktinai atrinkti tėvus p', p'' ∈ P;
    generuoti palikuonius p°, p°° ∈ Π_n;
    taikyti Hierarchinė_iteratyvioji_tabu_paieška palikuoniui p°, gauti
pagerintą sprendinį p☆;
    if z(p☆) < z(p*) then p* := z(p☆); // įsimenamas geriausias sprendinys
    atnaujinti populiaciją P;
    taikyti Hierarchinė_iteratyvioji_tabu_paieška palikuoniui p°°, gauti
pagerintą sprendinį p☆;
    if z(p☆) < z(p*) then p* := z(p☆); // įsimenamas geriausias sprendinys
    atnaujinti populiaciją P;
    if L_idle_gen pasiektas then begin
       sugeneruoti populiaciją iš naujo;
       if min_{p∈P}{z(p)} < z(p*) then p* := argmin_{p∈P}{z(p)}
    endif
  endfor
end.
```

4 pav. Hibridinio genetinio algoritmo PŠF uždaviniui pseudokodas

Algoritmas pradedamas fiksuotojo dydžio (*PS*) pradinės populiacijos kūrimu atsižvelgiant į populiacijos narių kokybę (tinkamumą) ir jų panašumą (atstumą). Algoritmas vykdomas tam tikrą skaičių generacijų (N_{gen}). Kiekvienos generacijos metu atliekamos standartinės genetinio algoritmo procedūros (išskyrus mutaciją) – atranka, kryžminimas, populiacijos atnaujinimas. Mutacijos procedūra yra integruota į palikuonių pagerinimo algoritmą – hierarchinę iteratyviąją tabu paiešką (angl. *hierarchical iterated tabu search*) (HITP). Siekiant išvengti evoliucinio proceso stagnacijos, algoritme inkorporuota populiacijos pertvarkymo (angl. *restart*) procedūra.

Kiekviena HGA dalis išsamiau aprašoma tolesniuose skyreliuose.

2.4.1. Pradinės populiacijos konstravimas

Pradinė populiacija konstruojama taip:

1. Nustatomi parametrai $flag = $'OFF', $P = \emptyset$, $k = 1$, $l = 1$ (l – sugeneruoto sprendinio leksikografinis indeksas).

2. Jeigu $flag = $'OFF', tada generuojamas atsitiktinis perstatymas (sprendinys) p_1; priešingu atveju generuojamas papildymu paremtas atsitiktinis sprendinys p_l. $l = l + 1$.

[6] HGA parametrų pavadinimų santrumpos yra tokios kaip ir programos tekste.

3. Sugeneruotam sprendiniui taikoma hierarchinė iteratyvioji tabu paieška ir gaunamas naujas pagerintas sprendinys p_l^*.

4. Jeigu $(flag = \text{'OFF'})$ ir $(k = 1)$, tada: a) p_1^* sprendinys įtraukiamas į populiaciją P; b) $flag = \text{'ON'}$; c) grįžtama į 2 žingsnį.

5. Jeigu $(flag = \text{'ON'}), (k = 1)$ ir $(z(p_l^*) < z(p): p \in P)$, tada: a) pirmas populiacijos narys pakeičiamas p_l^* sprendiniu; b) grįžtama į 2 žingsnį.

6. Jeigu $\left((z(p_l^*) \neq z(p): \forall p \in P) \text{ ir } \left(\min_{p \in P}\{\delta(p_l^*, p)\} \geq DT\right)\right)$ arba $\left(z(p_l^*) < \min_{p \in P}\{z(p)\}\right)$, tada sprendinys p_l^* įtraukiamas į populiaciją. Kitu atveju į populiaciją įtraukiamas atsitiktinis sprendinys p_l.

7. $k = k + 1$. Jeigu $k \leq PS$, tada grįžtama į 2 žingsnį. Kitu atveju pradinės populiacijos konstravimas (generavimas) stabdomas.

Kalbant apie priešingų (opozicinių) sprendinių generavimą, tinkamas būdas tai atlikti yra ilgalaikio dažnių masyvo f sukūrimas. Šiuo atveju $f(i)$ saugo elemento i pasikartojimų sprendinyje skaičių. Masyvas f kuriamas labai paprastai, pradžioje užpildomas nulinėmis reikšmėmis ir atnaujinamas kaskart, kai sugeneruojamas naujas sprendinys, t. y. $f\big(p_l(i)\big) = f\big(p_l(i)\big) + 1$; čia p_l – šiuo metu sugeneruotas sprendinys. Norint gauti papildymu paremtą sprendinį, pakanka pasirinkti m elementų, kurie pasikartoja mažiausiai. Tada $\delta(p_l, p_{l-1}) = m$ $(l = 2, 3, \ldots)$; p_l, p_{l-1} – nuosekliai vienas po kito sugeneruoti sprendiniai.

Kiekvienas sugeneruotas sprendinys yra gerinamas taikant HITP. Tada tikrinama, ar atstumas tarp pagerinto sprendinio p_l^* ir populiacijos P $\delta(p_l^*, P) = \min_{p \in P}\{\delta(p_l^*, p)\}$ yra didesnis už iš anksto nustatytą slenkstį DT arba yra jam lygus. Jeigu ši sąlyga tenkinama, pagerintas sprendinys įtraukiamas į populiaciją. Taip pat yra ir su sprendiniu, kuris geresnis už geriausią turimą sprendinį. Kitu atveju populiaciją papildo atsitiktinai sugeneruoti sprendiniai. Toks pradinės populiacijos generavimo metodas padeda išlaikyti ir sprendinių kokybę, ir jų įvairovę.

2.4.2. Tėvų atranka

Kiekvienos generacijos metu iš populiacijos P parenkami du perstatymai (sprendiniai) p' ir p''. Toliau vykdant algoritmą šie perstatymai naudojami kaip tėvai naujiems palikuoniams generuoti.

2.4.3. Kryžminimas

Kryžminimo (rekombinavimo) tikslas yra iš atrinktų tėvų poros sukurti naują palikuonį. Pagrindinis taikomo kryžminimo principas yra sprendinio pagrindo suformavimas ir priešingo (opozicinio) sprendinio radimas [15, 21]. Tai leidžia išsaugoti bendrus elementus (genus), kuriuos turi abu atrinkti tėvai, ir sprendinį papildyti naujais genais. Sprendinio pagrindas yra iš dalies optimizuotas siekiant geros palikuonių kokybės. Tam pasitelkta godžioji adaptyvioji procedūra (angl. *greedy adaptive procedure*) (GAP). Sugeneruojami du palikuonių sprendiniai: optimizuotas palikuonis ir jo papildinys (priešingo sprendinio palikuonis).

Laikinojo masyvo (angl. *short-term array*) f_{ST}^{cross}, saugančio genų pasikartojimo dažnumą, reikšmės apskaičiuojamos pagal formulę:

$$f_{ST}^{cross}(i) = \left|\{i: i \in \{p'(k): k = 1, \dots, m\}, i \in \{p''(k): k = 1, \dots, m\}\}\right|; \quad (19)$$

čia $i = 1, \dots n$, p', p'' – atitinkami tėvų sprendiniai.

Genai (m skaičius), kurie dažniausiai pasikartoja, atrenkami sprendinio pagrindui $p^{\divideontimes}$ formuoti. Tada daliai (m/2) genų, pasižyminčių didžiausiu pasikartojimo dažnumu, taikoma GAP, kuri kaip pradinius duomenis gauna dalinį sprendinį $p^{\divideontimes}$ (elementus $p^{\divideontimes}(1)$, ..., $p^{\divideontimes}(m/2)$). GAP renkasi po vieną elementą ir prideda jį prie esamo dalinio sprendinio. Kiekvienos iteracijos q metu ($q = 1, \dots, m/2$) GAP prideda elementą iš dar nepasirinktų elementų $\{j: j = 1, \dots, n\} \setminus \{p(i): i = 1, \dots, m/2 + q - 1\}$, pasižyminčių mažiausia atstumų suma (žr. (13) formulę), t. y. $j = \underset{j \in \{j: j=1,\dots,n\} \setminus \{p(i): i=1,\dots,m/2+q-1\}}{\operatorname{argmin}} \{c(p(j))\}$. Tęsiama tol, kol suformuojamas sprendinys. Tikslo funkcijos reikšmė z gali būti gaunama iš c reikšmių pagal formulę:

$$z = 2\sum_{j=1}^{m} c(p(j)). \quad (20)$$

Šios kryžminimo procedūros (vadinamos pagrindo atrinkimu ir priešingumu pagrįstu (opoziciniu) kryžminimu (angl. *backbone and opposition-based crossover*)) pseudokodas pateikiamas 5 pav.

```
procedure Pagrindo_atrinkimu_ir_priešingumu_pagrįstas_kryžminimas
// duomenys: n, m
//           p', p'' – tėvai
// rezultatas: p° – iš dalies pagerintas sprendinys, p°° – priešingas (opozicinis) sprendinys

begin
  // / sprendinio pagrindo konstravimas atsižvelgiant į sprendinius (tėvus) p', p'';
  apskaičiuoti laikinojo masyvo reikšmes (dažnumus) f_ST^cross(i), //žr. (20) formulę
  parinkti m genų su didžiausiu dažnumu,
  iš jų suformuoti sprendinio pagrindą p^*;
  // dalinis sprendinio pagrindo optimizavimas
  ignoruoti m/2 genų iš sprendinio p^*
  taikyti Godžioji_adaptyvioji_procedūra daliniam sprendiniui p^*,
  gauti dalinį optimizuotą palikuonio sprendinį p°;
  // priešingo (opozicinio) sprendinio konstravimas
  generuoti priešingą palikuonio sprendinį p°° atsižvelgiant į p°, t.y. p°° = p°‾
end.
```

5 pav. Kryžminimo procedūros PŠF uždaviniui pseudokodas

```
procedure Godžioji_adaptyvioji_procedūra
// duomenys: n, m, B,
//          p – dalinis sprendinys, kur elementai p(m/2 + 1), …, p(m) yra ignoruojami
// rezultatas: p – suformuotas sprendinys

begin
  for i := 1 to n do begin c(i) := 0; Selected(i) := FALSE endfor;
  for i := 1 to n do for j := 1 to m/2– 1 do c(i) := c(i) + B(i, p(j));
  for i := 1 to m/2 do Selected(p(i)) := TRUE; //Selected parametro
inicializavimas
  i := m/2; k := p(m/2);
  for q := 1 to m/2 do begin //ciklas kartojamas tol, kol suformuojamas
sprendinys
    minimum_contribution := ∞;
    for j := 1 to n do
      if Selected(j) = FALSE then begin
         c(j) := c(j) + B(j, k); if c(j) < minimum_contribution then begin
         minimum_contribution := c(j); j_min := j endif
      endif;
    i := i + 1; p(i) := j_min; // elementas su minimalia parametro minimum_contribution
                                  reikšme  įtraukiamas į sprendinį
    Selected(p(i)) := TRUE; k := j_min
  endfor;
  i := m + 1; for j := 1 to n do // priskiriamos reikšmės elementams p(m + 1),
…, p(n)
               if Selected(j) = FALSE then begin p(i) := j; i := i + 1 endif
end.
```

6 pav. Godžiosios adaptyviosios procedūros pseudokodas

GAP veikimo principas nėra naujas. Savo prigimtimi jis panašus į godžiosios randomizuotos adaptyviosios paieškos procedūrą (angl. *greedy randomized adaptive search procedure*) [83]. GAP vadinama adaptyviąja todėl, kad ji renkasi esamą elementą, atsižvelgdama į jau pasirinktus elementus, taip pat pasirinktų elementų rinkinys atnaujinamas kiekvienos iteracijos metu. GAP vadinama godžiąja todėl, kad visada pasirenkami elementai, turintys minimalią įmanomą atstumų reikšmę c. GAP neturi atsitiktinumų.

Priešingam sprendiniui generuoti naudojamas ilgalaikis dažnių masyvas (angl. *long-term array*) f_{LT}^{cross}. Masyvas f_{LT}^{cross} inicijuojamas prieš pradedant vykdyti genetinį algoritmą. Šio masyvo reikšmės atnaujinamos kiekvieną kartą, kai konstruojamas naujas optimizuotas pagrindo sprendinys, t. y. $f_{LT}^{cross}\left(p^{\circ}(i)\right) = f_{LT}^{cross}\left(p^{\circ}(i)\right) + 1$; p° – optimizuotas sprendinio pagrindas.

2.4.4. Hierarchinės iteratyviosios tabu paieškos algoritmas

Pristatomas algoritmas paremtas hierarchinės iteratyviosios paieškos algoritmo paradigma [84]. Algoritmo idėja yra ta, kad lokaliąja paieška paremtus algoritmus galima toliau tobulinti protingai konstruojant vidinę algoritmo struktūrą (architektūrą) ir kuriant hierarchiniu būdu struktūrizuotus (hierarchinius) algoritmus (HA). Pagrindinis hierarchinių algoritmų principas yra daugkartinis (pakartotinis) žinomų algoritmų (pvz., lokalioji paieška, tabu paieška) taikymas. Iteratyvioji tabu paieška gaunama taikant tabu paiešką ir tam tikrą sprendinių pertvarkymą. Toliau ITP

algoritmas kombinuojamas pats su savimi (su ITP algoritmu). Taip gaunamas ITP-ITP algoritmas. Šis principas gali būti kartojamas norimą skaičių kartų. Kiekvienas toks kartojimas turi tris į save panašias (angl. *self-similar*) dalis: 1) ITP procedūra; 2) kandidato pasirinkimas; 3) sprendinio kandidato pertvarkymas.

A. *Tabu paieška*

Tabu paieška atlieka pagrindinį vaidmenį ITP algoritme. Šiai procedūrai taikoma vieno elemento keitimo su vienu aplinkos elementu funkcija Θ_1. TP pradedama nuo esamo sprendinio, sukeičiamas vienas elementas iš $M = \{p(i): i = 1, \dots, m\}$ rinkinio su kitu elementu iš $N = \{p(i): i = m+1, \dots, n\}$ rinkinio taip, kad tikslo funkcijos reikšmė būtų minimizuojama, atsižvelgiant į tabu sąlygą ir aspiracijos kriterijų.

Siekiant sumažinti skaičiavimų laiką, naudojama modifikuota nagrinėjamų sprendinių aplinka Θ_1^*, kurią būtų galima apibrėžti taip[7]:

$$\Theta_1^*(p) = \{p': p' \in \{p(i): i = 1, \dots, m\} \setminus \{p(v)\} \cup \{p(w)\}, p(v) \in M', p(w) \in N'\}; \quad (21)$$

čia $v = 1, \dots, |M'|$, $w = m+1, \dots, m+|M'|$. Rinkiniai M', N' apibrėžiami taip:

$$M' = \{p(i): c(p(i)) \geq SL_1,\ i = 1, \dots, m\}, \quad (22)$$

$$N' = \{p(i): c(p(i)) \leq SL_2,\ i = m+1, \dots, n\}; \quad (23)$$

čia $SL_1 = \max\{c(p(i)): i = 1, \dots, m\} - \rho BMax$, $SL_2 = \min\{c(p(i)): i = m+1, \dots, n\} + \rho BMax$, $BMax = \max\{b_{kl}: k = 1, \dots n, l = 1, \dots n\}$, ρ ($\rho > 0$) – aplinkos apimties parametras.

Naudojamas tabu (draudimų) sąrašas (*TabuList*) – matrica, kurioje įrašas $TabuList(p(v), p(w))$ saugo esamos iteracijos numerį ir uždraudimų valdymo (*Tabu Tenure*) parametro reikšmę (parametro reikšmė $h = \lfloor 0{,}3m \rfloor$), t. y. iteracijų numerį, nuo kurio atitinkami elementai $(p(v), (p(w))$ vėl gali būti kaitaliojami. Sukeisti elementus $p(v)$ ir $p(w)$ draudžiama, jeigu $TabuList(p(v), p(w))$ reikšmė yra lygi esamam iteracijų skaičiui arba už jį didesnė. Draudimai ignoruojami, jeigu tenkinamas aspiracijos kriterijus, t. y. kaip sukeitimo rezultatas gautas sprendinys, kurio tikslo funkcijos reikšmė yra geresnė už geriausią surastą tikslo funkcijos reikšmę. Be to, atsižvelgiama į draudimų statusą su maža tikimybe a ($a = 0{,}02$), netgi jeigu aspiracijos kriterijus netaikomas. Tai šiek tiek padidina galimų ėjimų skaičių ir padeda išvengti paieškos stagnacijos.

Be tabu sąrašo, taip pat yra saugomi surasti, bet nepasirinkti geri sprendiniai [85]. Taip siekiama diversifikuoti paieškos procesą ir ištyrinėti daugiau paieškos erdvės sričių. Tam sukuriamas ir naudojamas archyvas ($Archive$), kuriame yra kaupiami vadinamieji antrieji sprendiniai. Kiekvienoje iteracijoje geriausias sprendinys tampa esamuoju sprendiniu, o geriausias antrasis sprendinys $p^{\sim}$ yra išsaugomas archyve ($Archive$). Kai L_{idle_iter} skaičių iteracijų geriausias sprendinys nėra pagerinamas, TP pradedama vykdyti nuo vieno iš $Archive$ sprendinio. Parametro

[7] Išsamiau pateikiama straipsnyje [16].

L_{idle_iter} reikšmė yra lygi $\lfloor 0{,}2\tau \rfloor$; τ – TP iteracijų skaičius. TP baigus darbą, $Archive$ išvalomas. TP pseudokodas pateikiamas 7 pav.

```
procedure Tabu_paieška // tabu paieškos algoritmas
//įvestis: n, m, B, p – pradinis sprendinys
//rezultatas: p• – rastas geriausias sprendinys
//parametrai: τ – TP iteracijų skaičius, h – draudimų valdymas, α – randomizavimo koeficientas,
//            ρ – aplinkos dydžio reguliavimo parametras, L_idle_iter – tuščiosios eigos iteracijų limitas
begin
  clear tabu list TabuList;
  p• := p; k := 1; k' := 1; archive_counter := 0; improved := FALSE;
  while (k ≤ τ) or (improved = TRUE) then begin //pagrindinis ciklas
    M', N'; m' := |M'|; n' := m + |N'|; Δ'min := ∞; Δ''min := ∞; v' := 1; w' := m + 1
    for i := 1 to m' do
      for j := m + 1 to n' do begin //m'(n' – m) tikrinami sprendinio p kaimynai
        Δ := 2(c(p(j)) – c(p(i)) – B(p(i), p(j)));
        forbidden := iif(((TabuList(p(i), p(j)) ≥ k) and (random() ≥ α)), TRUE, FALSE);
        aspired := iif(z(p) + Δ < z(p•), TRUE, FALSE);
        if ((Δ < Δ'min) and (forbidden = FALSE)) or (aspired = TRUE) then begin
          if Δ < Δ'min then begin Δ''min := Δ'min; v'' := v'; w'' := w';
            Δ'min := Δ; v' := i; w' := j endif
          else if Δ < Δ''min then begin Δ''min := Δ; v'' := i; w'' := j endif
        endif
      endfor;
    if Δ''min < ∞ then begin // išsaugomas antrasis geriausias sprendinys, c, v'', w''
      archive_counter := archive_counter + 1; Archive(archive_counter) ← p, c, v'', w''
    endif;
    if Δ'min < ∞ then begin // esamo sprendinio pakeitimas ir c perskaičiavimas
      p := p^{v',w'}; for i := 1 to n do c(i) := c(i) + B(i, p(v')) – B(i, p(w'));
      if z(p) < z(p•) then begin p• := p; k' := k endif;
      TabuList(p(v'), p(w')) := k + h; TabuList(p(w'), p(v')) := k + h //p(v'), p(w') tampa tabu
    endif;
    improved := iif(Δ'min < 0, TRUE, FALSE);
    if (improved = FALSE) and (k – k' > L_idle_iter) and (k < τ – L_idle_iter) then begin
      random_access_index := random(archive_counter * 0.8, archive_counter);
      p, c, v'', w'' ← Archive(random_access_index);
      p := p^{v'',w''}; for i := 1 to n do c(i) := c(i) + B(i, p(v'')) – B(i, p(w''));
      clear tabu list TabuList;
      TabuList(p(v''), p(w'')) := k + h; TabuList(p(w''), p(v'')) := k + h; //p(v''), p(w'') – tabu
      k' := k
    endif; k := k + 1
  endwhile
end.
```

7 pav. Tabu paieškos pseudokodas[8]

[8] Pastabos. Tiesioginė **if** funkcija (angl. *immediate if function)* **iif**(x, y_1, y_2) grąžina y_1, jeigu x = TRUE, kitu atveju grąžina y_2 reikšmę. Funkcija **random**() grąžina pseudoatsitiktinį skaičių iš intervalo [0, 1]. Funkcija **random**(x_1, x_2) grąžina skaičių ribose $[x_1, x_2]$.

B. *Iteratyvioji tabu paieška*

Iteratyviosios tabu paieškos algoritmas apima prieš tai aprašytą savarankišką TP algoritmą, kombinuotą kartu su tam tikrais pertvarkymais (angl. *perturbations*). TP transformuoja esamą sprendinį į optimizuotą (pagerintą) sprendinį. Tada pertvarkymo procedūra yra taikoma pasirinktam optimizuotam sprendiniui. Kandidatas pasirenkamas pagal kandidatų atrankos taisyklę (žr. toliau). Pertvarkytas sprendinys grąžinamas kaip naujas esamasis sprendinys TP algoritmui, o šis vykdomas iš karto po pertvarkymo procedūros. TP grąžina optimizuotą sprendinį, jam taikoma pertvarkymo procedūra ir t. t. ITP algoritmo rezultatas yra geriausias rastas sprendinys. Procesas tęsiamas, kol pasiekiamas nustatytas iteracijų skaičius (8 pav.).

```
procedure Iteratyvioji_tabu_paieška;
// duomenys: p – esamas sprendinys
// rezultatas: p∇ – geriausias rastas sprendinys
// parametrai: Q – iteracijų skaičius
begin
  p∇ := p;
  for q := 1 to Q do begin
    taikyti Tabu_paieška sprendiniui p ir gauti p•;
    if z(p•) < z(p∇) p∇ := p•; // įsimenamas geriausias sprendinys
    if q < Q then begin
       p := Kandidatų_atrinkimas(p•, p∇);
       taikyti Pertvarkymas sprendiniui p
    endif
  endfor
end.
```

8 pav. Iteratyviosios tabu paieškos pseudokodas

C. *Hierarchinė iteratyvioji tabu paieška*

1-ojo lygio hierarchinis iteratyviosios tabu paieškos (1-HITP) algoritmas gali būti gaunamas iš ITP algoritmo. Algoritmo struktūra išlieka beveik nepakitusi, išskyrus tai, kad vietoj TP algoritmo taikomas ITP algoritmas (9 pav.).

```
procedure 1-Hierarchinė_iteratyvioji_tabu_paieška ; //1-ojo lygio HITP
// duomenys: p – pradinis sprendinys
// rezultatas: p(1) – geriausias rastas sprendinys; parametrai: Q1 – iteracijų skaičius
begin
  p(1) := p;
  for q1 := 1 to Q1 do begin
    taikyti Iteratyvioji_tabu_paieška sprendiniui p ir gauti p∇;
    if z(p∇) < z(p(1)) p(1) := p∇; //įsimenamas geriausias sprendinys
    if q1 < Q1 then begin
       p := Kandidatų_atrinkimas(p∇, p(1));taikyti Pertvarkymas sprendiniui p
    endif
  endfor
end.
```

9 pav. Pirmojo lygio hierarchinės iteratyviosios tabu paieškos pseudokodas

```
procedure Hierarchinė_iteratyvioji_tabu_paieška ; //7 (aukščiausio) lygio HITP
// duomenys: n, m, B,
//        p – pradinis sprendinys
// rezultatas: p☆ – geriausias rastas sprendinys
// parametrai: Q, Q1, . . . , Q7 – iteracijų skaičius
begin
  for i := 1 to n do begin c(i) := 0;
  for i := 1 to n do for j := 1 to m do c(i) := c(i) + B(i, p(j));
  p⟨7⟩ := p;
  for q7 := 1 to Q7 do begin
    . . .
    p⟨1⟩ := p;
    for q1 := 1 to Q1 do begin
      p▽ := p;
      for q := 1 to Q do begin
        taikyti Tabu_paieška sprendiniui p ir gauti p•;
        if z(p•) < z(p▽) then p▽ := p•; // įsimenamas geriausias sprendinys
        if q < Q then begin
          p := Kandidatų_atrinkimas (p•, p▽); taikyti Pertvarkymas spr. p
        endif
      endfor;                                          Iteratyvioji_tabu_paieška
      if z(p▽) < z(p⟨1⟩) then p⟨1⟩ := p▽; //įsimenamas geriausias sprendinys
      if q1 < Q1 then begin
        p := Kandidatų_atrinkimas (p▽, p⟨1⟩); taikyti Pertvarkymas spr. p
      endif
    endfor;                                    1-Hierarchinė_iteratyvioji_tabu_paieška
    . . .
    if z(p⟨6⟩) < z(p⟨7⟩) then p⟨7⟩ := p⟨6⟩; // įsimenamas geriausias sprendinys
    if q7 < Q7 then begin
      p := Kandidatų_atrinkimas (p⟨6⟩, p⟨7⟩); taikyti Pertvarkymas spr. p
    endif
  endfor;
  p☆ := p⟨7⟩
end.                                          7-Hierarchinė_iteratyvioji_tabu_paieška
```

10 pav. Septintojo lygio hierarchinės iteratyviosios tabu paieškos pseudokodas

Kiti lygiai generuojami taip: 2-HITP algoritmas yra toks pats kaip ir 1-HITP, tik vietoj ITP procedūros įkomponuojama 1-HITP procedūra. Kitaip tariant, algoritmai kombinuojami su panašiais į save algoritmais. Taip konstruojamas norimas skaičius lygių. Pristatomame algoritme yra 7 lygiai (10 pav.).

Kandidatų atrankos funkcija (angl. *candidate acceptance*) gali būti realizuota skirtingais būdais. Šiuo atveju taikyta taisyklė, kad visada pasirenkamas paskutinis gautas optimizuotas sprendinys.

Pertvarkymo procedūra (angl. *perturbation*) susideda iš: a) atsitiktinės mutacijos (sumaišymo) ir b) po mutacijos gauto sprendinio pertvarkymo taikant greitąją godžiąją adaptyviąją procedūrą (11 pav.).

```
procedure Pertvarkymas;
// duomenys: p – pradinis sprendinys
// rezultatas: p – pertvarkytas sprendinys
// parametrai: μ – mutacijos lygis
begin
   taikyti Mutacija sprendiniui p naudojant mutacijos lygį μ, gauti p⁻;
   taikyti Godžioji_adaptyvioji_procedūra sprendiniui p⁻,
   gauti pertvarkytą sprendinį p
   // sprendinys p gerinamas taikant TS
end.
```

11 pav. Sprendinio pertvarkymo procedūros pseudokodas

Pirma, atrinktam kandidatui taikoma atsitiktinės mutacijos procedūra, kai μ skaičius elementų sprendinyje yra ignoruojamas (parametras μ vadinamas mutacijos lygiu (angl. *mutation rate*). Ignoruojami μ elementai pasirenkami atsitiktinai (12 pav.). Algoritme parametro μ reikšmė yra gana nedidelė ($\mu = \lfloor 0{,}15m \rfloor$), taigi tik nedidelė elementų dalis mutuoja.

```
procedure Mutacija;  //atsitiktinės mutacijos procedūra
//duomenys: m,
//          p – pradinis sprendinys
//rezultatas: p⁻ – gautas dalinis sprendinys
begin
   for i := 1 to m − 1 do begin  //sumaišomi (sukeičiami vietomis) elementai p(1), …, p(m)
      atsitiktinai generuoti sveiką skaičių j, i ≤ j ≤ m;
      p := p^{i,j}  //elementai p(i) ir p(j) sukeičiami
   endfor;
   //po sumaišymo elementai p(m − μ + 1), …, p(m) ignoruojami
   p⁻ := p
end.
```

12 pav. Mutacijos procedūros pseudokodas

Antra, sprendinys, gautas pritaikius mutaciją, rekonstruojamas taikant greitąją godžiąją adaptyviąją procedūrą (angl. *fast greedy adaptive procedure*) (GGAP), kuri yra identiška taikytai per kryžminimo operaciją, išskyrus tai, kad yra pritaikytas efektyvesnis c reikšmių skaičiavimo metodas (13 pav.). Taikant šiuos metodus, GGAP ir HITP algoritmų vykdymas yra labai greitas tol, kol μ reikšmė nėra didelė.

```
procedure Greitoji_godžioji_adaptyvioji_procedūra;  //GGAP
// duomenys: n, m, B,
//        p – dalinis sprendinys, kur elementai p(m – μ + 1), …, p(m) yra ignoruojami
// rezultatas: p – suformuotas sprendinys
// parametrai: μ – mutacijos lygis
begin
  for i := 1 to n do for j := m – μ + 1 to m do c(i) := c(i) – B(i, p(j));
//greitas sumų (c) perskaičiavimas
  for i := 1 to n do Selected(i) := FALSE; for i := 1 to m – μ do Selected(p(i)) := TRUE;
//Masyvo Selected inicializavimas
  i := m – μ; k := p(m – μ);
  for q := 1 to μ do begin  //ciklas kartojamas tol, kol sprendinys bus suformuotas
    minimum_contribution := ∞;
    for j := 1 to n do
      if Selected(j) = FALSE then begin
        c(j) := c(j) + B(j, k);
        if c(j) < minimum_contribution then begin
          minimum_contribution := c(j); j_min := j
        endif;
      endif;
    i := i + 1; p(i) := j_min;  //elementas su minimalia minimum_contribution reikšme įtraukiamas
    Selected(p(i)) := TRUE; k := j_min
  endfor;
  i := m + 1;
  for j := 1 to n do if Selected(j) = FALSE then begin p(i) := j; i := i + 1
endif
end.
```

13 pav. GGAP procedūros pseudokodas

2.4.5. Populiacijos atnaujinimas

Įvykdžius HITP, gaunamas pagerintas palikuonis ir tikrinama, ar naujasis sprendinys ($p^{\star}$) skiriasi nuo kitų populiacijos sprendinių. Jeigu skiriasi, tikrinama, ar naujasis sprendinys yra geresnis už geriausią sprendinį populiacijoje arba ar atstumas tarp naujojo sprendinio ir populiacijos ($\delta(p^{\star}, P) = \min_{p \in P}\{\delta(p^{\star}, p)\}$) yra didesnis už nustatytą atstumų slenkstį *DT* (angl. *distance threshold*) arba yra jam lygus. Jeigu taip, tada naujasis sprendinys įtraukiamas į populiaciją vietoj blogiausio sprendinio ($P = P \cup \{p^{\star}\} \setminus \{p_{worst}\}$, kur $p_{worst} = \operatorname{argmin}_{p \in P}\{z(p)\}$). Priešingu atveju populiacija išlieka nepakeista ir vykdoma kita generacija. Ši taisyklė leidžia išlaikyti sprendinių kokybę ir pakankamą populiacijos narių įvairovę.

2.4.6. Populiacijos pertvarkymas

Jeigu sprendiniai nėra pagerinami L_{idle_gen} skaičių generacijų (L_{idle_gen} – tuščiosios eigos generacijų skaičius, nustatytas į $\lfloor 0{,}15N_{gen} \rfloor$, N_{gen} – generacijų skaičius), tada atliekama populiacijos pertvarkymo (angl. *restart*) procedūra, kurios metu populiacija generuojama iš naujo (žr. 2.4.1 skyrelį).

2.5. Hibridinio genetinio algoritmo komponentai

HGA tyrimas buvo paremtas HGA atskirų savybių, t. y. komponentų, analize (algoritmo komponentine analize). Eksperimentuose tirta išskirtų esminių komponentų svarba ir įtaka bendrajam HGA efektyvumui (našumui). Eksperimentai atlikti su 7 komponentais ir jų modifikacijomis [86].

Kiekvienai visų tirtų komponentų modifikacijai žymėti toliau darbe vartojamos santrumpos iš dviejų ar trijų raidžių ir skaičiaus. Raidės nurodo komponento pavadinimą, o skaičius – to komponento modifikacijos numerį (pvz., **`PP-1`** žymi komponento „PRADINĖ POPULIACIJA“ pirmąją modifikaciją).

2.5.1. Bazinis hibridinio genetinio algoritmo variantas

Bazinis HGA (**`HGA-BV`**) variantas buvo sudarytas atsižvelgiant į preliminarius eksperimentų rezultatus rengiant publikaciją (83). HGA sudarytas iš komponentų: **`PP-3`**, **`LP-5`**, **`STA-3`**, **`KP-10`**, **`KS-2`**, **`PA-5`**, **`PR-7`** (komponentai išsamiai aprašomi 2.5.2–2.5.8 skyreliuose). Bazinio HGA parametrų reikšmės pateiktos 1 lentelėje.

1 lentelė. `HGA-BV` naudotų parametrų reikšmės

Parametras	Reikšmė	Aprašymas
Populiacijos dydis, PS	20	
Generacijų skaičius, N_{gen}	100	
Tuščiosios eigos generacijų limitas, L_{idle_gen}	$\lfloor 0{,}03N_{gen} \rfloor$	$0 < L_{idle\_gen} \leq N_{gen}$
Atstumų slenkstis, DT	$\lfloor 0{,}15m \rfloor$	$0 \leq DT \leq m$
HITP iteracijų skaičius, Q_{HIER}	256	$Q_{HIER} = Q \times Q_1 \times Q_2 \times Q_3 \times Q_4 \times Q_5 \times Q_6 \times Q_7$ [†]
Pradinis HITP iteracijų skaičius[††], Q_{IHIER}	4096	$Q_{IHIER} = Q \times Q_1 \times Q_2 \times Q_3 \times Q_8 \times Q_9 \times Q_{10} \times Q_{11}$ [†††]
Pakartotinio paleidimo HITP iteracijų skaičius[†††], Q_{RHIER}		$Q_{RHIER} = Q \times Q_1 \times Q_2 \times Q_3 \times Q_{12} \times Q_{13} \times Q_{14} \times Q_{15}$ [†††††]
TP iteracijų skaičius, τ	50	
Uždraudimai, h	$\lfloor 0{,}3m \rfloor$	$h > 0$
Tuščiosios eigos iteracijų limitas, L_{idle_iter}	$\lfloor 0{,}2\tau \rfloor$	$0 < L_{idle\_iter} \leq \tau$
Aplinkos dydžio parametras, ρ	0,4	$\rho > 0$
Atsitiktinumų koeficientas, α	0,02	$0 < \alpha < 1$
Mutacijos lygis, μ	$\lfloor 0{,}15m \rfloor$	$0 < \mu < m$

[†] $Q = Q_1 = Q_2 = Q_3 = Q_4 = Q_5 = Q_6 = Q_7 = 2$;

[††] tabu paieškos iteracijų skaičius, naudojamas gerinant pradinę populiaciją;

[†††] $Q_8 = Q_9 = Q_{10} = Q_{11} = 4$ ($Q_{IHIER} = 16Q_{HIER}$);

[††††] tabu paieškos iteracijų skaičius, naudojamas populiacijos pakartotinio paleidimo procedūroje;

[†††††] $Q_{12} = Q_{13} = Q_{14} = Q_{15} = 3$.

2.5.2. Komponentas „PRADINĖ POPULIACIJA“

Tirtos trys komponento „PRADINĖ POPULIACIJA“ (angl. „INITIAL POPULATION“) modifikacijos.

A. *Atsitiktinė populiacija* (**PP-1**)

Populiacija generuojama atsitiktinai ir negerinama, t. y., sukonstravus pradinę populiaciją (*PS* dydžio), HITP netaikoma ($N_{gen} = 200$). Kitos parametrų reikšmės yra tokios pačios, kaip bazinėje algoritmo versijoje.

B. *Padidinta ir pagerinta populiacija* (**PP-2**)

HGA pradeda dirbti su pradine populiacija, kurios dydis yra *INIT_PS* = 2*PS*. *PS* – populiacijos dydis, kuris bus naudojamas toliau genetiniame algoritme. Sukonstruota padidinta populiacija pagerinama taikant HITP. Į galutinę populiaciją įtraukiami geresnieji individai iš padidintos populiacijos. Taip sukonstruojama geros kokybės pradinė populiacija. Kitos parametrų reikšmės yra tokios pačios, kaip bazinėje algoritmo versijoje (1 lent.).

C. *Pagerinta populiacija* (**PP-3**)

HGA pradeda dirbti su pradine populiacija, kurios dydis yra *PS*. Pradinė populiacija pagerinama taikant HITP. Kitos parametrų reikšmės yra tokios pačios, kaip bazinėje algoritmo versijoje.

2.5.3. Komponentas „LOKALUSIS PAGERINIMAS“

Bendrąja prasme genetinė lokalioji paieška yra paremta intensifikavimu ir diversifikavimu. Tinkamai parinkta intensifikavimo strategija gali turėti didelę įtaką hibridinio genetinio algoritmo rezultatų efektyvumui [87]. Siekiant nustatyti tinkamiausią strategiją (kombinaciją), buvo atlikti eksperimentai su keturiomis komponento „LOKALUSIS PAGERINIMAS“ (angl. „LOCAL IMPROVEMENT“) modifikacijomis. Šiuo atveju tabu paieškos iteracijų skaičius, hierarchinės iteratyviosios tabu paieškos iteracijų skaičius, generacijų skaičius eksperimentų metu buvo didinami (mažinami).

A. *Tabu paieškos iteracijų skaičiaus mažinimas* (**LP-1**)

Panaudotas sumažintas tabu paieškos iteracijų skaičius $\tau = 25$ ir padidintas generacijų skaičius $N_{gen} = 200$. Kitos parametrų reikšmės yra tokios pačios, kaip bazinėje algoritmo versijoje.

B. *Tabu paieškos iteracijų skaičiaus didinimas* (**LP-2**)

Panaudotas padidintas tabu paieškos iteracijų skaičius $\tau = 100$ ir sumažintas generacijų skaičius $N_{gen} = 50$. Kitos parametrų reikšmės yra nepakitusios.

C. *Hierarchinės iteratyviosios tabu paieškos iteracijų skaičiaus mažinimas* (**LP-3**)

Panaudotas sumažintas hierarchinės iteratyviosios tabu paieškos iteracijų skaičius $Q_{HIER} = 1{\times}2{\times}2{\times}2{\times}2{\times}2{\times}2 = 128$ ir padidintas generacijų skaičius $N_{gen} = 200$. Kitos parametrų reikšmės yra nepakitusios.

D. *Hierarchinės iteratyviosios tabu paieškos iteracijų skaičiaus didinimas* (**LP-4**)

Panaudotas padidintas hierarchinės iteratyviosios tabu paieškos iteracijų skaičius $Q_{HIER} = 2{\times}2{\times}2{\times}2{\times}2{\times}2{\times}4 = 512$ ir sumažintas generacijų skaičius $N_{gen} = 50$. Kitos parametrų reikšmės yra nepakitusios.

E. *Nepakeistas tabu paieškos iteracijų skaičius* (**LP-5**)

Visų parametrų reikšmės yra tokios pačios, kaip bazinėje algoritmo versijoje.

2.5.4. Komponentas „SPRENDINIŲ-TĖVŲ ATRANKA“

Atliekant eksperimentus su komponento (savybės) „SPRENDINIŲ-TĖVŲ ATRANKA“ (angl. „SELECTION“) modifikacijomis, buvo taikomos trys skirtingos atrankos procedūros.

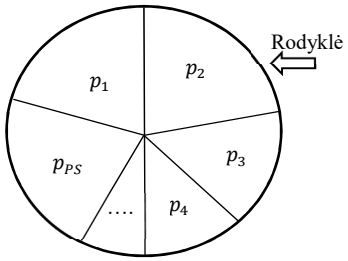
A. *Ruletės ratą modeliuojanti atrankos procedūra* (**`STA-1`**)

Ruletės ratą modeliuojančią atrankos procedūrą (proporcinga tikslo funkcijai (tinkamumui) atranka) (angl. *roulette wheel selection* arba *fitness proportionate selection*) glaustai galima aprašyti taip: visos populiacijos individai suskirstomi į tam tikrus sektorius (sritis) pagal tikslo funkcijos santykį su visų individų tikslo funkcijų suma (žr. (24) formulę); kiekvienas sektorius yra vienas individas; ruletės ratas „sukamas“ *PS* kartų (*PS* – populiacijos dydis), pasirenkamas tas individas, kurio sektoriuje rodyklė sustojo (14 pav.) [88–90].

Jeigu populiacijos dydis yra *PS*, $P = \{p_1, p_2, \ldots, p_{PS}\}$, o individo p_i tikslo funkcijos reikšmė yra $f(p_i)$, tai tikimybė individui p_i būti pasirinktam apskaičiuojama pagal formulę:

$$t(p_i) = \frac{f(p_i)}{\sum_{j=1}^{PS} f(p_j)}; \tag{24}$$

čia $i = 1, 2, \ldots, PS$.



14 pav. Ruletės ratą modeliuojanti atranka

B. *Sprendinių rangu pagrįsta atrankos procedūra* (**`STA-2`**)

Taikant sprendinių (individų) rangu pagrįstą atranką (netiesiogiai proporcinga tikslo funkcijai (tinkamumui) atranka) (angl. *rank-based selection*), populiacijos nariai pirmiausia surikiuojami atsižvelgiant į jų tikslo funkcijos reikšmes ir jiems priskiriamus rangus. Pagal rangą priskiriama atrinkimo tikimybė. Individai atrenkami pagal jiems priskirtą atrinkimo tikimybę [91].

Rangu pagrįstos atrankos taisyklė yra tokia [61]: eilinio parenkamo tėvo pozicija *u* sutvarkytoje, t. y. surikiuotoje, populiacijoje nustatoma pagal šią formulę: $u = \lfloor v^{\sigma} \rfloor$; čia v – atsitiktinis sveikasis skaičius iš intervalo $\left[1, PS^{\frac{1}{\sigma}}\right]$, *PS* – populiacijos dydis, o σ – realusis skaičius iš intervalo [1, 2]. Pirmojo ir antrojo tėvo pozicijos u reikšmė turi skirtis. σ yra laikomas atrankos parametru. HGA algoritme $\sigma = 2$. Akivaizdu, jog kuo geresnis individas, tuo didesnė tikimybė, kad jis bus atrinktas tolesniam kryžminimui.

C. *Atsitiktinė atranka* (**STA-3**)

Šiuo atveju sprendiniai-tėvai atrenkami tiesiog generuojant atsitiktinį skaičių intervale $[1, PS]$; čia PS yra populiacijos dydis.

2.5.5. Komponentas „KRYŽMINIMO PROCEDŪRA“

Atliekant eksperimentus su komponento (savybės) „KRYŽMINIMO PROCEDŪRA“ (angl. „CROSSOVER“) modifikacijomis, taikyta dešimt skirtingų kryžminimo procedūrų.

A. *Dažnumu paremtas godusis kryžminimas* (**KP-1**)

Taikytas (tėvų genų) dažnumu paremtas godusis kryžminimas (angl. *frequency(backbone)-based greedy crossover*). Ši kryžminimo procedūra yra tokia pati, kaip aprašytame HGA (žr. 2.4.3 skyrelį), tik vietoj dviejų palikuonių generuojamas vienas (nesukuriamas priešingasis (opozicinis) palikuonis).

B. *Godusis priešingasis (opozicinis) kryžminimas* (**KP-2**)

Taikytas godusis priešingasis (opozicinis) kryžminimas su priešingųjų (opozicinių) palikuonių formavimu (angl. *greedy opposition-based crossover*). Sugeneruojami du palikuoniai p°, $p^{\circ\circ}$.

C. *Godusis kryžminimas* (**KP-3**)

Taikytas godusis kryžminimas (godusis kryžminimas be priešingųjų (opozicinių) palikuonių formavimo) (angl. *greedy crossover*). Gaunamas vienas palikuonis p°.

D. *Euristinis priešingasis (opozicinis) kryžminimas* (**KP-4**)

Taikytas euristinis priešingasis (opozicinis) kryžminimas su priešingųjų (opozicinių) palikuonių formavimu (angl. *heuristic opposition-based crossover*). Euristinio kryžminimo procedūroje atsižvelgiama į tai, kiek yra bendrų tėvų genų. Jų pagrindu ir formuojamas palikuonis. Jeigu yra daug bendrų genų, įtraukiami ir svetimi genai iš kitų individų. Euristinio kryžminimo procedūra atlieka ir dalinį palikuonių pagerinimą. Ši kryžminimo procedūra išsamiau aprašyta Misevičiaus ir Dreznerio [20, 24]. Taikant euristinį opozicinį kryžminimą sugeneruojami du palikuoniai p°, $p^{\circ\circ}$.

E. *Euristinis kryžminimas* (**KP-5**)

Taikytas euristinis kryžminimas (euristinis kryžminimas be priešingųjų (opozicinių) palikuonių formavimo) (angl. *heuristic crossover*). Kryžminimo procedūra yra tokia pati, kaip 4 varianto, tik sugeneruojamas palikuonis p° be priešingojo palikuonio.

F. *Daugelio tėvų genų dažnumu paremtas priešingasis (opozicinis) kryžminimas* (**KP-6**)

Taikytas daugelio tėvų genų dažnumu paremtas kryžminimas su priešingųjų (opozicinių) palikuonių formavimu (angl. *multi-parent frequency and opposition-based crossover*). Daugelio tėvų kryžminimo procedūroje gali būti panaudojami visi populiacijos nariai arba jų dalis [62, 92]. Eksperimentams buvo panaudoti visi populiacijos nariai. Palikuonio p° i-tajam elementui priskiriama reikšmė j su tikimybe t $(p^{\circ}(i) = j)$ (jei j nebuvo panaudotas anksčiau). Tikimybė t apskaičiuojama pagal formulę:

$$t(p^{\circ}(i) = j) = q_{ij} \,/\, \sum_{j}^{n} q_{ij}; \tag{25}$$

čia $t(p^{\circ}(i) = j)$ apibrėžia tikimybę, kad $p^{\circ}(i) = j$; q_{ij} yra matricos $Q = (q_{ij})_{n \times n}$ elementas; q_{ij} nurodo, kiek kartų i-tojoje vietoje buvo j-oji reikšmė; $q_{ij} \leq v$ (v – tėvų skaičius). Jeigu yra kelios reikšmės $(j_1, j_2, \dots)$ su vienoda tikimybe, tada reikšmė pasirenkama atsitiktinai [24]. Šiuo atveju sugeneruojami du palikuoniai p°, $p^{\circ\circ}$.

G. *Daugelio tėvų genų dažnumu paremtas kryžminimas* (**KP-7**)

Taikytas daugelio tėvų genų dažnumu paremtas kryžminimas be priešingųjų (opozicinių) palikuonių formavimo (angl. *multi-parent frequency-based crossover*). Šiuo atveju sugeneruojamas vienas palikuonis p°.

H. *Priderintasis (specifinis) priešingasis (opozicinis) kryžminimas* (**KP-8**)

Taikytas priderintasis (specifinis) kryžminimas su priešingųjų (opozicinių) palikuonių formavimu (angl. *tailored (problem-specific) opposition-based crossover*). Pasitelktas kryžminimas, kurį taikant analitinio sprendinio elementai traktuojami kaip grafiniai juodi taškai, išdėstyti tinklelio eilutėse. Tada palikuonio elementai generuojami kaip dviejų gretimų taškų geometrinis vidurkis. Sugeneruojami du palikuoniai p°, $p^{\circ\circ}$.

I. *Priderintasis (specifinis) kryžminimas* (**KP-9**)

Taikytas priderintasis (specifinis) kryžminimas be priešingųjų (opozicinių) palikuonių formavimo (angl. *tailored (problem-specific) crossover*). Šiuo būdu sugeneruojamas vienas palikuonis p°.

J. *Dažnumu paremtas godusis priešingasis (opozicinis) kryžminimas* (**KP-10**)

Ši kryžminimo procedūra yra tokia pati, kaip ir aprašytame baziniame HGA variante (žr. 2.4.3 skyrelį).

Atliekant eksperimentus su komponentu „KRYŽMINIMO PROCEDŪRA“, kitų parametrų reikšmės išlieka tokios pačios, kaip bazinėje algoritmo versijoje.

2.5.6. Komponentas „KRYŽMINIMŲ SKAIČIUS“

A. *Padidintas kryžminimų skaičius* (**KS-1**)

Tiriant komponentą (savybę) „KRYŽMINIMŲ SKAIČIUS“ (angl. „NUMBER OF CROSSOVERS PER GENERATION“), vienos generacijos metu buvo panaudotas padidintas kryžminimų skaičius $\frac{PS}{2} = 10$; čia PS – populiacijos dydis (lygus 20). Generacijų skaičius N_{gen} sumažintas iki 20, taip pat sumažintas hierarchinės iteratyviosios tabu paieškos iteracijų skaičius: $Q_{HIER} = Q \times Q_1 \times Q_2 \times Q_3 \times Q_4 \times Q_5 \times Q_6 \times Q_7 = 1 \times 2 \times 2 \times 2 \times 2 \times 2 \times 2 \times 2 = 128$. Kitos parametrų reikšmės yra tokios pačios, kaip bazinėje algoritmo versijoje.

B. *Vienas kryžminimas* (**KS-2**)

Atliekamas vienas kryžminimas. Kitos parametrų reikšmės yra nepakitusios.

2.5.7. Komponentas „POPULIACIJOS ATNAUJINIMAS“

Tiriant komponentą (savybę) „POPULIACIJOS ATNAUJINIMAS“ (angl. „POPULATION REPLACEMENT“), buvo taikomos keturios populiacijos atnaujinimo strategijos.

A. *Blogiausio populiacijos nario pakeitimas geresniu palikuoniu* (**PA-1**)

Šiuo atveju naudojama atnaujinimo schema, kurią formaliai galima apibrėžti kaip „$\mu+\lambda$, ε“ schemą. Šioje schemoje naujai generacijai atrenkama μ geriausių

individų iš $P_\mu \cup P_\lambda$; čia P_μ yra esama (esamos generacijos) populiacija (dar be naujų narių), o P_λ žymi aibę (rinkinį), sudarytą iš naujų (pagerintų) palikuonių. HGA algoritme $\mu = PS$. Taip pat turi būti patenkinta sąlyga: $\min_{p \in P}\{\delta(p_\lambda, p)\} \geq \varepsilon$ visiems $p_\lambda \in P_\lambda$. HGA algoritme $\varepsilon = DT$; čia DT žymi atstumo tarp individų minimalųjį slenkstį.

Šioje modifikacijoje $\lambda = 1$. Taigi, naujasis palikuonis tiesiog pakeičia esamos populiacijos blogiausią individą (jeigu palikuonis yra geresnis už blogiausią individą; priešingu atveju palikuonis atmetamas (neįtraukiamas į populiaciją). Turi būti patenkinta sąlyga: $\min_{p \in P}\{\delta(p_\lambda, p)\} \geq \varepsilon$; čia p_λ yra naujasis palikuonis. Jei sąlyga nepatenkinta, palikuonis ignoruojamas.

B. *Blogiausio populiacijos nario pakeitimas* (**PA-2**)

Panaudota atnaujinimo schema, kurią formaliai galima apibrėžti kaip „μ, λ, ε“ schemą. Šioje schemoje λ naujų individų, t. y. palikuonių, pakeičia tokį patį skaičių esamos populiacijos individų. Į palikuonių kokybę (tinkamumą) neatsižvelgiama. Turi būti patenkinta sąlyga: $\min_{p \in P}\{\delta(p_\lambda, p)\} \geq \varepsilon$ visiems $p_\lambda \in P_\lambda$; čia P_λ yra aibė (rinkinys), sudaryta iš naujų palikuonių.

Šioje modifikacijoje $\lambda = 1$. Šiuo atveju naujuoju palikuoniu tiesiog pakeičiamas esamos populiacijos blogiausias individas, nekreipiant dėmesio į palikuonio kokybę (tikslo funkcijos reikšmę). Tačiau turi būti patenkinta sąlyga: $\min_{p \in P}\{\delta(p_\lambda, p)\} \geq \varepsilon$; čia p_λ yra naujasis palikuonis.

C. *Blogesniojo tėvo pakeitimas geresniu palikuoniu* (**PA-3**)

Ši modifikacija yra labai panaši į modifikaciją **PA-1**. Tik šiuo atveju palikuonis pakeičia blogesnįjį iš savo tėvų (jeigu palikuonis yra geresnis už blogesnįjį tėvą).

D. *Blogesniojo tėvo pakeitimas* (**PA-4**)

Ši modifikacija yra labai panaši į **PA-3**, tik šiuo atveju palikuonis pakeičia blogesnįjį iš savo tėvų besąlygiškai (neatsižvelgiama į palikuonio kokybę).

E. *Blogiausio populiacijos nario pakeitimas su papildoma sąlyga* (**PA-5**)

Bazinėje (standartinėje) HGA versijoje naudojama atnaujinimo schema, kuri iš esmės yra „μ+1, ε“ schema. Tik šioje schemoje (papildomai) tikrinama, ar gautas naujas palikuonis yra geresnis už patį geriausią esamos populiacijos narį. Jei taip yra, palikuonis pakeičia būtent tą narį. Visa kita yra taip, kaip ir modifikacijoje **PA-1**.

Visų parametrų reikšmės yra tokios pačios, kaip ir bazinėje versijoje.

2.5.8. Komponentas „POPULIACIJOS PERTVARKYMAS“

Atlikti eksperimentai su septyniomis komponento „POPULIACIJOS PERTVARKYMAS“ (angl. „POPULATION RESTART (RESET)“) modifikacijomis.

A. *Populiacijos pertvarkymo netaikymas* (**PR-1**)

Pirmojoje modifikacijoje nebuvo atliekamas populiacijos pertvarkymas, t. y. buvo leidžiamas neribotas skaičius tuščiosios eigos generacijų. Visų parametrų reikšmės (išskyrus tuščiosios eigos generacijų limitą, L_{idle_gen}) yra tokios pačios, kaip ir bazinėje versijoje.

B. *Multimutacija paremtas populiacijos pertvarkymas* (**PR-2**)

Šiuo atveju populiacijos pertvarkymui taikyta esamos populiacijos multimutacija, t. y. vyko visų populiacijos narių mutacija. Mutacijai taikoma sprendinio elementų porinių sukeitimų dviejų žingsnių procedūra. Ją atliekant pirmajame žingsnyje vykdomi tik atsitiktiniai elementų sukeitimai, o antrajame žingsnyje atliekami tik tie atsitiktinai parinktų elementų sukeitimai, kurie sumažina tikslo funkcijos reikšmę. Mutacijos lygis (stiprumas) yra lygus $0{,}1 \times m$. (t. y. pasikeičia 10 % elementų).

C. *Priešingų (opozicinių) sprendinių generavimu paremtas populiacijos pertvarkymas* (**PR-3**)

Populiacijos pertvarkymo metu generuojami priešingi sprendiniai (priešingos, antipodinės (esamos atžvilgiu) populiacijos generavimas). Generuojama tiek priešingų sprendinių, koks yra populiacijos dydis.

D. *Genų translokacija paremtas populiacijos pertvarkymas* (**PR-4**)

Populiacijos pertvarkymui taikoma speciali esamos populiacijos individų genų išmaišymo procedūra – vadinamoji genų translokacija (angl. *gene translocation*). Genų translokaciją galima traktuoti kaip vertikaliąją mutaciją, t. y. keičiami savi populiacijos genai populiacijos viduje, skirtingai nuo multimutacijos, kai vyksta horizontalioji genų migracija (mutacijos procese dalyvauja ir svetimi genai). Genų translokacijos procedūra vykdoma iteratyviuoju būdu. Iteracijų skaičius yra lygus k, $k = max\{1; 0{,}01 \cdot c \cdot PS(n-1)\}$; čia c – translokacijos koeficientas (procentais) ($0 < c \leq 100$), PS – populiacijos dydis, n – chromosomos ilgis. Kiekvienos iteracijos metu sugeneruojami trys atsitiktiniai skaičiai $\zeta_1, \zeta_2, \zeta_3$. Pirmieji du ($\zeta_1, \zeta_2$) yra iš intervalo $[1, PS]$, $\zeta_1 \neq \zeta_2$, o ζ_3 yra iš intervalo $[1, n]$. Tada sukeičiami ζ_1-ojo ir ζ_2-ojo narių genai, esantys pozicijoje ζ_3 (jeigu sprendinyje neatsiranda vienodų genų). Genų translokacijos iteracijos tęsiamos tol, kol pasiekiamas nustatytas iteracijų skaičius. Rezultatas – gaunama nauja populiacija su c procentų naujų genų [62].

E. *Determinuotuoju chaosu pagrįstas populiacijos pertvarkymas* (**PR-5**)

Populiacijos pertvarkymui taikoma determinuotuoju chaosu pagrįsta populiacijos individų generavimo procedūra (chaotinis generavimas). Generuojant populiacijos sprendinių elementus buvo pasinaudota chaotinėse dinaminėse sistemose (šiuo atveju logistikos žemėlapio (angl. *logistic map*) taikomu metodu pagal formulę [93]:

$$x_{j+1} = k \times x_j \times (1 - x_j); \tag{26}$$

čia $k = 4$, x_0 – vartotojo pasirinktas realusis skaičius iš intervalo $[0, 1]$, $j = 0{,}1{,}2, \ldots, n-1$.

F. *Determinuotuoju chaosu pagrįstas modifikuotas populiacijos pertvarkymas* (**PR-6**)

Populiacijos pertvarkymui taip pat taikoma chaotinio generavimo procedūra. Ypatumas yra tas, kad generuojami chaotiniai realieji skaičiai tiesiogiai susiejami su sprendinių (perstatymų) sveikosiomis elementų reikšmėmis.

G. *Atsitiktinės populiacijos generavimu paremtas populiacijos pertvarkymas* (**PR-7**)

Šiuo atveju populiacijos pertvarkymo procedūra yra tiesiog atsitiktinės populiacijos generavimas. Sugeneruoti sprendiniai yra pagerinami taikant HITP – kaip ir visose kitose modifikacijose.

2.6. Antrojo skyriaus išvados

1. Sukurtas patobulintas (naujas) hibridinis genetinis algoritmas, kuriame hibridinis genetinis algoritmas integruotas su naujojo tipo hierarchinės iteratyviosios tabu paieškos procedūra.
2. Atsižvelgiant į pagrindinius algoritmo parametrus, galinčius daryti įtaką rezultatų efektyvumui, buvo atliktas HGA tyrimas, pagrįstas esminių algoritmo komponentų apibrėžimu ir išskyrimu. Jo pagrindu, atsižvelgiant į pagrindinius algoritmo parametrus, sudarytos skirtingos septynių HGA komponentų modifikacijos.

3. EKSPERIMENTINIAI TYRIMAI

3.1. Eksperimentinių tyrimų metodika

Naujasis hibridinis genetinis algoritmas buvo testuojamas sprendžiant vidutinio ir didelio dydžio PŠF uždavinius, kai $n = 256$ ir $n = 1024$. Matricos B reikšmės buvo generuojamos atsižvelgiant į metodą, aprašytą Misevičiaus, Guogio ir Stanevičienės straipsnyje [10][9]. Tinklelio matmenys 16×16, $n_1 = n_2 = 16$ ir 32×32, $n_1 = n_2 = 32$. Spalvos intensyvumo parametras *m* kito nuo 2 iki 128 ir nuo 2 iki 512.

Eksperimentuose buvo naudojamos HGA valdymo parametrų reikšmės pateiktos 2 lentelėje.

Iš pradžių eksperimentai atlikti pasirinkus uždavinio dydį *n* = 256. HGA buvo lyginamas su pagerintuoju genetiniu evoliuciniu algoritmu (PGEA), aprašytu Misevičiaus [20] ir laikytinu efektyviausiu algoritmu šio dydžio PŠF uždaviniui spręsti. PGEA paremtas intensifikavimo ir diversifikavimo metodais. Pagrindinės šio algoritmo dalys yra speciali sprendinių rekombinavimo procedūra ir algoritmas, taikomas po rekombinavimo (angl. *post-recombination algorithm*). Šiuo atveju rekombinavimo procedūra pasižymi tiek diversifikavimo, tiek intensifikavimo savybėmis. O algoritmas, taikomas po rekombinavimo, sudarytas iš iteratyviosios tabu paieškos ir mutacijos procedūrų, kur tabu paieška yra pagrindinis intensifikavimo mechanizmas.

Kadangi HGA ir PGEA per eksperimentus pasiekė geriausius žinomus sprendinius (angl. *best known solutions*) (GŽS), buvo lyginamas ir skaičiavimų laikas.

Kompiuterinio eksperimentinio tyrimo metu PGEA ir HGA buvo paleidžiami atitinkamai 10 ir 100 vykdymų. Kiekvieno algoritmų paleidimo metu naudojamas duotasis *m*, kiekvieną kartą pradedama nuo naujai atsitiktinai sugeneruotos populiacijos. Algoritmų vykdymas stabdomas tada, kai randamas geriausias žinomas sprendinys (GŽS) (jeigu dar nepasiektas generacijų limitas N_{gen}).

3.2. Hibridinio genetinio algoritmo palyginimo su PGEA rezultatai

Algoritmų vykdymo laikas (centrinio procesoriaus (CP) laikas), reikalingas surasti GŽS su kiekviena *m* reikšme, pateikiamas 3 lentelėje. PGEA yra pateikiamas laiko vidurkis (per 10 vykdymų). HGA pateikiamas trumpiausias laikas per 100 paleidimų. Gauti rezultatai rodo, kad HGA vykdymo laikas yra geresnis už PGEA. Dar pastebėta, kad galimai optimalus (pseudooptimalus) sprendinys yra randamas ankstyvojoje algoritmo stadijoje, sukonstravus pradinę populiaciją ir pritaikius HITP.

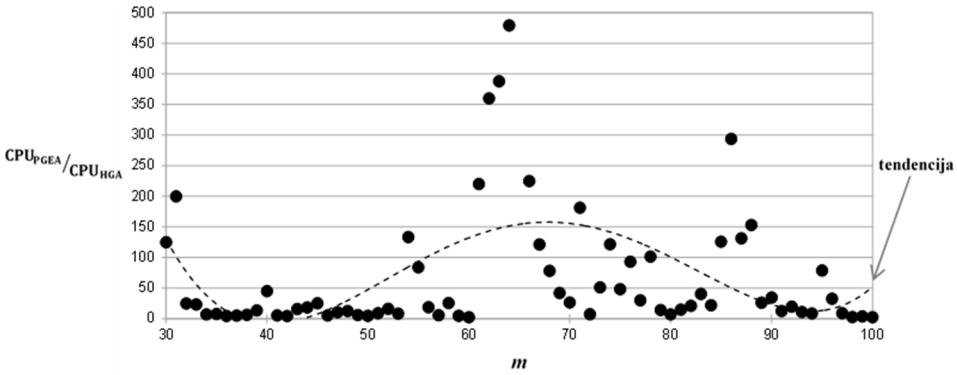
Atliekant eksperimentus su didelio dydžio PŠF uždaviniu, pavyko geriausias iki šiol žinomas tikslo funkcijos reikšmes pagerinti daugiau nei 190 atvejų. Rezultatai pateikiami 4 lentelėje. Šiuo atveju vykdymo laikas nebuvo lyginamas.

[9] Matricos *B* reikšmes galima rasti adresu http://www.personalas.ktu.lt/~alfmise/.

2 lentelė. Hibridinio genetinio algoritmo parametrų reikšmės, naudotos palyginimui

Parametrai	**Reikšmės**	**Komentarai**
Populiacijos dydis, PS	20	
Generacijų skaičius, N_{gen}	40	
Tuščiosios eigos generacijų limitas, L_{idle_gen}	$\lfloor 0{,}15N_{gen} \rfloor$	$0 < L_{idle\_gen} \leq N_{gen}$
Atstumų slenkstis, DT	$\lfloor 0{,}25m \rfloor$	$0 \leq DT \leq m$
HITP iteracijų skaičius, Q_{HIER}	384	$Q_{HIER} = Q{\times}Q_1{\times}Q_2{\times}Q_3{\times}Q_4{\times}Q_5{\times}Q_6{\times}Q_7$[†]
TP iteracijų skaičius, τ	80	
TP uždraudimų laikotarpis (matuojamas iteracijomis), h	$\lfloor 0{,}3m \rfloor$	$h > 0$
Tuščiosios eigos iteracijų limitas, L_{idle_iter}	$\lfloor 0{,}2\tau \rfloor$	$0 < L_{idle\_iter} \leq \tau$
Aplinkos dydžio parametras, ρ	0,4	$\rho > 0$
Randomizavimo koeficientas, α	0,02	$0 < \alpha < 1$
Mutacijos lygis TP procedūroje, μ	$\lfloor 0{,}15m \rfloor$	$0 < \mu < m$

[†] $Q = Q_1 = Q_2 = Q_3 = Q_4 = Q_5 = Q_6 = 2, Q_7 = 3.$



15 pav. HGA ir PGEA algoritmų veikimo santykinio laiko grafikas, kai *m* kinta nuo 30 iki 100

Reikia pažymėti, kad su kai kuriomis m reikšmėmis ($m = 26, 101, 102, 103$) rezultatų pagerinti nepavyko (3 lent.), tačiau 15 pav. pateikiamas santykinis lyginamų algoritmų laiko grafikas rodo, kad keliais atvejais, kai $m = 30, 31, 32, \ldots, 100$, vykdymo laiką pavyko pagerinti gana nemažai.

3 lentelė. Eksperimentų su HGA ir PGEA rezultatai (PŠF uždaviniui, $n = 256$)

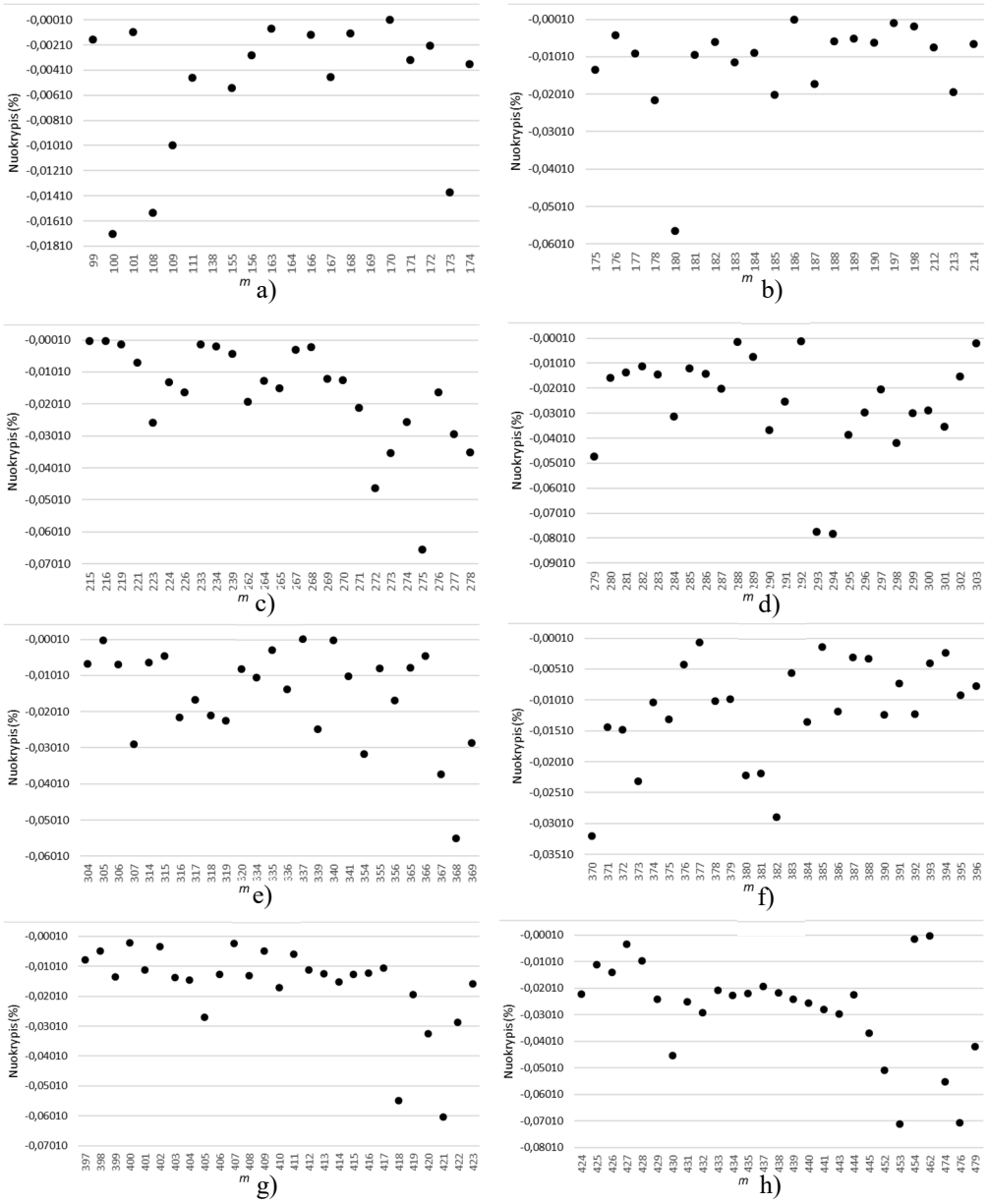
m	GŽR†	θ(%)††	CP laikas (s)		m	GŽR†	θ(%)††	CP laikas (s)		m	GŽR†	θ(%)††	CP laikas (s)	
			PGEA	HGA				PGEA	HGA				PGEA	HGA
2	1562	0	*0,0*	***0,00***	**45**	8674910	0	*150,0*	***5,91***	**87**	39389054	0	*25,0*	***0,19***
3	7810	0	*0,0*	***0,00***	**46**	9129192	0	*64,0*	***11,38***	**88**	40416536	0	*23,0*	***0,15***
4	15620	0	*0,0*	***0,00***	**47**	9575736	0	*3,1*	***0,29***	**89**	41512742	0	*183,0*	***6,98***
5	38072	0	*0,0*	***0,00***	**48**	10016256	0	*2,0*	***0,16***	**90**	42597626	0	*165,0*	***4,76***
6	63508	0	*0,0*	***0,00***	**49**	10518838	0	*3,4*	***0,56***	**91**	43676474	0	*224,0*	***17,64***
7	97178	0	*0,0*	***0,00***	**50**	11017342	0	*2,8*	***0,56***	**92**	44759294	0	*157,0*	***7,94***
8	131240	0	*0,0*	***0,00***	**51**	11516840	0	*7,5*	***0,83***	**93**	45870244	0	*214,0*	***19,18***
9	183744	0	*0,0*	***0,00***	**52**	12018388	0	*6,3*	***0,39***	**94**	46975856	0	*190,0*	***21,52***
10	242266	0	*0,0*	***0,00***	**53**	12558226	0	*4,6*	***0,54***	**95**	48081112	0	*169,0*	***2,14***
11	304722	0	*0,1*	***0,00***	**54**	13096646	0	*4,0*	***0,03***	**96**	49182368	0	*216,0*	***6,58***
12	368952	0	*0,1*	***0,00***	**55**	13661614	0	*10,1*	***0,12***	**97**	50344050	0	*213,0*	***23,88***
13	457504	0	*0,1*	***0,00***	**56**	14229492	0	*2,8*	***0,15***	**98**	51486642	0	*188,0*	***63,40***
14	547522	0	*0,1*	***0,00***	**57**	14793682	0	*2,2*	***0,37***	**99**	52660116	0	*201,0*	***50,53***
15	644036	0	*0,1*	***0,00***	**58**	15363628	0	*2,3*	***0,09***	**100**	53838088	0	*117,0*	***48,18***
16	742480	0	*0,1*	***0,00***	**59**	15981086	0	*3,5*	***0,71***	**101**	55014262	0	***84,0***	*156,00*
17	878888	0	*0,2*	***0,00***	**60**	16575644	0	*2,4*	***0,95***	**102**	56202826	0	***40,0***	*115,00*
18	1012990	0	*0,1*	***0,00***	**61**	17194812	0	*2,2*	***0,01***	**103**	57417112	0	***73,0***	*84,00*
19	1157992	0	*0,2*	***0,00***	**62**	17822806	0	*3,6*	***0,01***	**104**	58625240	0	*62,0*	***51,14***
20	1305744	0	*0,3*	***0,08***	**63**	18435790	0	*1,9*	***0,00***	**105**	59854744	0	*38,0*	***32,41***
21	1466210	0	*0,5*	***0,00***	**64**	19050432	0	*2,3*	***0,00***	**106**	61084902	0	*33,0*	***10,85***
22	1637794	0	*0,3*	***0,00***	**65**	19848790	0	*3,1*	***0,00***	**107**	62324634	0	*21,0*	***0,73***
23	1820052	0	*0,2*	***0,00***	**66**	20648754	0	*4,5*	***0,02***	**108**	63582416	0	*12,6*	***0,65***
24	2010846	0	*0,6*	***0,02***	**67**	21439396	0	*9,7*	***0,08***	**109**	64851966	0	*11,1*	***1,02***
25	2215714	0	*3,2*	***0,31***	**68**	22234020	0	*18,0*	***0,23***	**110**	66120434	0	*10,7*	***0,41***
26	2426298	0	***16,5***	*22,98*	**69**	23049732	0	*27,0*	***0,64***	**111**	67392724	0	*8,2*	***0,46***
27	2645436	0	*1,1*	***0,02***	**70**	23852796	0	*26,0*	***0,98***	**112**	68666416	0	*7,7*	***0,17***
28	2871704	0	*0,9*	***0,03***	**71**	24693608	0	*78,0*	***0,43***	**113**	69984758	0	*10,2*	***0,13***
29	3122510	0	*0,7*	***0,03***	**72**	25522408	0	*490,0*	***64,80***	**114**	71304194	0	*6,3*	***0,18***
30	3373854	0	*0,5*	***0,00***	**73**	26375828	0	*298,0*	***5,81***	**115**	72630764	0	*5,1*	***0,37***
31	3646344	0	*0,6*	***0,00***	**74**	27235240	0	*304,0*	***2,50***	**116**	73962220	0	*5,3*	***0,21***
32	3899744	0	*0,5*	***0,02***	**75**	28114952	0	*41,0*	***0,85***	**117**	75307424	0	*4,0*	***0,03***
33	4230950	0	*0,7*	***0,03***	**76**	29000908	0	*121,0*	***1,30***	**118**	76657014	0	*3,6*	***0,06***
34	4560162	0	*2,6*	***0,36***	**77**	29894452	0	*145,0*	***4,81***	**119**	78015914	0	*2,3*	***0,03***
35	4890132	0	*3,2*	***0,41***	**78**	30797954	0	*117,0*	***1,15***	**120**	79375832	0	*1,7*	***0,05***
36	5222296	0	*2,0*	***0,44***	**79**	31702182	0	*11,6*	***0,81***	**121**	80756852	0	*1,6*	***0,07***
37	5565236	0	*1,8*	***0,34***	**80**	32593088	0	*3,3*	***0,47***	**122**	82138768	0	*1,4*	***0,03***
38	5909202	0	*0,9*	***0,14***	**81**	33544628	0	*3,9*	***0,26***	**123**	83528554	0	*1,0*	***0,04***
39	6262248	0	*1,1*	***0,08***	**82**	34492592	0	*70,0*	***3,33***	**124**	84920540	0	*0,7*	***0,01***
40	6613472	0	*0,9*	***0,02***	**83**	35443938	0	*57,0*	***1,41***	**125**	86327812	0	*0,4*	***0,00***
41	7002794	0	*0,6*	***0,11***	**84**	36395172	0	*61,0*	***2,77***	**126**	87736646	0	*0,3*	***0,00***
42	7390586	0	*0,7*	***0,16***	**85**	37378800	0	*151,0*	***1,20***	**127**	89150166	0	*0,2*	***0,00***
43	7794422	0	*3,2*	***0,20***	**86**	38376438	0	*94,0*	***0,32***	**128**	90565248	0	*0,2*	***0,00***
44	8217264	0	*16,0*	***0,87***										

Pastaba. Lentelėje paryškintuoju šriftu pateikti rezultatai, kuriuos pavyko pagerinti.
† – GŽR pateikiamos straipsnyje [20].
†† – nuokrypis θ apskaičiuotas pagal formulę: $\theta = (z^* - G\check{Z}R)/G\check{Z}R$; čia z^* – algoritmo rasta geriausia tikslo funkcijos reikšmė.

4 lentelė. Eksperimentų su HGA rezultatai (PŠF uždaviniui, $n = 1024$)

m	GŽR	*m*	GŽR	*m*	GŽR	*m*	GŽR	*m*	GŽR	*m*	GŽR	*m*	GŽR	*m*	GŽR
2	390	66	5132250	130	23460170	194	57086766	258	106260632	322	176518174	386	**262819150**	450	363665156
3	1954	67	5312762	131	23897592	195	57739124	259	107273354	323	177756358	387	**264317294**	451	365361310
4	3908	68	5493398	132	24335656	196	58392270	260	108286116	324	178959204	388	**265791714**	452	**367056444**
5	9488	69	5675784	133	24773832	197	**59055298**	261	109299348	325	180158842	389	267231448	453	**368754728**
6	15882	70	5868614	134	25213730	198	**59710314**	262	**110313464**	326	181377560	390	**268698768**	454	**370451410**
7	24290	71	6061636	135	25653062	199	60357328	263	111323160	327	182598340	391	**270205824**	455	372157104
8	32808	72	6253544	136	26091040	200	61005880	264	**112349256**	328	183826382	392	**271688930**	456	373863658
9	45844	73	6451748	137	26536474	201	61656140	265	**113366358**	329	185039942	393	**273203922**	457	375575430
10	60310	74	6658646	138	**26980064**	202	62313154	266	114381534	330	186245648	394	**274660438**	458	377289052
11	75878	75	6866464	139	27426740	203	62979360	267	**115403394**	331	187480294	395	**276180114**	459	379005274
12	91852	76	7077272	140	27873238	204	63648372	268	**116415772**	332	188702624	396	**277670456**	460	380724964
13	114040	77	7287952	141	28319430	205	64329116	269	**117454596**	333	189934056	397	**279172978**	461	382449898
14	136706	78	7497962	142	28761578	206	65021762	270	**118462682**	334	**191134986**	398	**280677404**	462	**384174962**
15	160770	79	7708934	143	29211334	207	65721964	271	**119472414**	335	**192370030**	399	**282179032**	463	385902750
16	185552	80	7919112	144	29649520	208	66422364	272	**120505424**	336	**193596576**	400	**283712492**	464	387628048
17	218392	81	8147012	145	30118164	209	67136312	273	**121574572**	337	**194844496**	401	**285211916**	465	389368514
18	251618	82	8363950	146	30588480	210	67852496	274	**122629730**	338	196104848	402	**286746076**	466	391105794
19	288006	83	8600584	147	31065948	211	68585494	275	**123626322**	339	**197349714**	403	**288238662**	467	392843892
20	324794	84	8839620	148	31546098	212	**69315338**	276	**124716364**	340	**198600114**	404	**289784468**	468	394587900
21	365546	85	9079818	149	32025690	213	**70050648**	277	**125741472**	341	**199870596**	405	**291260654**	469	396332818
22	407406	86	9322672	150	32508848	214	**70789536**	278	**126807112**	342	201181102	406	**292833208**	470	398083462
23	451448	87	9563920	151	32992712	215	**71527862**	279	**127842588**	343	202519622	407	**294387042**	471	399837320
24	496888	88	9818424	152	33479148	216	**72259864**	280	**128937048**	344	203834574	408	**295934120**	472	401592882
25	549180	89	10074140	153	33968988	217	72990332	281	**130003342**	345	205167090	409	**297474388**	473	403351666
26	603368	90	10331422	154	34461110	218	73726832	282	**131077338**	346	206544390	410	**298998414**	474	**405112104**
27	659044	91	10600710	155	**34955468**	219	**74466810**	283	**132132040**	347	207925194	411	**300564200**	475	406875344
28	716280	92	10871062	156	**35450196**	220	75201458	284	**133187012**	348	209234904	412	**302129606**	476	**408637620**
29	777436	93	11138470	157	35944108	221	**75953890**	285	**134287484**	349	210612178	413	**303688980**	477	410409038
30	837798	94	11411510	158	36437606	222	76713866	286	**135364426**	350	211922934	414	**305224788**	478	412182568
31	907090	95	11679880	159	36933614	223	**77465610**	287	**136441394**	351	213304876	415	**306845268**	479	**413959340**
32	975008	96	11944352	160	37426912	224	**78218352**	288	**137549224**	352	214686716	416	**308393388**	480	415733856
33	1050792	97	12237102	161	37947342	225	78977922	289	**138637260**	353	216044260	417	**309969764**	481	417519180
34	1125558	98	12523996	162	38464394	226	**79744456**	290	**139677068**	354	**217376574**	418	**311420318**	482	419302686
35	1203646	99	**12813836**	163	**38982592**	227	80520900	291	**140801272**	355	**218738658**	419	**313094242**	483	421092758
36	1281132	100	**13103420**	164	**39500208**	228	81287994	292	**141875610**	356	**220109066**	420	**314652760**	484	422883164
37	1368444	101	**13398254**	165	40025416	229	82061894	293	**142916356**	357	221526988	421	**316172166**	485	424678088
38	1456842	102	13691306	166	**40550006**	230	82837128	294	**144026694**	358	222888300	422	**317825280**	486	426473544
39	1547598	103	13988062	167	**41078930**	231	83613898	295	**145175322**	359	224268836	423	**319428868**	487	428272184
40	1638808	104	14288780	168	**41606240**	232	84406568	296	**146290048**	360	225646838	424	**321022500**	488	430071632
41	1736236	105	14593444	169	**42140968**	233	**85225404**	297	**147454448**	361	227020504	425	**322637088**	489	431876322
42	1834074	106	14899130	170	**42673974**	234	**86030804**	298	**148527002**	362	228390592	426	**324266210**	490	433683572
43	1935946	107	15216394	171	**43219476**	235	86829778	299	**149672540**	363	229827232	427	**325898680**	491	435492454
44	2042792	108	**15537796**	172	**43787404**	236	87618540	300	**150827224**	364	231201722	428	**327457994**	492	437303524
45	2147200	109	**15857934**	173	**44361196**	237	88445972	301	**151952048**	365	**232578308**	429	**328993270**	493	439118116
46	2260650	110	16177106	174	**44933388**	238	89239062	302	**153122860**	366	**233960416**	430	**330616714**	494	440933678
47	2373506	111	**16504524**	175	**45511224**	239	**90075414**	303	**154282700**	367	**235302078**	431	**332246332**	495	442752278
48	2482832	112	16837956	176	**46091442**	240	90875504	304	**155417120**	368	**236712932**	432	**333874768**	496	444570032
49	2607474	113	17174378	177	**46680202**	241	91698908	305	**156547208**	369	**238200964**	433	**335514106**	497	446397066
50	2730510	114	17508602	178	**47274350**	242	92523578	306	**157684310**	370	**239582944**	434	**337154026**	498	448224550
51	2857088	115	17849756	179	47871440	243	93371894	307	**158836480**	371	**241044336**	435	**338796402**	499	450053654
52	2988998	116	18191920	180	**48462430**	244	94187252	308	159999648	372	**242479798**	436	340435998	500	451883116
53	3120248	117	18535442	181	**49056670**	245	95044544	309	161157378	373	**243893396**	437	**342076542**	501	453718668
54	3257234	118	18902942	182	**49654614**	246	95865322	310	162344014	374	**245361204**	438	**343718980**	502	455554546
55	3398018	119	19272770	183	**50258968**	247	96720682	311	163515706	375	**246783270**	439	**345362184**	503	457391626
56	3535048	120	19631156	184	**50876864**	248	97531736	312	164641724	376	**248246874**	440	**347009592**	504	459230104
57	3684478	121	20001764	185	**51494526**	249	98361638	313	165838280	377	**249710800**	441	**348669786**	505	461073188
58	3829950	122	20370638	186	**52115066**	250	99225594	314	**167015058**	378	**251141622**	442	350325606	506	462916382
59	3984538	123	20746696	187	**52731636**	251	100062350	315	**168180928**	379	**252535872**	443	**351981700**	507	464761614
60	4136400	124	21117234	188	**53348334**	252	100898116	316	**169354960**	380	**254011358**	444	**353638456**	508	466607612
61	4291962	125	21484868	189	**53959660**	253	101733670	317	**170529852**	381	**255486362**	445	**355296090**	509	468457260
62	4447434	126	21852518	190	**54571808**	254	102566006	318	**171723964**	382	**256914322**	446	356954184	510	470307298
63	4604860	127	22218924	191	55185346	255	103399158	319	**172910768**	383	**258421702**	447	358612252	511	472158510
64	4762688	128	22581376	192	55788864	256	104232704	320	**174113066**	384	**259861698**	448	360270272	512	474010112
65	4949042	129	23021790	193	56452088	257	105247082	321	175343494	385	**261377866**	449	361968220		

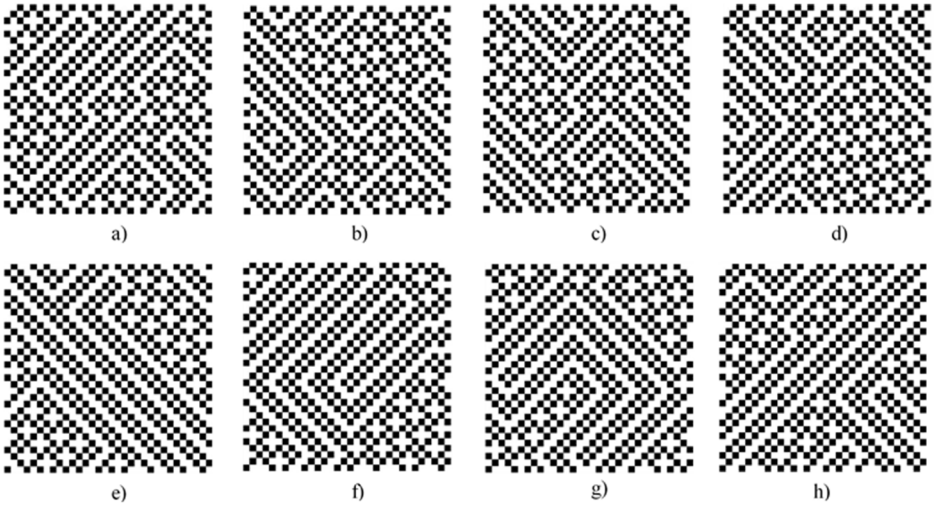
Iki šiol geriausios žinomos tikslo funkcijos reikšmės, kai PŠF uždavinio dydis yra $n = 1024$, pateiktos Misevičiaus, Guogio ir Stanevičienės [19]. Pagerintų tikslo funkcijos reikšmių nuokrypis (proc.) pateikiamas 16 paveiksle.



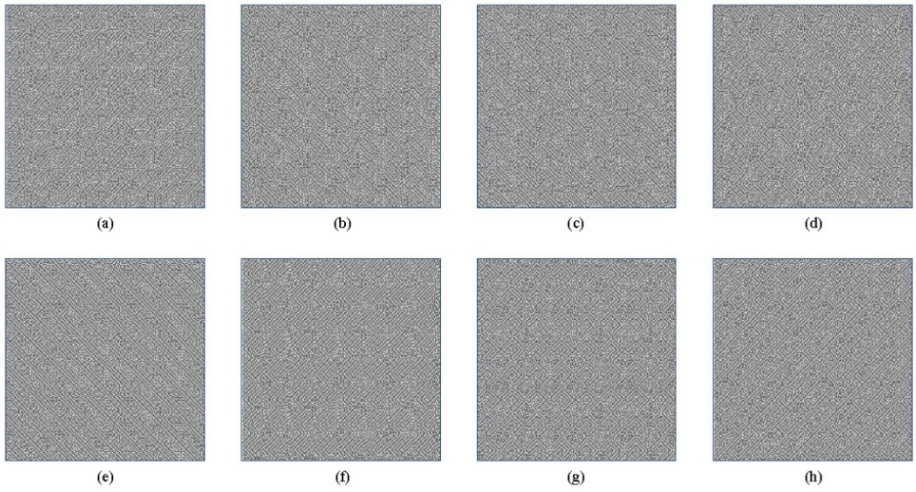
16 pav. Pagerintų tikslo funkcijos reikšmių nuokrypis (%) nuo GŽR ($n = 1024$): a) m kinta nuo 99 iki 174; b) m kinta nuo 175 iki 214; c) m kinta nuo 215 iki 278; d) m kinta nuo 279 iki 303; e) m kinta nuo 304 iki 369; f) m kinta nuo 370 iki 396; g) m kinta nuo 397 iki 423; h) m kinta nuo 424 iki 479

16 pav. pateiktuose grafikuose matyti, kad kuo didesnė absoliutinė nuokrypio reikšmė, tuo geresnis sprendinys pasiektas.

Keletas rastų galimai optimalių (pseudooptimalių) sprendinių (perstatymų) yra atvaizduoti grafiškai esant skirtingiems masteliams (17, 18 pav.). Spalvos intensyvumo m reikšmės yra lygios: $401, 402, 403, 404,\ 405, 406, 407, 408$, PŠF uždavinio dydis $n = 1024$.



17 pav. (Pseudo)optimalių pilkųjų šablonų pavyzdžiai (32×32): a) $m = 401$; b) $m = 402$; c) $m = 403$; d) $m = 404$; e) $m = 405$; f) $m = 406$; g) $m = 407$; h) $m = 408$



18 pav. (Pseudo)optimalių pilkųjų šablonų pavyzdžiai (32×32)[10]: a) $m = 401$; b) $m = 402$; c) $m = 403$; d) $m = 404$; e) $m = 405$; f) $m = 406$; g) $m = 407$; h) $m = 408$

[10] Šiame paveikslėlyje kiekvienas $n = 1024$ dydžio tinklelis pakartotas 8 kartus vertikaliai ir 8 kartus horizontaliai.

3.3. Algoritmo komponentų eksperimentinių tyrimų rezultatai

Siekiant nustatyti efektyviausias HGA modifikacijas, buvo atlikti išplėstiniai eksperimentiniai tyrimai su pagrindiniais algoritmo komponentais. Šie eksperimentai buvo atlikti su PŠF uždaviniais, kai $n = 256$, $m = 95, 96, \dots, 104$ (preliminariai nustatyti kaip sunkūs duomenų pavyzdžiai) ir $n = 1024$, $m = 50, 60, 70, \dots, 110$. Esant didelio dydžio uždaviniui, didesnės parametro m reikšmės nebuvo naudojamos išplėstiniuose eksperimentuose dėl labai didelio skaičiavimo laiko. Vykdant šiuos eksperimentus, buvo lyginamas vidutinis nuokrypis nuo geriausios žinomos reikšmės ir sėkmingų bandymų skaičius (angl. *succsess rate*). Vidutinis nuokrypis buvo apskaičiuojamas pagal formulę: $\theta = 100\frac{z - z_{GŽR}}{z_{GŽR}}[\%]$; čia z yra vidutinė tikslo funkcijos reikšmė per 10 vykdymų, $z_{GŽR}$ – geriausia žinoma tikslo funkcijos reikšmė. Sėkmingų bandymų skaičius nurodo, kiek kartų per 10 algoritmo vykdymų pavyko rasti GŽR.

Pateikiami šių eksperimentų rezultatai: su komponento „PRADINĖ POPULIACIJA“ modifikacijomis (5, 6 lent., 19, 20 pav.); su komponento „LOKALUSIS PAGERINIMAS“ modifikacijomis (7, 8 lent., 21–23 pav.); su komponento „SPRENDINIŲ-TĖVŲ ATRANKA“ modifikacijomis (9, 10 lent., 24, 25 pav.); su komponento „KRYŽMINIMO PROCEDŪRA“ modifikacijomis (11–16 lent., 26, 27 pav.); su komponento „KRYŽMINIMŲ SKAIČIUS“ modifikacijomis (17, 18 lent., 28 pav.); su komponento „POPULIACIJOS ATNAUJINIMAS“ modifikacijomis (19–22 lent., 29–31 pav.); su komponento „POPULIACIJOS PERTVARKYMAS“ modifikacijomis (23–26 lent., 32, 33 pav.).

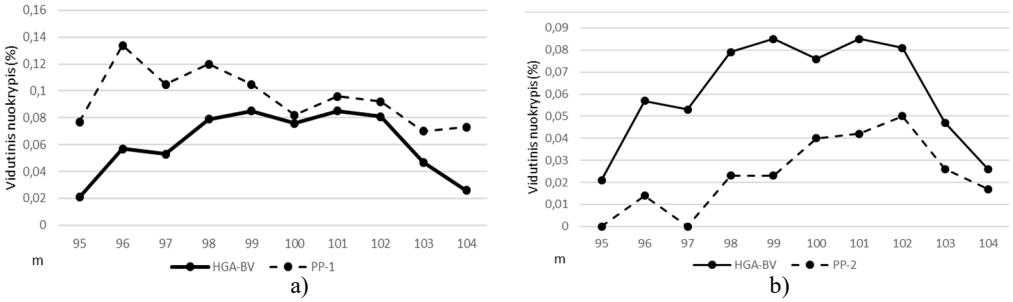
Kiekviena algoritmo komponento modifikacija pateiktuose eksperimentinių tyrimų rezultatuose žymima santrumpomis: **`PP-1`**, **`PP-2`** (žr. 2.5.2 skyrelį), **`LP-1`**, **`LP-2`**, **`LP-3`**, **`LP-4`** (žr. 2.5.3 skyrelį), **`STA-1`**, **`STA-2`** (žr. 2.5.4 skyrelį), **`KP-1`**, **`KP-2`**, **`KP-3`**, **`KP-4`**, **`KP-5`**, **`KP-6`**, **`KP-7`**, **`KP-8`**, **`KP-9`** (žr. 2.5.5 skyrelį), **`KS-1`** (žr. 2.5.6 skyrelį), **`PA-1`**, **`PA-2`**, **`PA-3`**, **`PA-4`** (žr. 2.5.7 skyrelį), **`PR-1`**, **`PR-2`**, **`PR-3`**, **`PR-4`**, **`PR-5`**, **`PR-6`** (žr. 2.5.8 skyrelį). Bazinės HGA versijos (**`HGA-BV`**) parametrai pateikti 1 lentelėje.

5 lentelė. Eksperimentų su komponento „PRADINĖ POPULIACIJA“ modifikacijomis rezultatai ($n = 256$)

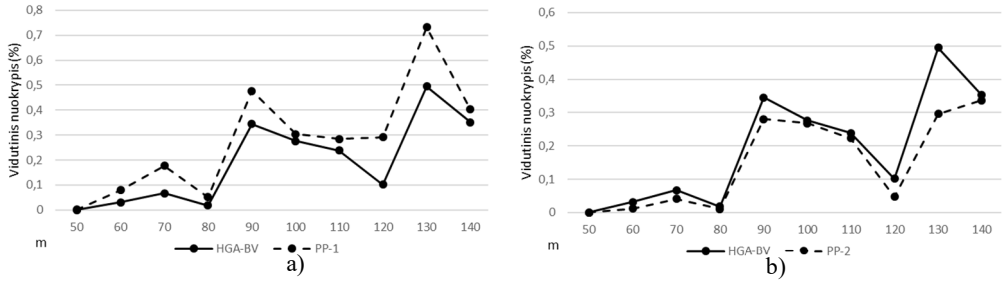
		HGA-BV		PP-1		PP-2	
m	GŽR	Vidutinis nuokrypis (%)	Sėkmingų bandymų skaičius	Vidutinis nuokrypis (%)	Sėkmingų bandymų skaičius	Vidutinis nuokrypis (%)	Sėkmingų bandymų skaičius
95	48081112	0,021	4	0,077	0	0,000	10
96	49182368	0,057	3	0,134	0	0,014	8
97	50344050	0,053	1	0,105	0	0,000	10
98	51486642	0,079	1	0,120	0	0,023	7
99	52660116	0,085	0	0,105	0	0,023	6
100	53838088	0,076	0	0,082	0	0,040	1
101	55014262	0,085	0	0,096	0	0,042	2
102	56202826	0,081	0	0,092	0	0,050	0
103	57417112	0,047	0	0,070	0	0,026	0
104	58625240	0,026	0	0,073	0	0,017	1
	Vidurkis:	0,061	0,9	0,095	0	0,024	4,5

6 lentelė. Eksperimentų su komponento „PRADINĖ POPULIACIJA“ modifikacijomis rezultatai ($n = 1024$)

		HGA-BV		PP-1		PP-2	
m	GŽR	Vidutinis nuokrypis (%)	Sėkmingų bandymų skaičius	Vidutinis nuokrypis (%)	Sėkmingų bandymų skaičius	Vidutinis nuokrypis (%)	Sėkmingų bandymų skaičius
50	2730510	0,000	10	0,002	7	0,000	10
60	4136400	0,031	0	0,080	0	0,012	0
70	5868614	0,067	0	0,177	0	0,041	0
80	7919112	0,018	0	0,052	0	0,011	0
90	10331422	0,345	0	0,477	0	0,280	0
100	13103420	0,276	0	0,304	0	0,268	0
110	16177106	0,239	0	0,284	0	0,223	0
120	19631156	0,102	0	0,292	0	0,047	0
130	23460170	0,495	0	0,732	0	0,296	0
140	27873238	0,352	0	0,403	0	0,337	0
	Vidurkis:	0,193	1	0,280	0,7	0,152	1



19 pav. Vidutinis nuokrypis (%) ($n = 256$): a) **`HGA-BV`** ir **`PP-1`**; b) **`HGA-BV`** ir **`PP-2`**

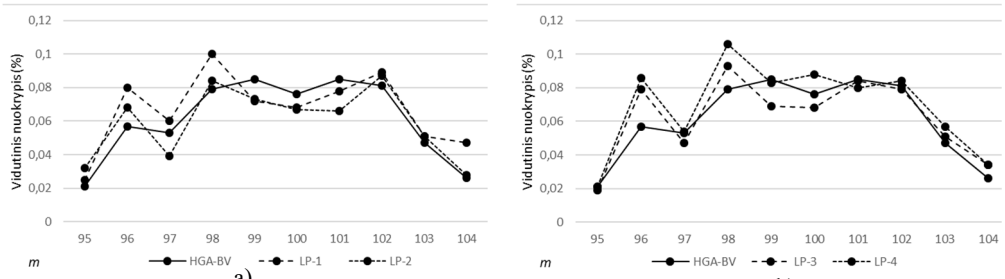


20 pav. Vidutinis nuokrypis (%) ($n = 1024$): a) **`HGA-BV`** ir **`PP-1`**; b) **`HGA-BV`** ir **`PP-2`**

7 lentelė. Eksperimentų su komponento „LOKALUSIS PAGERINIMAS“ modifikacijomis rezultatai ($n = 256$)

		`HGA-BV`		**`LP-1`**		**`LP-2`**		**`LP-3`**		**`LP-4`**	
m	GŽR	Vidutinis nuokrypis (%)	*SB*[†]	Vidutinis nuokrypis (%)	*SB*	Vidutinis nuokrypis (%)	*SB*	Vidutinis nuokrypis (%)	*SB*	Vidutinis nuokrypis (%)	*SB*
95	48081112	0,021	4	0,025	3	0,032	1	0,019	3	0,021	3
96	49182368	0,057	3	0,080	0	0,068	2	0,079	0	0,086	0
97	50344050	0,053	1	0,060	2	0,039	3	0,047	1	0,054	0
98	51486642	0,079	1	0,100	0	0,084	1	0,093	0	0,106	0
99	52660116	0,085	0	0,072	0	0,073	0	0,069	0	0,083	0
100	53838088	0,076	0	0,068	0	0,067	0	0,068	0	0,088	0
101	55014262	0,085	0	0,078	0	0,066	0	0,084	0	0,080	0
102	56202826	0,081	0	0,089	0	0,087	0	0,079	0	0,084	0
103	57417112	0,047	0	0,051	1	0,051	0	0,051	0	0,057	0
104	58625240	0,026	0	0,047	0	0,028	0	0,034	0	0,034	0
	Vidurkis:	0,061	0,9	0,067	0,6	0,060	0,7	0,062	0,4	0,069	0,3

[†] *SB* yra sėkmingų bandymų skaičius (vnt.).

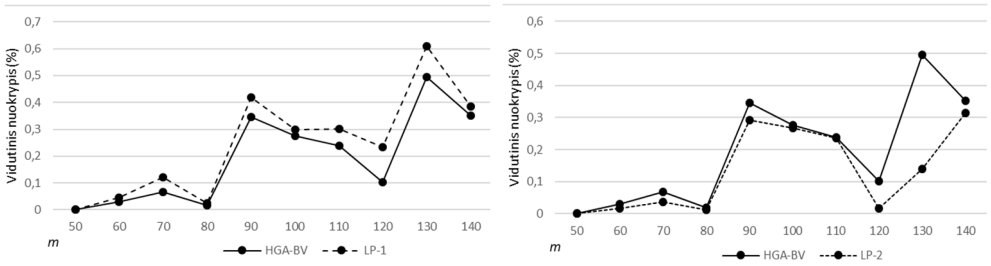


a) b)

21 pav. Vidutinis nuokrypis (%) ($n = 256$): a) **`HGA-BV`**, **`LP-1`** ir **`LP-2`** modifikacijos; b) **`HGA-BV`**, **`LP-3`** ir **`LP-4`** modifikacijos

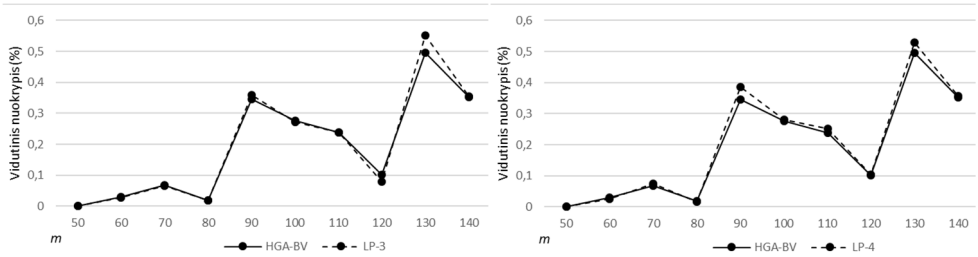
8 lentelė. Eksperimentų su komponento „LOKALUSIS PAGERINIMAS“ modifikacijomis rezultatai ($n = 1024$)

		`HGA-BV`		**`LP-1`**		**`LP-2`**		**`LP-3`**		**`LP-4`**	
m	GŽR	Vidutinis nuokrypis (%)	*SB*	Vidutinis nuokrypis (%)	*SB*	Vidutinis nuokrypis (%)	*SB*	Vidutinis nuokrypis (%)	*SB*	Vidutinis nuokrypis (%)	*SB*
50	2730510	0,000	10	0,000	10	0,000	10	0,000	10	0,000	10
60	4136400	0,031	0	0,046	0	0,017	0	0,027	0	0,025	0
70	5868614	0,067	0	0,120	0	0,037	0	0,065	0	0,074	0
80	7919112	0,018	0	0,025	0	0,012	0	0,018	0	0,017	0
90	10331422	0,345	0	0,420	0	0,291	0	0,358	0	0,385	0
100	13103420	0,276	0	0,300	0	0,267	0	0,271	0	0,281	0
110	16177106	0,239	0	0,302	0	0,237	0	0,238	0	0,252	0
120	19631156	0,102	0	0,233	0	0,016	1	0,079	0	0,105	0
130	23460170	0,495	0	0,609	0	0,140	0	0,552	0	0,529	0
140	27873238	0,352	0	0,385	0	0,315	0	0,355	0	0,356	0
	Vidurkis:	0,193	1	0,244	1	0,133	1,1	0,196	1	0,202	1



a) b)

22 pav. Vidutinis nuokrypis (%) ($n = 1024$): a) **`HGA-BV`** ir **`LP-1`**; b) **`HGA-BV`** ir **`LP-2`**

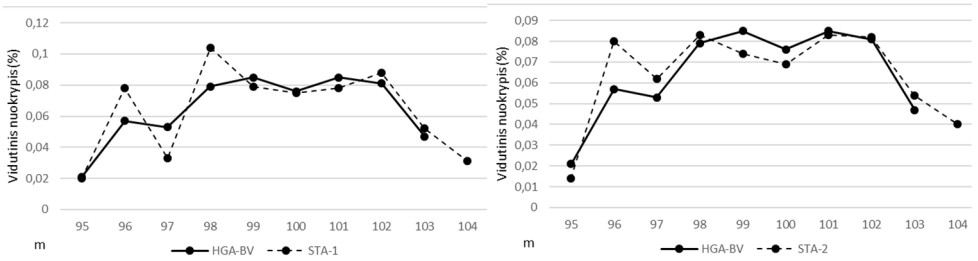


a) b)

23 pav. Vidutinis nuokrypis (%) ($n = 1024$): a) **`HGA-BV`** ir **`LP-3`**; b) **`HGA-BV`** ir **`LP-4`**

9 lentelė. Eksperimentų su komponento „SPRENDINIŲ-TĖVŲ ATRANKA“ modifikacijomis rezultatai ($n = 256$)

		`HGA-BV`		**`STA-1`**		**`STA-2`**	
m	GŽR	Vidutinis nuokrypis (%)	Sėkmingų bandymų skaičius	Vidutinis nuokrypis (%)	Sėkmingų bandymų skaičius	Vidutinis nuokrypis (%)	Sėkmingų bandymų skaičius
95	48081112	0,021	4	0,020	6	0,014	5
96	49182368	0,057	3	0,078	0	0,080	0
97	50344050	0,053	1	0,033	4	0,062	0
98	51486642	0,079	1	0,104	0	0,083	0
99	52660116	0,085	0	0,079	0	0,074	0
100	53838088	0,076	0	0,075	0	0,069	0
101	55014262	0,085	0	0,078	0	0,083	0
102	56202826	0,081	0	0,088	0	0,082	0
103	57417112	0,047	0	0,052	0	0,054	0
104	58625240	0,026	0	0,031	0	0,040	0
	Vidurkis:	0,061	0,9	0,064	1	0,064	0,5

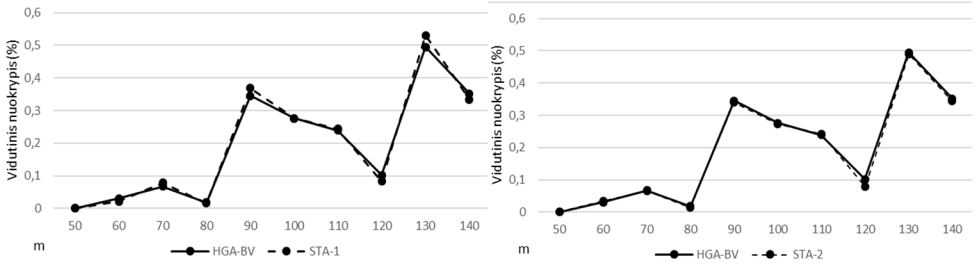


a) b)

24 pav. Vidutinis nuokrypis (%) ($n = 256$): a) **`HGA-BV`** ir **`STA-1`**; b) **`HGA-BV`** ir **`STA-2`**

10 lentelė. Eksperimentų su komponento „SPRENDINIŲ-TĖVŲ ATRANKA“ modifikacijomis rezultatai ($n = 1024$)

		HGA-BV		**STA-1**		**STA-2**	
m	GŽR	Vidutinis nuokrypis (%)	Sėkmingų bandymų skaičius	Vidutinis nuokrypis (%)	Sėkmingų bandymų skaičius	Vidutinis nuokrypis (%)	Sėkmingų bandymų skaičius
50	2730510	0,000	10	0,000	10	0,000	0
60	4136400	0,031	0	0,022	0	0,035	0
70	5868614	0,067	0	0,079	0	0,065	0
80	7919112	0,018	0	0,016	0	0,015	1
90	10331422	0,345	0	0,369	0	0,340	0
100	13103420	0,276	0	0,276	0	0,273	0
110	16177106	0,239	0	0,244	0	0,242	0
120	19631156	0,102	0	0,083	0	0,078	0
130	23460170	0,495	0	0,530	0	0,490	0
140	27873238	0,352	0	0,334	0	0,345	0
	Vidurkis:	0,193	1	0,195	1	0,188	0,1



a) b)

25 pav. Vidutinis nuokrypis (%) ($n = 1024$): a) **HGA-BV** ir **STA-1**; b) **HGA-BV** ir **STA-2**

11 lentelė. Eksperimentų su komponento „KRYŽMINIMO PROCEDŪRA“ modifikacijomis rezultatai ($n = 256$, **`KP-1`**, **`KP-2`**, **`KP-3`**)

		`HGA-BV`		**`KP-1`**		**`KP-2`**		**`KP-3`**	
m	GŽR	Vidutinis nuokrypis (%)	*SB*	Vidutinis nuokrypis (%)	*SB*	Vidutinis nuokrypis (%)	*SB*	Vidutinis nuokrypis (%)	*SB*
95	48081112	0,021	4	0,011	6	0,017	5	0,005	8
96	49182368	0,057	3	0,078	1	0,085	0	0,074	0
97	50344050	0,053	1	0,048	2	0,031	4	0,046	2
98	51486642	0,079	1	0,095	0	0,086	0	0,086	0
99	52660116	0,085	0	0,074	0	0,078	0	0,071	0
100	53838088	0,076	0	0,069	0	0,077	0	0,056	1
101	55014262	0,085	0	0,079	0	0,086	0	0,076	0
102	56202826	0,081	0	0,088	0	0,078	0	0,071	0
103	57417112	0,047	0	0,052	0	0,067	0	0,051	0
104	58625240	0,026	0	0,021	0	0,034	0	0,028	0
	Vidurkis:	0,061	0,9	0,062	0,9	0,064	0,9	0,056	1,1

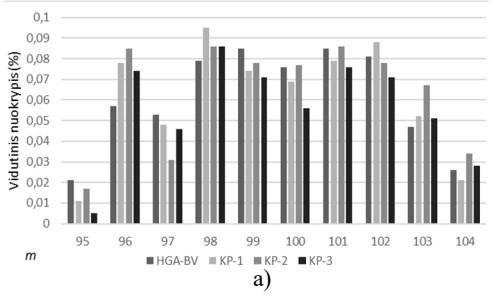
12 lentelė. Eksperimentų su komponento „KRYŽMINIMO PROCEDŪRA“ modifikacijomis rezultatai ($n = 256$, **`KP-4`**, **`KP-5`**, **`KP-6`**)

		`HGA-BV`		**`KP-4`**		**`KP-5`**		**`KP-6`**	
m	GŽR	Vidutinis nuokrypis (%)	*SB*	Vidutinis nuokrypis (%)	*SB*	Vidutinis nuokrypis (%)	*SB*	Vidutinis nuokrypis (%)	*SB*
95	48081112	0,021	4	0,027	4	0,021	3	0,021	3
96	49182368	0,057	3	0,061	3	0,076	1	0,080	1
97	50344050	0,053	1	0,066	0	0,031	4	0,061	1
98	51486642	0,079	1	0,092	0	0,065	2	0,083	1
99	52660116	0,085	0	0,075	0	0,072	0	0,077	0
100	53838088	0,076	0	0,062	1	0,071	0	0,075	0
101	55014262	0,085	0	0,069	0	0,065	0	0,077	0
102	56202826	0,081	0	0,083	0	0,084	0	0,081	0
103	57417112	0,047	0	0,051	0	0,049	0	0,055	0
104	58625240	0,026	0	0,028	0	0,028	0	0,041	0
	Vidurkis:	0,061	0,9	0,061	0,8	0,056	1	0,065	0,6

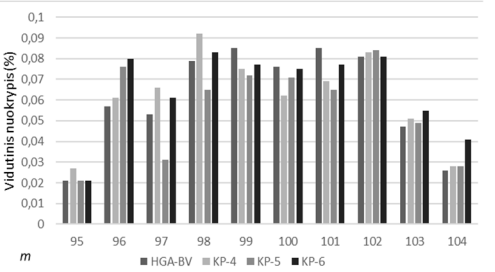
13 lentelė. Eksperimentų su komponento „KRYŽMINIMO PROCEDŪRA“ modifikacijomis rezultatai ($n = 256$, **KP-7**, **KP-8**, **KP-9**)

		HGA-BV		**KP-7**		**KP-8**		**KP-9**	
m	GŽR	Vidutinis nuokrypis (%)	*SB*	Vidutinis nuokrypis (%)	*SB*	Vidutinis nuokrypis (%)	*SB*	Vidutinis nuokrypis (%)	*SB*
95	48081112	0,021	4	0,015	4	0,017	3	0,014	3
96	49182368	0,057	3	0,077	0	0,090	0	0,076	0
97	50344050	0,053	1	0,039	2	0,055	0	0,052	1
98	51486642	0,079	1	0,082	0	0,091	0	0,088	0
99	52660116	0,085	0	0,069	0	0,088	0	0,067	0
100	53838088	0,076	0	0,054	0	0,063	0	0,062	0
101	55014262	0,085	0	0,075	0	0,084	0	0,065	0
102	56202826	0,081	0	0,071	0	0,082	0	0,083	0
103	57417112	0,047	0	0,052	0	0,058	0	0,036	0
104	58625240	0,026	0	0,026	0	0,031	0	0,034	0
	Vidurkis:	0,061	0,9	0,056	0,6	0,066	0,3	0,058	0,4

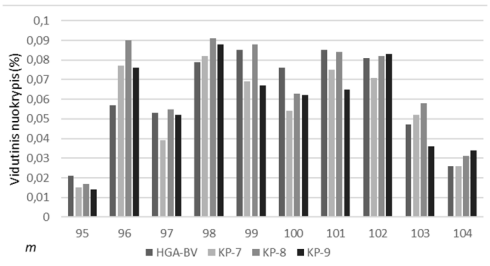
26 pav. Vidutinis nuokrypis (%) ($n = 256$): a) **HGA-BV**, **KP-1**, **KP-2**, **KP-3**; b) **HGA-BV**, **KP-4**, **KP-5**, **KP-6**; c) **HGA-BV**, **KP-7**, **KP-8**, **KP-9**



a)



b)



c)

14 lentelė. Eksperimentų su komponento „KRYŽMINIMO PROCEDŪRA“ modifikacijomis rezultatai ($n = 1024$, **`KP-1`**, **`KP-2`**, **`KP-3`**)

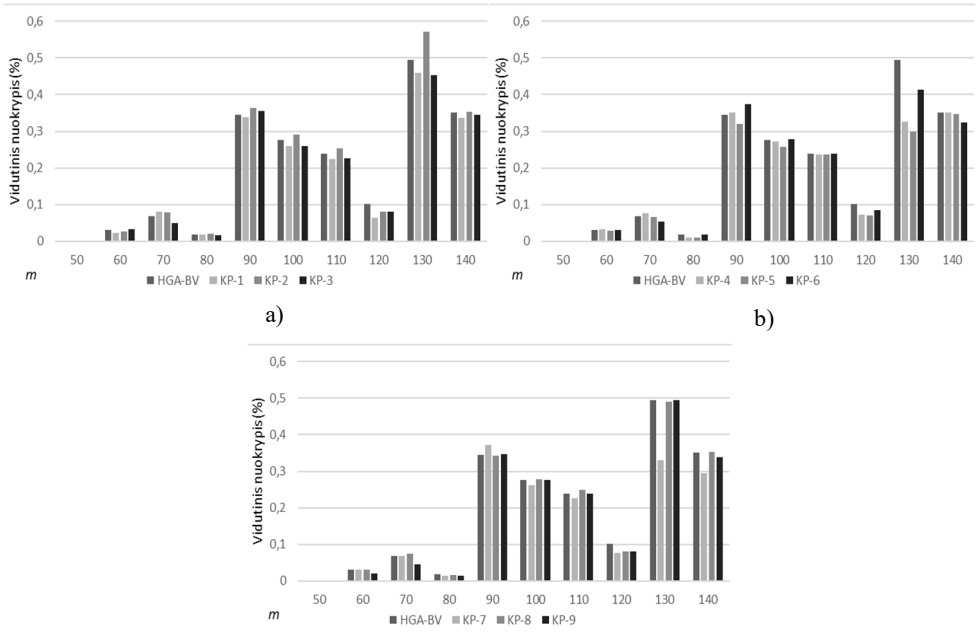
		`HGA-BV`		**`KP-1`**		**`KP-2`**		**`KP-3`**	
m	GŽR	Vidutinis nuokrypis (%)	*SB*	Vidutinis nuokrypis (%)	*SB*	Vidutinis nuokrypis (%)	*SB*	Vidutinis nuokrypis (%)	*SB*
50	2730510	0,000	10	0,000	10	0,000	10	0,000	10
60	4136400	0,031	0	0,022	0	0,027	0	0,033	0
70	5868614	0,067	0	0,081	0	0,079	0	0,050	0
80	7919112	0,018	0	0,017	0	0,019	0	0,015	0
90	10331422	0,345	0	0,338	0	0,363	0	0,355	0
100	13103420	0,276	0	0,260	0	0,291	0	0,260	0
110	16177106	0,239	0	0,225	0	0,254	0	0,227	0
120	19631156	0,102	0	0,063	0	0,080	0	0,081	0
130	23460170	0,495	0	0,460	0	0,571	0	0,454	0
140	27873238	0,352	0	0,336	0	0,354	0	0,344	0
	Vidurkis:	0,193	1	0,18	1	0,204	1	0,182	1

15 lentelė. Eksperimentų su komponento „KRYŽMINIMO PROCEDŪRA“ modifikacijomis rezultatai ($n = 1024$, **`KP-4`**, **`KP-5`**, **`KP-6`**)

		`HGA-BV`		**`KP-4`**		**`KP-5`**		**`KP-6`**	
m	GŽR	Vidutinis nuokrypis (%)	*SB*	Vidutinis nuokrypis (%)	*SB*	Vidutinis nuokrypis (%)	*SB*	Vidutinis nuokrypis (%)	*SB*
50	2730510	0,000	10	0,000	10	0,000	0	0,000	10
60	4136400	0,031	0	0,033	0	0,028	0	0,030	0
70	5868614	0,067	0	0,077	0	0,065	0	0,053	0
80	7919112	0,018	0	0,009	1	0,009	1	0,018	0
90	10331422	0,345	0	0,350	0	0,319	0	0,373	0
100	13103420	0,276	0	0,271	0	0,258	0	0,278	0
110	16177106	0,239	0	0,237	0	0,237	0	0,238	0
120	19631156	0,102	0	0,073	0	0,070	0	0,084	0
130	23460170	0,495	0	0,325	0	0,299	0	0,413	0
140	27873238	0,352	0	0,350	0	0,346	0	0,323	0
	Vidurkis:	0,193	1	0,173	1,1	0,163	0,1	0,181	1

16 lentelė. Eksperimentų su komponento „KRYŽMINIMO PROCEDŪRA“ modifikacijomis rezultatai ($n = 1024$, **`KP-7`**, **`KP-8`**, **`KP-9`**)

		`HGA-BV`		**`KP-7`**		**`KP-8`**		**`KP-9`**	
m	GŽR	Vidutinis nuokrypis (%)	*SB*	Vidutinis nuokrypis (%)	*SB*	Vidutinis nuokrypis (%)	*SB*	Vidutinis nuokrypis (%)	*SB*
50	2730510	0,000	10	0,000	10	0,000	10	0,000	10
60	4136400	0,031	0	0,030	0	0,030	0	0,019	0
70	5868614	0,067	0	0,067	0	0,075	0	0,045	0
80	7919112	0,018	0	0,013	1	0,016	1	0,013	0
90	10331422	0,345	0	0,372	0	0,343	0	0,347	0
100	13103420	0,276	0	0,262	0	0,278	0	0,277	0
110	16177106	0,239	0	0,227	0	0,250	0	0,238	0
120	19631156	0,102	0	0,077	0	0,080	0	0,080	0
130	23460170	0,495	0	0,331	0	0,490	0	0,494	0
140	27873238	0,352	0	0,294	0	0,353	0	0,339	0
	Vidurkis:	0,193	1	0,167	1,1	0,192	1,1	0,185	1



a) b) c)

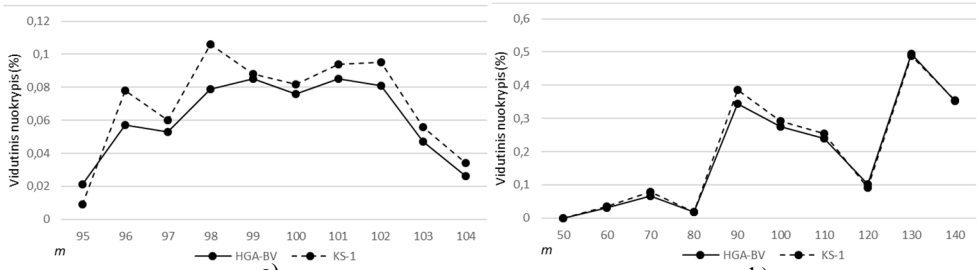
27 pav. Vidutinis nuokrypis (%) ($n = 1024$): a) **`HGA-BV`**, **`KP-1`**, **`KP-2`**, **`KP-3`**; b) **`HGA-BV`**, **`KP-4`**, **`KP-5`**, **`KP-6`**; c) **`HGA-BV`**, **`KP-7`**, **`KP-8`**, **`KP-9`**

17 lentelė. Eksperimentų su komponento „KRYŽMINIMŲ SKAIČIUS“ modifikacijomis rezultatai ($n = 256$)

		HGA-BV		KS-1	
m	GŽR	Vidutinis nuokrypis (%)	Sėkmingų bandymų skaičius	Vidutinis nuokrypis (%)	Sėkmingų bandymų skaičius
95	48081112	0,021	4	0,009	2
96	49182368	0,057	3	0,078	0
97	50344050	0,053	1	0,060	1
98	51486642	0,079	1	0,106	0
99	52660116	0,085	0	0,088	0
100	53838088	0,076	0	0,082	0
101	55014262	0,085	0	0,094	0
102	56202826	0,081	0	0,095	0
103	57417112	0,047	0	0,056	0
104	58625240	0,026	0	0,034	0
	Vidurkis:	0,061	0,9	0,070	0,3

18 lentelė. Eksperimentų su komponento „KRYŽMINIMŲ SKAIČIUS“ modifikacijomis rezultatai ($n = 1024$)

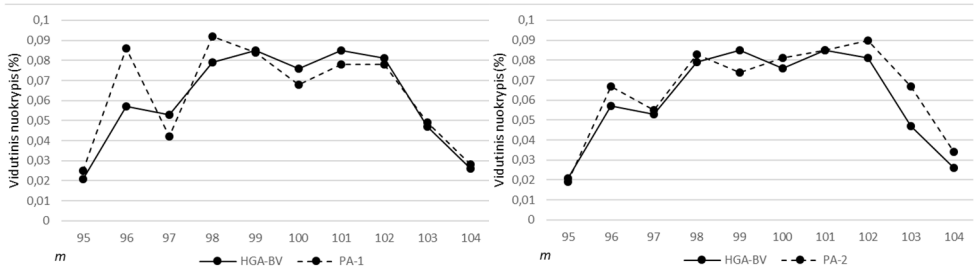
		HGA-BV		KS-1	
m	GŽR	Vidutinis nuokrypis (%)	Sėkmingų bandymų skaičius	Vidutinis nuokrypis (%)	Sėkmingų bandymų skaičius
50	2730510	0,000	10	0,000	10
60	4136400	0,031	0	0,034	0
70	5868614	0,067	0	0,079	0
80	7919112	0,018	0	0,019	0
90	10331422	0,345	0	0,385	0
100	13103420	0,276	0	0,291	0
110	16177106	0,239	0	0,254	0
120	19631156	0,102	0	0,092	0
130	23460170	0,495	0	0,489	0
140	27873238	0,352	0	0,354	0
	Vidurkis:	0,193	1	0,200	1



28 pav. Vidutinis nuokrypis (%): a) $n = 256$, **HGA-BV**, **KS-1**; b) $n = 1024$, **HGA-BV**, **KS-1**

19 lentelė. Eksperimentų su komponento „POPULIACIJOS ATNAUJINIMAS“ modifikacijomis rezultatai ($n = 256$, **HGA-BV**, **PA-1**, **PA-2**)

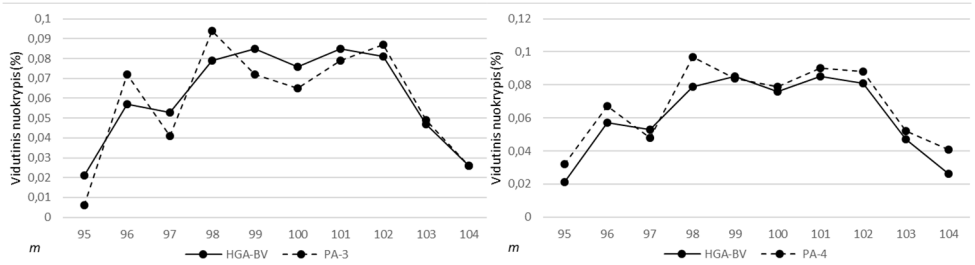
		HGA-BV		**PA-1**		**PA-2**	
m	GŽR	Vidutinis nuokrypis (%)	Sėkmingų bandymų skaičius	Vidutinis nuokrypis (%)	Sėkmingų bandymų skaičius	Vidutinis nuokrypis (%)	Sėkmingų bandymų skaičius
95	48081112	0,021	4	0,025	4	0,019	4
96	49182368	0,057	3	0,086	0	0,067	1
97	50344050	0,053	1	0,042	2	0,055	1
98	51486642	0,079	1	0,092	0	0,083	1
99	52660116	0,085	0	0,084	0	0,074	0
100	53838088	0,076	0	0,068	0	0,081	0
101	55014262	0,085	0	0,078	0	0,085	0
102	56202826	0,081	0	0,078	0	0,090	0
103	57417112	0,047	0	0,049	0	0,067	0
104	58625240	0,026	0	0,028	0	0,034	0
	Vidurkis:	0,061	0,9	0,063	0,6	0,066	0,7



29 pav. Vidutinis nuokrypis (%) ($n = 256$): a) **HGA-BV** ir **PA-1**; b) **HGA-BV** ir **PA-2**

20 lentelė. Eksperimentų su komponento „POPULIACIJOS ATNAUJINIMAS“ modifikacijomis rezultatai ($n = 256$, **`HGA-BV`**, **`PA-3`**, **`PA-4`**)

		`HGA-BV`		**`PA-3`**		**`PA-4`**	
m	GŽR	Vidutinis nuokrypis (%)	Sėkmingų bandymų skaičius	Vidutinis nuokrypis (%)	Sėkmingų bandymų skaičius	Vidutinis nuokrypis (%)	Sėkmingų bandymų skaičius
95	48081112	0,021	4	0,006	6	0,032	3
96	49182368	0,057	3	0,072	1	0,067	2
97	50344050	0,053	1	0,041	2	0,048	1
98	51486642	0,079	1	0,094	0	0,097	0
99	52660116	0,085	0	0,072	0	0,084	0
100	53838088	0,076	0	0,065	0	0,079	0
101	55014262	0,085	0	0,079	0	0,090	0
102	56202826	0,081	0	0,087	0	0,088	0
103	57417112	0,047	0	0,049	0	0,052	0
104	58625240	0,026	0	0,026	0	0,041	0
	Vidurkis:	0,061	0,9	0,059	0,9	0,068	0,6



a) b)

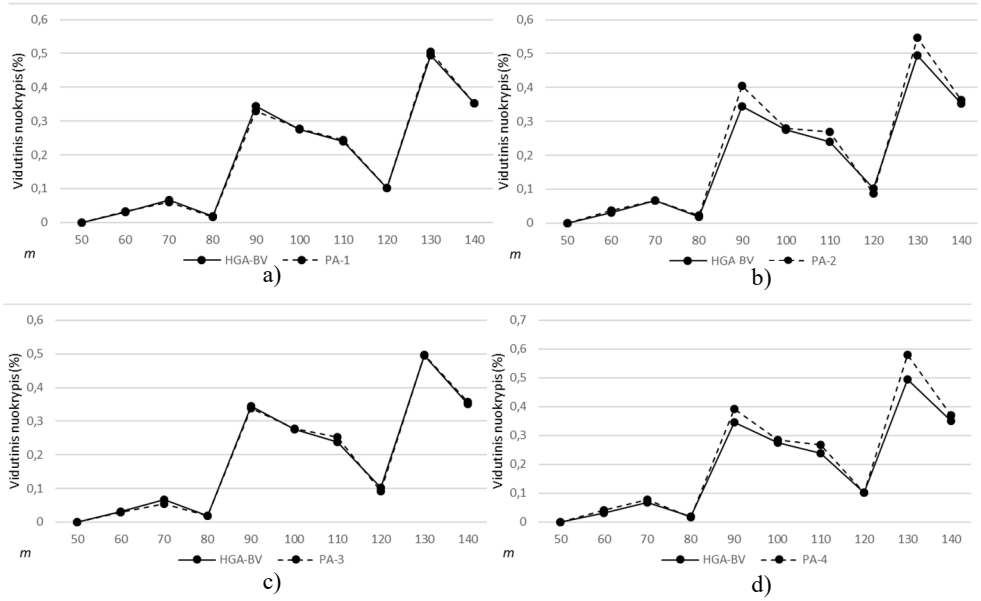
30 pav. Vidutinis nuokrypis (%) ($n = 256$): a) **`HGA-BV`** ir **`PA-3`**; b) **`HGA-BV`** ir **`PA-4`**

21 lentelė. Eksperimentų su komponento „POPULIACIJOS ATNAUJINIMAS“ modifikacijomis rezultatai ($n = 1024$, **`HGA-BV`**, **`PA-1`**, **`PA-2`**)

		`HGA-BV`		**`PA-1`**		**`PA-2`**	
m	GŽR	Vidutinis nuokrypis (%)	Sėkmingų bandymų skaičius	Vidutinis nuokrypis (%)	Sėkmingų bandymų skaičius	Vidutinis nuokrypis (%)	Sėkmingų bandymų skaičius
50	2730510	0,000	10	0,000	10	0,000	10
60	4136400	0,031	0	0,032	0	0,037	0
70	5868614	0,067	0	0,060	0	0,067	0
80	7919112	0,018	0	0,016	0	0,022	0
90	10331422	0,345	0	0,329	0	0,404	0
100	13103420	0,276	0	0,278	0	0,279	0
110	16177106	0,239	0	0,244	0	0,268	0
120	19631156	0,102	0	0,102	0	0,087	0
130	23460170	0,495	0	0,506	0	0,546	0
140	27873238	0,352	0	0,352	0	0,364	0
	Vidurkis:	0,193	1	0,192	1	0,207	1

22 lentelė. Eksperimentų su komponento „POPULIACIJOS ATNAUJINIMAS“ modifikacijomis rezultatai ($n = 1024$, **`HGA-BV`**, **`PA-3`**, **`PA-4`**)

		`HGA-BV`		**`PA-3`**		**`PA-4`**	
m	GŽR	Vidutinis nuokrypis (%)	Sėkmingų bandymų skaičius	Vidutinis nuokrypis (%)	Sėkmingų bandymų skaičius	Vidutinis nuokrypis (%)	Sėkmingų bandymų skaičius
50	2730510	0,000	10	0,000	10	0,000	10
60	4136400	0,031	0	0,029	0	0,040	0
70	5868614	0,067	0	0,053	0	0,078	0
80	7919112	0,018	0	0,018	1	0,016	0
90	10331422	0,345	0	0,339	0	0,392	0
100	13103420	0,276	0	0,278	0	0,286	0
110	16177106	0,239	0	0,252	0	0,269	0
120	19631156	0,102	0	0,091	0	0,103	0
130	23460170	0,495	0	0,498	0	0,581	0
140	27873238	0,352	0	0,358	0	0,370	0
	Vidurkis:	0,193	1	0,192	1,1	0,214	1



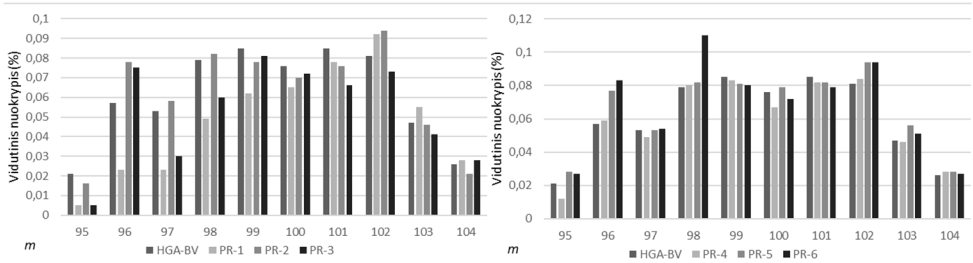
31 pav. Vidutinis nuokrypis (%) ($n = 1024$): a) **HGA-BV** ir **PA-1**; b) **HGA-BV** ir **PA-2**; c) **HGA-BV** ir **PA-3**; d) **HGA-BV** ir **PA-4**

23 lentelė. Eksperimentų su komponento „POPULIACIJOS PERTVARKYMAS“ modifikacijomis rezultatai ($n = 256$, **HGA-BV**, **PR-1**, **PR-2**, **PR-3**)

		HGA-BV		**PR-1**		**PR-2**		**PR-3**	
m	GŽR	Vidutinis nuokrypis (%)	*SB*	Vidutinis nuokrypis (%)	*SB*	Vidutinis nuokrypis (%)	*SB*	Vidutinis nuokrypis (%)	*SB*
95	48081112	0,021	4	0,005	8	0,016	4	0,005	7
96	49182368	0,057	3	0,023	7	0,078	0	0,075	0
97	50344050	0,053	1	0,023	5	0,058	1	0,030	4
98	51486642	0,079	1	0,049	4	0,082	0	0,060	6
99	52660116	0,085	0	0,062	1	0,078	0	0,081	0
100	53838088	0,076	0	0,065	1	0,070	0	0,072	0
101	55014262	0,085	0	0,078	0	0,076	0	0,066	0
102	56202826	0,081	0	0,092	0	0,094	0	0,073	0
103	57417112	0,047	0	0,055	0	0,046	0	0,041	0
104	58625240	0,026	0	0,028	0	0,021	0	0,028	0
	Vidurkis:	0,061	0,9	0,048	2,6	0,062	0,5	0,053	1,7

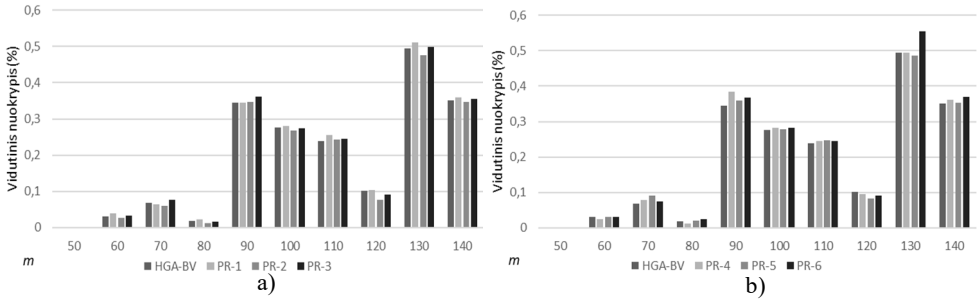
24 lentelė. Eksperimentų su komponento „POPULIACIJOS PERTVARKYMAS“ modifikacijomis rezultatai ($n = 256$, **HGA-BV**, **PR-4**, **PR-5**, **PR-6**)

		HGA-BV		**PR-4**		**PR-5**		**PR-6**	
m	GŽR	Vidutinis nuokrypis (%)	*SB*	Vidutinis nuokrypis (%)	*SB*	Vidutinis nuokrypis (%)	*SB*	Vidutinis nuokrypis (%)	*SB*
95	48081112	0,021	4	0,012	6	0,028	3	0,027	4
96	49182368	0,057	3	0,059	3	0,077	0	0,083	0
97	50344050	0,053	1	0,049	1	0,053	1	0,054	1
98	51486642	0,079	1	0,080	0	0,082	1	0,110	0
99	52660116	0,085	0	0,083	0	0,081	0	0,080	0
100	53838088	0,076	0	0,067	0	0,079	0	0,072	0
101	55014262	0,085	0	0,082	0	0,082	0	0,079	0
102	56202826	0,081	0	0,084	0	0,094	0	0,094	0
103	57417112	0,047	0	0,046	0	0,056	0	0,051	0
104	58625240	0,026	0	0,028	0	0,028	0	0,027	0
	Vidurkis:	0,061	0,9	0,059	1	0,066	0,5	0,068	0,5



a) b)

32 pav. Vidutinis nuokrypis (%) ($n = 256$): a) **HGA-BV**, **PR-1**, **PR-2**, **PR-3**; b) **HGA-BV**, **PR-4**, **PR-5**, **PR-6**



a) b)

33 pav. Vidutinis nuokrypis (%) ($n = 1024$): a) **HGA-BV**, **PR-1**, **PR-2**, **PR-3**; b) **HGA-BV**, **PR-4**, **PR-5**, **PR-6**

25 lentelė. Eksperimentų su komponento „POPULIACIJOS PERTVARKYMAS“ modifikacijomis rezultatai ($n = 1024$, **HGA-BV**, **PR-1**, **PR-2**, **PR-3**)

		HGA-BV		**PR-1**		**PR-2**		**PR-3**	
m	GŽR	Vidutinis nuokrypis (%)	*SB*	Vidutinis nuokrypis (%)	*SB*	Vidutinis nuokrypis (%)	*SB*	Vidutinis nuokrypis (%)	*SB*
50	2730510	0,000	10	0,000	10	0,000	10	0,000	10
60	4136400	0,031	0	0,038	0	0,026	0	0,033	0
70	5868614	0,067	0	0,063	0	0,060	0	0,076	0
80	7919112	0,018	0	0,023	0	0,012	0	0,015	0
90	10331422	0,345	0	0,345	0	0,347	0	0,362	0
100	13103420	0,276	0	0,281	0	0,268	0	0,274	0
110	16177106	0,239	0	0,255	0	0,243	0	0,244	0
120	19631156	0,102	0	0,103	0	0,076	0	0,091	0
130	23460170	0,495	0	0,511	0	0,476	0	0,498	0
140	27873238	0,352	0	0,360	0	0,347	0	0,356	0
	Vidurkis:	0,193	1	0,198	1	0,186	1	0,195	1

26 lentelė. Eksperimentų su komponento „POPULIACIJOS PERTVARKYMAS“ modifikacijomis rezultatai ($n = 1024$, **HGA-BV**, **PR-4**, **PR-5**, **PR-6**)

		HGA-BV		**PR-4**		**PR-5**		**PR-6**	
m	GŽR	Vidutinis nuokrypis (%)	*SB*	Vidutinis nuokrypis (%)	*SB*	Vidutinis nuokrypis (%)	*SB*	Vidutinis nuokrypis (%)	*SB*
50	2730510	0,000	10	0,000	10	0,000	10	0,000	10
60	4136400	0,031	0	0,025	0	0,031	0	0,031	0
70	5868614	0,067	0	0,078	0	0,090	0	0,074	0
80	7919112	0,018	0	0,011	0	0,020	0	0,024	0
90	10331422	0,345	0	0,385	0	0,359	0	0,367	0
100	13103420	0,276	0	0,283	0	0,279	0	0,282	0
110	16177106	0,239	0	0,245	0	0,247	0	0,244	0
120	19631156	0,102	0	0,094	0	0,082	0	0,091	0
130	23460170	0,495	0	0,494	0	0,486	0	0,556	0
140	27873238	0,352	0	0,361	0	0,354	0	0,370	0
	Vidurkis:	0,193	1	0,198	1	0,195	1	0,204	1

27–28 lent. pateikiami apibendrinti rezultatai, rodantys populiacijos pertvarkymų vidutinį skaičių per 10 algoritmo vykdymų. Pateikiami bazinio algoritmo varianto (**HGA-BV**) ir kiekvieno komponento vienos modifikacijos (pasirinktos atvaizduoti pirmos modifikacijos, išskyrus komponentą „POPULIACIJOS PERTVARKYMAS“, nes šio komponento pirmoje modifikacijoje

netaikomas joks populiacijos pertvarkymas) rezultatai (**PP-1**, **LP-1**, **STA-1**, **KP-1**, **KS-1**, **PA-1**, **PR-2**).

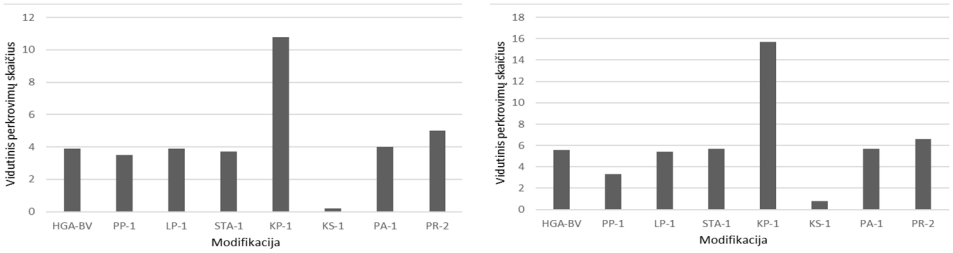
27 lentelė. Vidutinis populiacijos pertvarkymų skaičius[11] ($n = 256$)

	HGA-BV	PP-1	LP-1	STA-1	KP-1	KS-1	PA-1	PR-2
95	3,00	3,10	2,60	1,90	6,80	0,10	3,00	3,10
96	3,00	3,30	3,80	4,40	12,10	0,00	4,70	4,30
97	3,90	3,20	3,10	2,50	11,70	0,00	4,10	4,30
98	4,20	3,10	3,90	4,20	14,20	0,10	4,20	5,10
99	4,10	3,40	4,00	4,80	12,50	0,40	4,00	5,10
100	4,70	3,40	4,50	4,30	13,00	0,40	4,70	5,50
101	5,00	3,50	4,60	4,30	11,60	0,40	3,90	5,30
102	4,10	3,80	4,30	3,90	9,70	0,10	3,50	4,80
103	3,50	4,10	4,20	3,20	8,00	0,00	3,90	5,20
104	3,60	4,10	4,00	3,20	8,50	0,10	3,70	4,40
Vidurkis:	3,90	3,50	3,90	3,70	10,80	0,20	4,00	5,00

28 lentelė. Vidutinis populiacijos pertvarkymų skaičius ($n = 1024$)

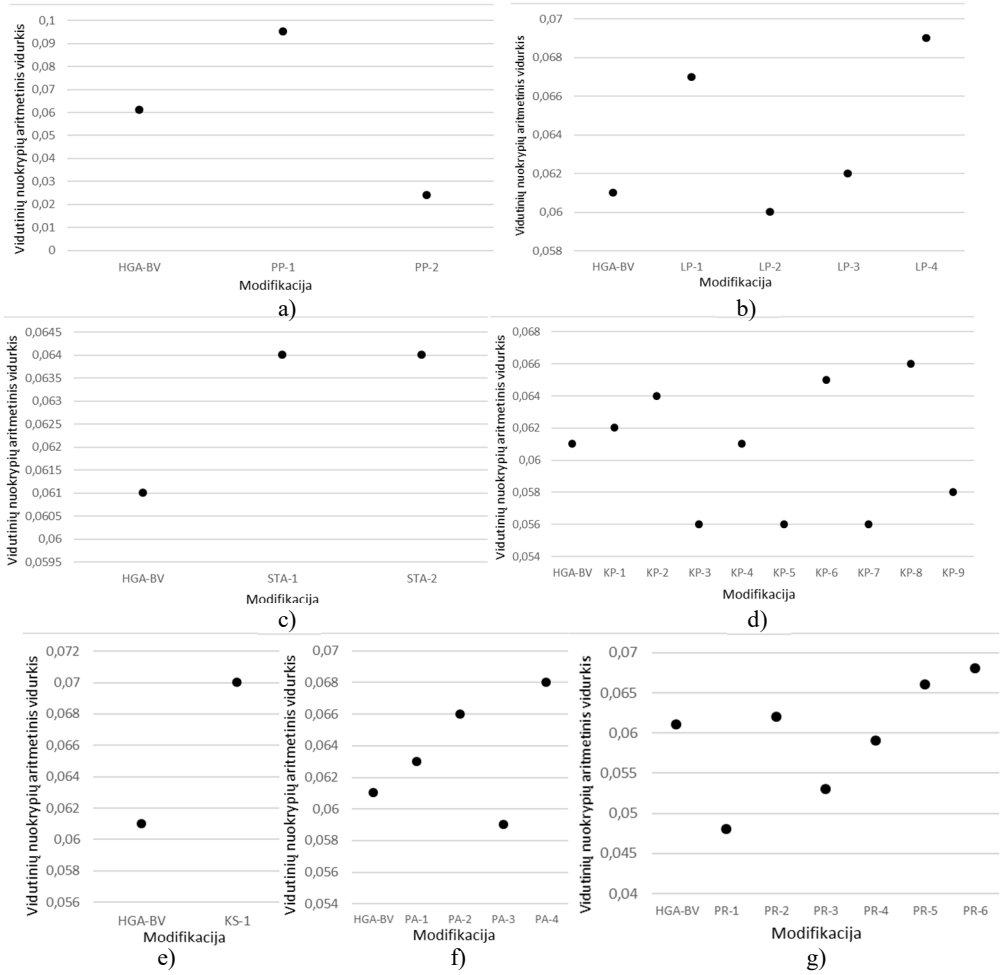
	HGA-BV	PP-1	LP-1	STA-1	KP-1	KS-1	PA-1	PR-2
50	0,00	1,70	0,10	0,00	0,00	0,00	0,00	0,00
60	6,40	3,80	6,00	6,00	16,60	0,80	6,20	7,60
70	7,00	3,60	6,10	7,20	19,60	1,10	7,30	8,50
80	5,30	3,40	5,30	5,20	12,40	0,70	4,50	9,10
90	5,40	3,70	6,30	5,70	17,00	0,90	6,20	6,30
100	5,70	3,30	4,90	5,30	16,30	0,50	6,10	5,20
110	6,00	3,40	6,10	6,50	17,30	0,80	5,80	6,10
120	7,30	3,30	6,60	7,30	19,80	1,10	7,30	8,80
130	7,00	3,30	7,00	8,00	19,40	0,90	8,00	7,70
140	5,70	3,20	5,40	6,10	18,80	0,90	5,70	6,70
Vidurkis:	5,60	3,30	5,40	5,70	15,70	0,80	5,70	6,60

[11] Vidutinis populiacijos pertvarkymų skaičius vieno algoritmo paleidimo metu.



a) b)

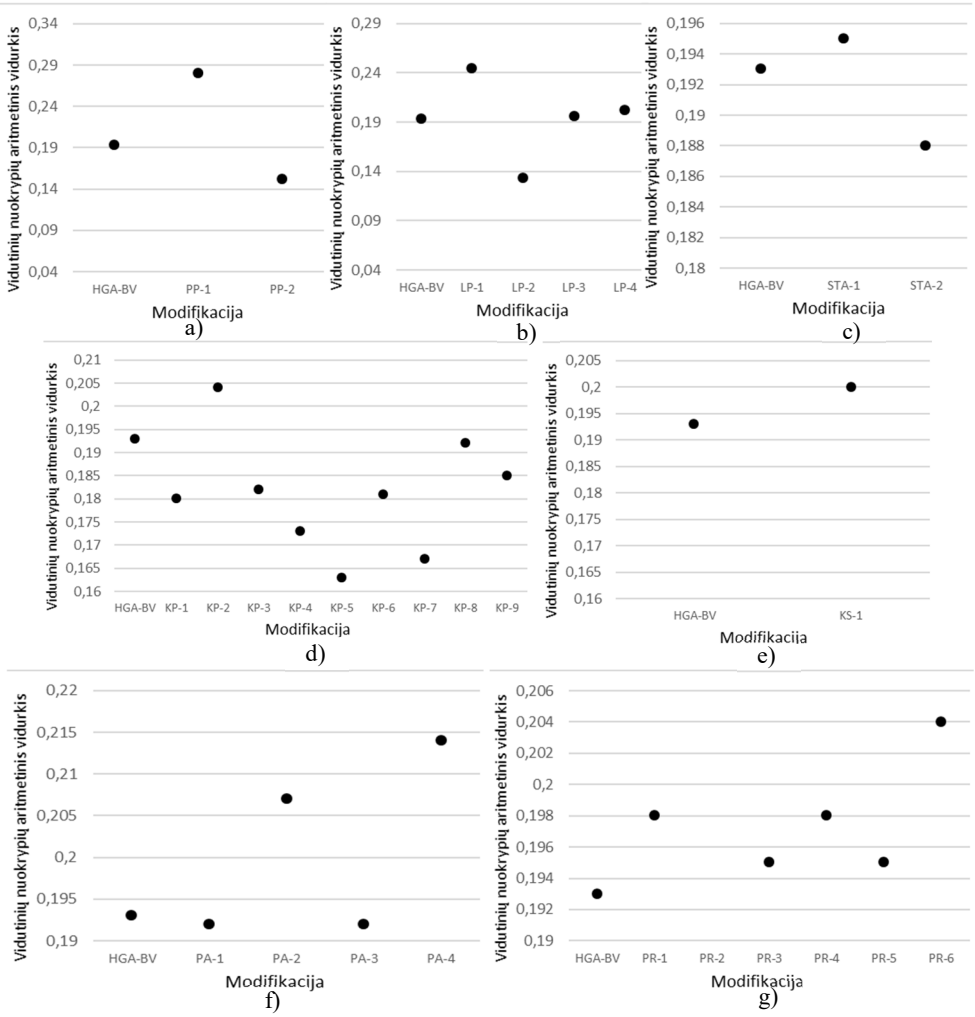
34 pav. Vidutinis pertvarkymų skaičius ($n = 1024$) vykdant algoritmo modifikacijas **`HGA-BV`**, **`PV-1`**, **`LP-1`**, **`STA-1`**, **`KP-1`**, **`KS-1`**, **`PA-1`**, **`PR-2`**: a) $n = 256$; b) $n = 1024$



a) b) c) d) e) f) g)

35 pav. Visų komponentų modifikacijų vidutinių nuokrypių vidurkiai (kai $m = 95, 96, \ldots, 104$, $n = 254$): a) „PRADINĖ POPULIACIJA“; b) „LOKALUSIS PAGERINIMAS“; c) „SPRENDINIŲ-TĖVŲ ATRANKA“ d) „KRYŽMINIMO PROCEDŪRA“; e) „KRYŽMINIMŲ SKAIČIUS“; f) „POPULIACIJOS ATNAUJINIMAS“; g) „POPULIACIJOS PERTVARKYMAS“

35, 36 pav. pateikiami visų komponentų modifikacijų (su visomis tirtomis m reikšmėmis) vidutinių nuokrypių nuo geriausios žinomos tikslo funkcijos reikšmės vidurkiai, kai uždavinio dydis yra $n = 256$ ir $n = 1024$.



36 pav. Visų komponentų modifikacijų vidutinių nuokrypių vidurkiai (kai $m = 50, 60, \ldots, 140$, $n = 1024$): a) „PRADINĖ POPULIACIJA“; b) „LOKALUSIS PAGERINIMAS“; c) „SPRENDINIŲ-TĖVŲ ATRANKA“ d) „KRYŽMINIMO PROCEDŪRA“; e) „KRYŽMINIMŲ SKAIČIUS“; f) „POPULIACIJOS ATNAUJINIMAS“; g) „POPULIACIJOS PERTVARKYMAS“

Buvo atliktas komponentų rinkinių eksperimentinis tyrimas. 29, 30 lent., 37, 38 pav. pateikiami rezultatai su komponentų rinkiniais, kurių eksperimentinio tyrimo rezultatai geriausi (daugiau rezultatų pateikiama publikacijoje [86]). Rinkiniai sudaryti iš šių komponentų: 1) **`STA-2`**, **`KP-6`**, **`KS-1`**, **`LP-2`**, **`PR-2`**; 2) **`STA-2`**, **`KP-5`**, **`KS-1`**, **`LP-2`**, **`PR-2`**; 3) **`STA-2`**, **`KP-5`**, **`KS-3`** (šiuo atveju panaudota papildoma komponento „KRYŽMINIMŲ SKAIČIUS“ modifikacija – vienos

generacijos metu panaudotas padidintas kryžminimų skaičius, lygus PS; čia PS – populiacijos dydis (lygus 20)), **`LP-2`**, **`PR-2`**. Visais atvejais naudota padidinta ir pagerinta pradinė populiacija (**`PP-2`**).

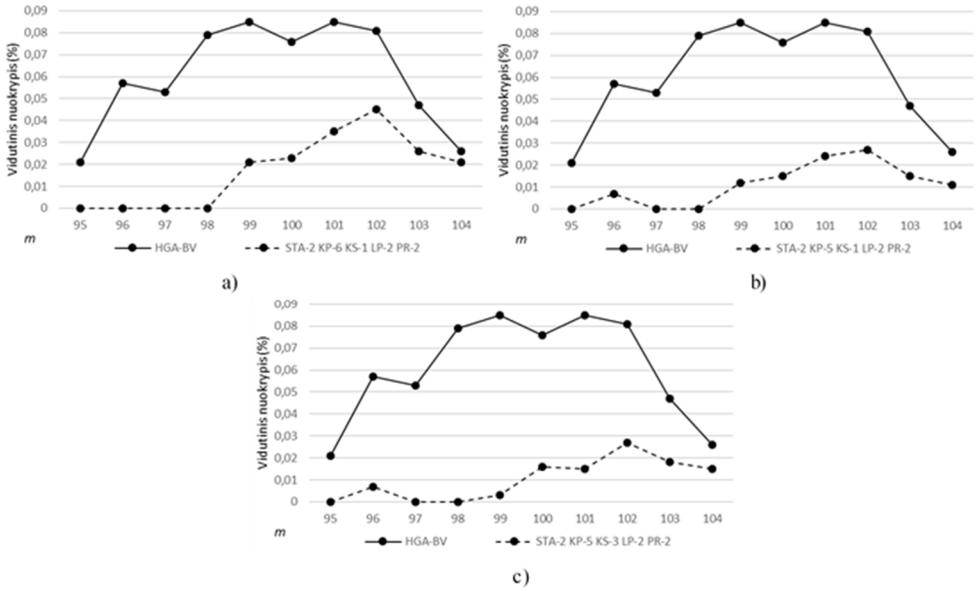
29 lentelė. Eksperimentų su komponentų modifikacijų rinkiniais rezultatai ($n = 256$)

		`HGA-BV`		**`STA-2, KP-6, KS-1, LP-2, PR-2`**		**`STA-2, KP-5, KS-1, LP-2, PR-2`**		**`STA-2, KP-5, KS-3,`**$^{\blacklozenge}$ **`LP-2, PR-2`**$^{\blacklozenge\blacklozenge}$	
m	GŽR	Vidutinis nuokrypis (%)	*SB*	Vidutinis nuokrypis (%)	*SB*	Vidutinis nuokrypis (%)	*SB*	Vidutinis nuokrypis (%)	*SB*
95	48081112	0,021	4	0,000	10	0,000	10	0,000	10
96	49182368	0,057	3	0,000	10	0,007	9	0,007	9
97	50344050	0,053	1	0,000	10	0,000	10	0,000	10
98	51486642	0,079	1	0,000	10	0,000	10	0,000	10
99	52660116	0,085	0	0,021	5	0,012	8	0,003	9
100	53838088	0,076	0	0,023	3	0,015	5	0,016	4
101	55014262	0,085	0	0,035	1	0,024	2	0,015	5
102	56202826	0,081	0	0,045	0	0,027	4	0,027	4
103	57417112	0,047	0	0,026	0	0,015	4	0,018	3
104	58625240	0,026	0	0,021	0	0,011	3	0,015	2
	Vidurkis:	0,061	0,9	0,017	4,9	0,011	6,5	0,010	6,6

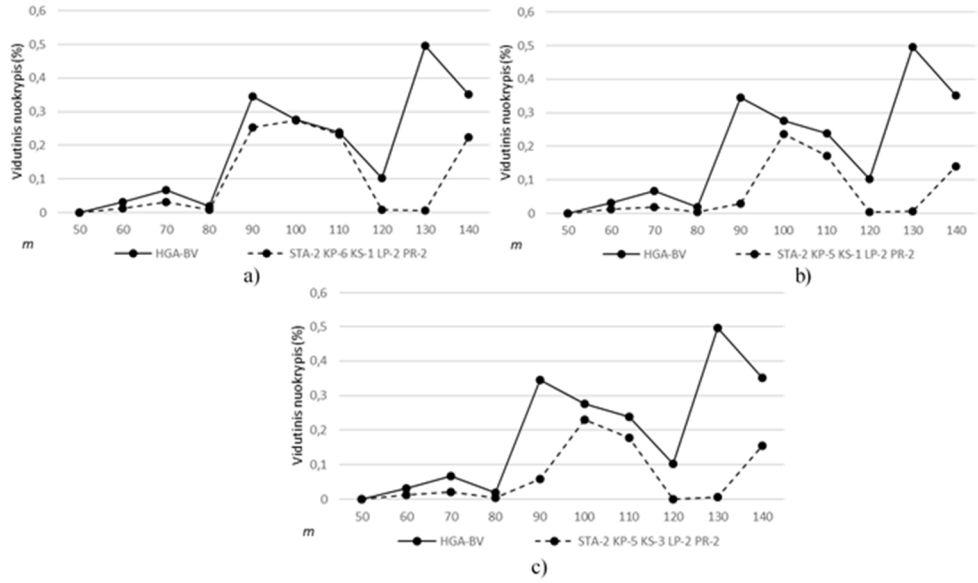
$^{\blacklozenge}$ – kryžminimų skaičius lygus PS;
$^{\blacklozenge\blacklozenge}$ – naudotas padidintas TS iteracijų skaičius, $\tau = 200$.

30 lentelė. Eksperimentų su komponentų modifikacijų rinkiniais rezultatai ($n = 1024$)

		`HGA-BV`		**`STA-2, KP-6, KS-1, LP-2, PR-2`**		**`STA-2, KP-5, KS-1, LP-2, PR-2`**		**`STA-2, KP-5, KS-3, LP-2, PR-2`**	
m	GŽR	Vidutinis nuokrypis (%)	*SB*	Vidutinis nuokrypis (%)	*SB*	Vidutinis nuokrypis (%)	*SB*	Vidutinis nuokrypis (%)	*SB*
50	2730510	0,000	10	0,000	10	0,000	10	0,000	10
60	4136400	0,031	0	0,012	0	0,012	0	0,012	0
70	5868614	0,067	0	0,032	1	0,018	2	0,021	0
80	7919112	0,018	0	0,008	0	0,005	2	0,004	0
90	10331422	0,345	0	0,253	0	0,029	0	0,058	0
100	13103420	0,276	0	0,275	0	0,237	0	0,231	0
110	16177106	0,239	0	0,233	0	0,171	0	0,177	0
120	19631156	0,102	0	0,009	2	0,005	2	0,001	5
130	23460170	0,495	0	0,007	1	0,007	0	0,007	0
140	27873238	0,352	0	0,223	0	0,140	0	0,155	0
	Vidurkis:	0,193	1,0	0,105	1,4	0,062	1,6	0,067	1,5



37 pav. Vidutinis nuokrypis (%) ($n = 256$): a) **HGA-BV**, rinkinys: **STA-2**, **KP-6**, **KS-1**, **LP-2**, **PR-2**; b) **HGA-BV**, rinkinys: **STA-2**, **KP-5**, **KS-1**, **LP-2**, **PR-2**; c) **HGA-BV**, rinkinys: **STA-2**, **KP-5**, **KS-3**, **LP-2**, **PR-2**



38 pav. Vidutinis nuokrypis (%) ($n = 1024$): a) **HGA-BV**, rinkinys: **STA-2**, **KP-6**, **KS-1**, **LP-2**, **PR-2**; b) **HGA-BV**, rinkinys: **STA-2**, **KP-5**, **KS-1**, **LP-2**, **PR-2**; c) **HGA-BV**, rinkinys: **STA-2**, **KP-5**, **KS-3**, **LP-2**, **PR-2**

3.4. Eksperimentinių tyrimų apibendrinamosios pastabos

Eksperimentiniai tyrimai (kai $n = 256$) parodė, kad HGA yra kur kas geresnis nei lygintas PGEA, išskyrus kelis atvejus, kai $m = 26, 101, 102, 103$. HGA vykdymo laikas ypač trumpas, kai m kinta nuo 30 iki 100 (15 pav.). Toks trumpas algoritmo vykdymo laikas rodo, kad HGA sugeba pasiekti galimai optimalias reikšmes jau ankstyvoje algoritmo stadijoje, sukonstravus pradinę populiaciją ir pritaikius tik HITP. Labai tikėtina, kad HGA efektyvumą būtų galima dar padidinti nustačius tinkamesnes pradines parametrų reikšmes.

Reikia pažymėti, kad didelio dydžio uždavinys ($n = 1024$) yra gerokai sudėtingesnis ir jo sprendimas užėmė daugiau laiko. Vis dėlto pavyko pasiekti geresnes tikslo funkcijos reikšmes su daugiau nei 190 m reikšmių (4 lent.). Šiuo atveju CP veikimo laikas nebuvo fiksuojamas. Visos likusios reikšmės (kurių nepavyko pagerinti) yra iš Misevičiaus, Guogio ir Stanevičienės straipsnio [19].

Atlikti papildomi eksperimentai su komponentų modifikacijų rinkiniais, sprendžiant didelio dydžio uždavinį ($n = 1024$), leido pagerinti geriausias iki šiol žinomas tikslo funkcijos reikšmes dar 91 kartą. Šie rezultatai pateikiami straipsnyje [87].

Tiriant komponentą „PRADINĖ POPULIACIJA“ nustatyta, kad akivaizdžiai tikslinga naudoti ne tik pagerintą, bet ir padidintą pradinę populiaciją (**`PP-2`**).

Tiriant komponentą „LOKALUSIS PAGERINIMAS“ nustatyta, kad pranašesnis yra variantas, kai didinamas tabu paieškos iteracijų skaičius (**`LP-2`**).

Komponento „SPRENDINIŲ-TĖVŲ ATRANKA“ tyrimas parodė, kad santykinai geresnė yra sprendinių rangu pagrįsta atrankos procedūra (**`STA-2`**).

Atlikus eksperimentus su komponentu „KRYŽMINIMO PROCEDŪRA“ PŠF uždaviniui, paaiškėjo, kad geri rezultatai gaunami ir nenaudojant priešingųjų (opozicinių) palikuonių. Pakanka tradicinių kryžminimo variantų, kai sugeneruojamas vienas palikuonis. Santykinai geresni rezultatai gauti vykdant dažnumu paremtą godųjį kryžminimą (**`KP-1`**), godųjį kryžminimą (**`KP-3`**) ir euristinį kryžminimą (**`KP-5`**).

Tiriant komponentą „KRYŽMINIMŲ SKAIČIUS“ nustatyta, kad santykinai geresnis yra variantas, kai sukuriamas nepadidintas skaičius palikuonių (**`KS-2`**), t. y. atliekamas vienas kryžminimas.

Eksperimentų su komponentu „POPULIACIJOS ATNAUJINIMAS“ rezultatai parodė, kad bazinis algoritmo variantas (**`HGA-BV`**) yra pakankamai geras. Taip pat geras variantas, kai palikuonis pakeičia blogesnįjį tėvą (jeigu palikuonis geresnis už blogesnįjį tėvą) (**`PA-3`**).

Atlikus eksperimentus su komponentu „POPULIACIJOS PERTVARKYMAS“, matyti, kad geri rezultatai gaunami ir netaikant populiacijos pertvarkymo (**`PR-1`**). Netikslinga populiacijos pertvarkymo metu taikyti didelio masto populiacijos „sugriovimą“ arba populiacijos generavimą iš naujo. Geriau taikyti multimutacija paremtą populiacijos pertvarkymą (**`PR-2`**), kai mutacijos lygis yra gana žemas.

Mažiausias vidutinis nuokrypis $0{,}024$ ($n = 256$) rastas, kai eksperimentai buvo atlikti su komponento „PRADINĖ POPULIACIJA“ modifikacija **`PP-2`**. Kai

uždavinio dydis $n = 1024$, mažiausias vidutinis nuokrypis $0{,}133$ buvo su kmponento „LOKALUSIS PAGERINIMAS“ modifikacija **`LP-2`**.

Vidutinis nuokrypis visoms testuotoms m reikšmėms buvo $0{,}061$, kai uždavinio dydis $n = 256$, ir $0{,}193$, kai uždavinio dydis $n = 1024$.

Geriausia žinoma reikšmė buvo surasta visų algoritmo 10 vykdymų metu, kai $m = 95, 97$, $n = 256$, ir $m = 50$, $n = 1024$.

Atlikus eksperimentinius tyrimus su komponentų modifikacijų rinkiniais, matyti, kad, nagrinėjant originalias (naujas) komponentų kombinacijas (konfigūracijas), galima siekti dar geresnių rezultatų. Geriausius rezultatus pavyko gauti su komponentų modifikacijų rinkiniais: **`STA-2`**, **`KP-5`**, **`KS-3`**, **`LP-2`**, **`PR-2`** (kai $n = 256$) ir **`STA-2`**, **`KP-5`**, **`KS-1`**, **`LP-2`**, **`PR-2`** (kai $n = 1024$).

Eksperimentų su komponentų modifikacijų rinkiniais rezultatų vidutinis nuokrypis nuo GŽR visoms testuotoms m reikšmėms buvo $0{,}013$, kai uždavinio dydis $n = 256$, ir $0{,}078$, kai uždavinio dydis $n = 1024$.

Geriausia žinoma reikšmė, kai eksperimentai buvo atlikti su komponentų modifikacijų rinkiniais, buvo surasta visų algoritmo 10 vykdymų metu, kai $m = 95, 96, 97, 98$, $n = 256$, ir, kai $m = 50$, $n = 1024$.

3.5. Trečiojo skyriaus išvados

1. Pasiūlytas HGA buvo sėkmingai pritaikytas PŠF uždaviniui. Atliktų eksperimentų rezultatai leidžia daryti išvadą, kad HGA yra efektyvus tiek sprendinių kokybės, tiek vykdymo laiko požiūriu.
2. HGA bazinio varianto eksperimentai parodė, kad tikslinga atlikti išsamesnius algoritmo atskirų komponentų modifikacijų tyrimus siekiant rasti geriausią algoritmo variantą.
   Atlikti eksperimentai su skirtingomis algoritmo komponentų modifikacijomis rodo, kad:
   - tikslinga naudoti pagerintą ir padidintą pradinę populiaciją;
   - geresni rezultatai gaunami, kai didinamas tabu paieškos iteracijų skaičius;
   - santykinai geresnė yra sprendinių rangu pagrįsta sprendinių-tėvų atrankos procedūra;
   - nustatyta, jog geresnius rezultatus leidžia gauti dažnumu paremtas godusis kryžminimas, godusis kryžminimas ir euristinis kryžminimas;
   - efektyviau, kai taikomas operatyvus populiacijos atnaujinimas (t. y. vienoje generacijoje – vienas kryžminimas);
   - patartina rinktis variantą, kai palikuonis pakeičia blogesnįjį tėvą; be to, turi būti patenkintas minimalaus atstumo kriterijus palikuonių įvairovei užtikrinti;
   - perspektyvu taikyti multimutacija paremtą populiacijos pertvarkymą (populiacijos „invaziją“), kai populiacijos individų mutacijos lygis yra neaukštas.

BENDROSIOS IŠVADOS

1. Atlikta išsami literatūros apžvalga parodė, kad hibridinių algoritmų taikymas sprendžiant sunkius kombinatorinius uždavinius yra efektyvus būdas surasti geros kokybės (galimai optimalius) sprendinius per priimtiną vykdymo laiką.
2. Pasiūlytas ir realizuotas inovatyvus hibridinis genetinis algoritmas (HGA). Esminės HGA savybės: a) algoritme įkomponuota hierarchinė iteratyvioji tabu paieška; b) panaudojama labai geros kokybės diversifikuotų sprendinių populiacija; c) pritaikytas efektyvus tabu paieškos ir godžiojo adaptyviojo sprendinių pertvarkymo kombinavimas.
3. Atliktas HGA efektyvumo eksperimentinis tyrimas leidžia daryti išvadą, kad HGA yra efektyvesnis laiko (kai lyginamieji eksperimentai atlikti su vidutinio dydžio uždaviniu, $n = 256$) ir sprendinių kokybės atžvilgiu (kai lyginamieji eksperimentai atlikti su dideliu uždaviniu, $n = 1024$) už iki šiol žinomus algoritmus, skirtus spręsti PŠF uždaviniui.
   Atlikus eksperimentus su skirtingomis algoritmo komponentų modifikacijomis, sudarytas komponentų rinkinys, tinkamas išplėstiniams eksperimentams. Ištyrus ir eksperimentiškai įvertinus HGA komponentus, paaiškėjo, kad iš komponentų modifikacijų sudarius komponentų rinkinį galima gauti efektyvesnius rezultatus.
4. Nustatyta, jog efektyviausi yra šie komponentų variantai: pagerinta pradinė populiacija, padidintas sprendinių pagerinimo (t. y. tabu paieškos) iteracijų skaičius, kryžminimo procedūros, kuriomis įvertinamos tikslo funkcija ir uždavinio specifika, operatyvus populiacijos atnaujinimas atsižvelgiant į atstumo kriterijų, neaukšto lygio mutacijomis paremtų populiacijos pertvarkymų taikymas.

LITERATŪROS SĄRAŠAS

1. MISEVIČIUS, A., BLONSKIS, J., BUKŠNAITIS, V. Kombinatorinis optimizavimas ir metaeuristiniai metodai: teoriniai aspektai. *Informacijos mokslai (Information Sciences)*, mokslo darbai. 2007, T. 42–43, 213–219 (lietuvių kalba).
2. OSMAN, I. H., KELLY, J. P. Meta-heuristics: an overview. In I. H. Osman, J. P. Kelly (eds.), *Meta-Heuristics: Theory and Applications*. 1996, 1–21.
3. PAPADIMITRIOU, C. H., STEIGLITZ, K. *Combinatorial Optimization: Algorithms and Complexity*. Englwood Cliffs: Prentice-Hall, 1982.
4. BLUM, C., ROLI, A. Metaheuristics in combinatorial optimization: overview and conceptual comparison. *ACM Computing Surveys*. 2003, Vol. 35, 268–308.
5. MICHALEWICZ, Z., FOGEL, D. B. *How to Solve It: Modern Heuristics*. Berlin-Heidelberg: Springer, 2000.
6. AARTS, E. H. L., LENSTRA, J. K. (eds.). *Local Search in Combinatorial Optimization*. Chichester: Wiley, 1997.
7. ÇELA, E. *The Quadratic Assignment Problem: Theory and Algorithms*. Dordrecht: Kluwer, 1998.
8. FINKE, G., BURKARD, R. E., RENDL, F. Quadratic assignment problems. *Annals of Discrete Mathematics*. 1987, Vol. 31, 61–82.
9. LOIOLA, E. M., DE ABREU, N. M. M., BOAVENTURA-NETTO, P. O., HAHN, P., QUERIDO, T. A survey for the quadratic assignment problem. *European Journal of Operational Research.* 2007, Vol. 176, 657–690.
10. TAILLARD, E. Comparison of iterative searches for the quadratic assignment problem. *Location Science*. 1995, Vol. 3, 87–105.
11. LAPORTE, G. The traveling salesman problem: an overview of exact and approximate algorithms. *European Journal of Operational Research*. 1992, Vol. 59, 231–247.
12. JOHNSON, D. S., MCGEOCH, L. A. The traveling salesman problem: a case study. In E. Aarts, J. K. Lenstra (eds.), *Local Search in Combinatorial Optimization*. Chichester: Wiley, 1997, 21–310.
13. JOHNSON, D. S. Local optimization and the traveling salesman problem. In M. Paterson (ed.), *Automata, Languages and Programming, 17th International Colloquium, ICALP90, Proceedings. Lecture Notes in Computer Science*. Berlin: Springer, 1990, Vol. 443, 446–461.
14. MARTA, R., GALLEGO, M., DUARTE, A. and PARDO, E. G. Heuristics and Metaheuristics for the Maximum Diversity Problem. *Journal of Heuristics*. 2013, Vol. 19, no. 4, 591–615, ISSN 1572-9397.
15. ZHOU, Y., HAO, J.-K., DUVAL, B. Opposition-based memetic search for the maximum diversity problem. *IEEE Transactions on Evolutionary Computation*. 2017, Vol. 21, 731–745.
16. WU, Q., HAO, J.-K. A hybrid metaheuristic method for the maximum diversity problem. *European Journal of Operational Research*. 2013, Vol. 231, 452–464.
17. ROLLAND, E., PIRKUL, H., GLOVER, F. Tabu search for graph partitioning. *Annals of Operations Research*. 1996, Vol. 63, 209–232.

18. MISEVIČIUS, A., BLONSKIS, J., BUKŠNAITIS, V., STANEVIČIENĖ, E., ŽELVYS, T. Aukštos kokybės skaitmeninių spalvų atspalvių kompiuterinis generavimas. *Computational Science and Techniques.* 2013, Vol. 1, no. 2, 126–140. ISSN 2029-9966.
19. MISEVIČIUS, A., GUOGIS, E., STANEVIČIENĖ, E. Computational algorithmic generation of high-quality colour patterns. In T. Skersys, R. Butleris, R. Butkienė (eds.), *Information and software technologies, 19th International Conference, ICIST 2013, Proceedings, Communications in Computer and Information Science (CCIS)*. Berlin-Heidelberg: Springer, 2013, Vol. 403, 285–296.
20. MISEVIČIUS, A. Generation of grey patterns using an improved genetic-evolutionary algorithm: some new results. *Information Technology and Control*. 2011, Vol. 40, 330–343.
21. MISEVIČIUS, A., STANEVIČIENĖ, E. A new hybrid genetic algorithm for the grey pattern quadratic assignment problem. *Information Technology and Control*. 2018, Vol. 47(3), 503–520.
22. LAU, D. L., ARCE, G. R. *Modern Digital Halftoning*. Sec. Ed. (Signal Processing and Communications Series, Liu, K. J. R. (ed.)). New York-Basel: Marcel Dekker, 2008.
23. ULICHNEY, R. A. *Digital Halftoning*. London: MIT Press, 1987.
24. DREZNER, Z. Finding a cluster of points and the grey pattern quadratic assignment problem. *OR Spectrum.* 2006, Vol. 28, 417–436.
25. TAILLARD, E. Robust taboo search for the QAP. *Parallel Computing*. 1991, Vol. 17, 443–455.
26. BATTITI, R., TECCHIOLLI, G. The reactive tabu search. *ORSA Journal on Computing*. 1994, Vol. 6, 126–140.
27. TALBI, E. G., HAFIDI, Z., GEIB, J. M. Parallel tabu search for large optimization problems. *In Meta-Heuristics: Advances and Trends in Local Search Paradigms for Optimization*. 1998, 345–358.
28. TAILLARD, E., GAMBARDELLA, L. M. Adaptive memories for the quadratic assignment problem. Tech. Report IDSIA-87-97. Lugano, Switzerland, 1997.
29. GAMBARDELLA, L. M., TAILLARD, E., DORIGO, M. Ant colonies for the quadratic assignment problem. *Journal of the Operational Research Society*. 1999, Vol. 50, 167–176.
30. FLEURENT, C., FERLAND, J. A. Genetic hybrids for the quadratic assignment problem. In P. M. Pardalos, H. Wolkowicz (eds.), *Quadratic Assignment and Related Problems. DIMACS Series in Discrete Mathematics and Theoretical Computer Science*. Providence: AMS, 1994, Vol. 16, 173–188.
31. LI, Y., PARDALOS, P. M., RESENDE, M. G. C. A greedy randomized adaptive search procedure for the quadratic assignment problem. In P. M. Pardalos, H. Wolkowicz (eds.), *Quadratic Assignment and Related Problems. DIMACS Series in Discrete Mathematics and Theoretical Computer Science*. Providence: AMS, Vol. 16, 1994, 237–261.
32. SUBHADRA, A. Greedy Algorithms: Analysis, Design & Applications. *International Journal of Informative & Futuristic Research.* 2016, Vol. 3(5), 1749–1764.

33. TEITZ, M. B., BART, P. Heuristic methods for estimating the generalized vertex median of a weighted graph. *Operation Research*. 1968, Vol. 16(5), 955–961.
34. FRANZIN, A., STÜTZLE, T. Revisiting simulated annealing: A component-based analysis. *Computers & Operations Research*. 2019, Vol. 104, 191–206.
35. KIRKPATRICK, S., GELAT, C. D., VECCHI, M. P. Optimization by simulated annealing. *Science*. 1983, Vol. 220(4598), 671–680.
36. COLORNI, A., DORIGO, M., MANIEZZO, V. Distributed Optimization by Ant Colonies. *Proceedings of the European Conference on Artificial Life*. Elsevier Publishing, 1991, 134–142.
37. HANSEN, P., JAUMARD, B. Algorithms for the maximum satisfiability problem. *RUTCOR Research Report.* USA: Rutgers University, 1987, 43–87.
38. GLOVER, F. Tabu search: part I. *ORSA Journal on Computing*. 1989, Vol. 1, 190–206.
39. GLOVER, F. Tabu search: part II. ORSA Journal on Computing. 1990, Vol. 2, 4–32.
40. DELL'AMICO, M., TRUBIAN, M. Applying tabu search to the job-shop scheduling problem. *Annals of Operations Research*. 1993, Vol. 41, 231–252.
41. LOURENCO, H. R., ZWIJNENBURG, M. Combining the large-step optimization with tabu search: application to the job-shop scheduling problem. In I. H. Osman, J. P. Kelly (eds.), *Meta-Heuristics: Theory and Applications*. Boston: Kluwer, 1996, 219–236.
42. GLOVER, F., LAGUNA, M. *Tabu Search*. Dordrecht: Kluwer, 1997.
43. GLOVER, F., MARTI, R. Metaheuristic Procedures for Training Neutral Networks (eds.). *Tabu Search*. Boston, MA: Springer US, 2006, 53–69.
44. MISEVICIUS, A. A tabu search algorithm for the quadratic assignment problem. *Computational Optimization and Applications*. 2005, Vol. 30, 95–111.
45. HERTZ, A., DE WERRA, D. Using tabu search techniques for graph coloring. *Computing*. 1987, Vol. 39, 345–351.
46. GENDREAU, M. An introduction to tabu search. In F. Glover, G. Kochenberger (eds.), *Handbook of Metaheuristics*. Norwell: Kluwer, 2002, 37–54.
47. GENDREAU, M., POTVIN, J. (eds.). *Handbook of Metaheuristics* (*International Series in Operations Research & Management Science*). New York-Dordrecht-Heidelberg-London: Springer, 2010.
48. HOLLAND, J. H. *Adaptation in Natural and Artificial Systems*. Ann Arbor: Michigan Press, 1975.
49. LUKE, S. *Essentials of Metaheuristics* [interaktyvus]. Lulu, 2009 [žiūrėta 2017-02-10]. Prieiga per https://cs.gmu.edu/~sean/book/metaheuristics/ Essentials.pdf
50. YANG, X.-S. *Nature-Inspired Metaheuristic Algorithms*. Frome: Luniver Press, 2010.
51. HERTZ, A., KOBLER, D. A framework for the description of evolutionary algorithms. *European Journal of Operational Research*. 2000, Vol. 126, 1–12.
52. GOLDBERG, D. E. *Genetic Algorithms in Search, Optimization and Machine Learning*. Reading: Addison-Wesley, 1989.
53. HAUPT, R. L., HAUPT, S. E. *Practical Genetic Algorithms*. Hoboken: Wiley-Interscience, 2004.

54. LIM, T. Y. Structured population genetic algorithms: a literature survey. *Artificial Intelligence Review*. 2014, Vol. 41, 385–399.
55. MISEVIČIUS, A., et al. *Genetiniai algoritmai komivojažieriaus uždaviniui: negatyvieji ir pozityvieji aspektai. Informacijos mokslai*. 2009, Vol. 50, 173–180.
56. MOSCATO, P. On evolution, search, optimization, genetic algorithms and martial arts: toward memetic algorithms. *Tech. Report Caltech Concurrent Computation Program 826*. Pasadena, USA: California Institute of Technology, 1989.
57. MOSCATO, P., COTTA, C. A gentle introduction to memetic algorithms. In F. Glover, G. Kochenberger (eds.), *Handbook of Metaheuristics*. Norwell: Kluwer, 2002, 105–144.
58. MISEVIČIUS, A., BLONSKIS, J., BUKŠNAITIS, V. Euristiniai algoritmai: tikslai, iššūkiai, metodologija, perspektyvos. *Informacijos mokslai (Information Sciences)*, mokslo darbai, 2006, T. 39, 103–112 (lietuvių kalba).
59. GROSAN, C., ABRAHAM, A., ISHIBUCHI, H. *Hybrid Evolutionary Algorithms*. Studies in Computational Intelligence, 2007.
60. DREZNER, Z., MISEVIČIUS, A. Enhancing the performance of hybrid genetic algorithms by differential improvement. *Computers & Operations Research*. 2013, Vol. 40, 1038–1046.
61. TATE, D. M., SMITH, A. E. A genetic approach to the quadratic assignment problem. *Computers & Operations Research*. 1995, Vol. 1, 73–83.
62. MISEVICIUS, A., KUZNECOVAITE, D., PLATUZIENE, J. Some Further Experiments with Crossover Operators for Genetic Algorithms. *Informatica*. 2018, Vol. 29(3), 499–51.
63. MISEVIČIUS, A. Testing of crossover operators for the grey pattern problem. *Ūkio technologinis ir ekonominis vystymas (Technological and Economic Development of Economy)*. 2006, Vol. 12(1), 37–43.
64. MISEVIČIUS, A. Experiments with hybrid genetic algorithm for the grey pattern problem. *Informatica*. 2006, Vol. 17(2), 237–258.
65. MERZ, P., FREISLEBEN, B. Fitness landscape analysis and memetic algorithms for the quadratic assignment problem. *IEEE Transactions on Evolutionary Computation*. 2000, Vol. 4, 337–352.
66. AHUJA, R. K., ORLIN, J. B., TIWARI, A. A greedy genetic algorithm for the quadratic assignment problem. *Computers & Operations Research*. 2000, Vol. 27, 917–934.
67. LIM, M. H., YUAN, Y., OMATU, S. Efficient genetic algorithms using simple genes exchange local search policy for the quadratic assignment problem. *Computational Optimization and Applications*. 2000, Vol. 15, 249–268.
68. DREZNER, Z. A new genetic algorithm for the quadratic assignment problem. *INFORMS Journal on Computing*. 2003, Vol. 15, 320–330.
69. BOESE, K. D., KAHNG, A. B., MUDDU, S. A new adaptive multi-start technique for combinatorial global optimizations. *Operations Research Letters*. 1994, Vol. 16, 101–113.
70. MISEVIČIUS, A., RUBLIAUSKAS, D. Performance of hybrid genetic algorithm for the grey pattern problem. *Information Technology and Control*. 2005, Vol. 34, 15–24.

71. DREZNER, Z., MISEVIČIUS, A., PALUBECKIS, G. Exact algorithms for the solution of the grey pattern quadratic assignment problem. *Mathematical Methods of Operations Research*. 2015, Vol. 82, 85–105.
72. KOOPMANS, T., BECKMANN, M. Assignment problems and the location of economic activities. *Econometrica*. 1957, Vol. 25, 53–76.
73. GAREY, M. R., JOHNSON, D. S. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. San Francisco: Freeman, 1979.
74. PARDALOS, P. M., RENDL, F., WOLKOWICZ, H. The quadratic assignment problem: a survey and recent developments. In P. M. Pardalos, H. Wolkowicz (eds.), *Quadratic Assignment and Related Problems*. *DIMACS Series in Discrete Mathematics and Theoretical Computer Science*. Vol. 16, Providence: AMS, 1994, 1–41.
75. MISEVIČIUS, A. A modified simulated annealing algorithm for the quadratic assignment problem. *Informatica*. 2003, Vol. 14, 497–514.
76. ANGEL, E. ZISSIMOPOULOS, V. On the hardness of the quadratic assignment problem with metaheuristics. *Journal of Heuristics*. 2002, Vol. 8, 399–414.
77. DREZNER, Z. A new heuristic for the quadratic assignment problem. *Journal of Applied Mathematics and Decision Sciences*. 2002, Vol. 6, 143–153.
78. DREZNER, Z. Compounded genetic algorithms for the quadratic assignment problem. *Operations Research Letters*. 2005, Vol. 33, 475–480.
79. GILMORE, P. C. Optimal and suboptimal algorithms for the quadratic assignment problem. *SIAM Journal of Applied Mathematics*. 1962, Vol. 14, 305–313.
80. LAWLER, E. L. The quadratic assignment problem. *Management Science*. 1963, Vol. 9, 586–599.
81. STANEVIČIENĖ, E., MISEVIČIUS, A. Empirical Study of Local Optimization Methods. *ALTA'17: Pažangios mokymosi technologijos: išmanusis mokymasis: tarptautinė konferencija, skirta IT idėjų sklaidai: konferencijos pranešimų medžiaga*. Kaunas: Kauno technologijos universitetas, 2017, 91–99.
82. TIZHOOSH, H. R. Opposition-based learning: A new scheme for machine intelligence. In M. Mohammadian (ed.), *Proceedings of International Conference on Computational Intelligence for Modeling, Control and Automation and International Conference on Intelligent Agents, Web Technologies and Internet Commerce (CIMCA/IAWTIC)*. Los Alamitos: IEEE Press, 2005, Vol. 2, 695−701.
83. FEO, T. A., RESENDE, M. G. C. Greedy randomized adaptive search procedures. *Journal of Global Optimization*. 1995, Vol. 6, 109–133.
84. HUSSIN, M. S., STÜTZLE, T. Hierarchical iterated local search for the quadratic assignment problem. In M. J. Blesa, C. Blum, L. Di Gaspero, A. Roli, M. Sampels, A. Schaerf (eds.), *Hybrid Metaheuristics. HM 2009. Lecture Notes in Computer Science*. Berlin, Heidelberg: Springer, 2009, Vol. 5818, 115−129.
85. DELL'AMICO, M., TRUBIAN, M. Solution of large weighted equicut problems. *European Journal of Operational Research*. 1998, Vol. 106, 500−521.
86. STANEVIČIENĖ, E., MISEVIČIUS, A., OSTREIKA, A. Experimental Analysis of Hybrid Genetic Algorithm for the Grey Pattern Quadratic Assignment Problem. *Information Technology and Control.* 2019, Vol. 48(2), 335–356.

87. MISEVIČIUS, A., RUBLIAUSKAS, D. Testing of hybrid genetic algorithms for structured quadratic assignment problems. *Informatica*. 2009, Vol. 20, 255–272.
88. LIPOWSKI, A., LIPOWSKA, D. Roulette-wheel selection via stochastic acceptance. *Physica A: Statistical Mechanics and its Applications*. 2012, Vol. 391(6), 2193–2196.
89. RAZALI, N., M., JOHN, G. Genetic Algorithm Performance with Different Selection Strategies in Solving TSP. *In the 2011 International Conference of Computational Intelligence and Intelligent Systems*. London: Imperial College, 2011, Vol. 2, 4–9.
90. ZHONG, J., HU, X., ZHANG, J., GU, M. Comparison of Performance between Different Selection Strategies on Simple Genetic Algorithms. *International Conference on Computational Intelligence for Modelling, Control and Automation and International Conference on Intelligent Agents, Web Technologies and Internet Commerce (CIMCA-IAWTIC'06)*. Vienna, 2005, 1115–1121.
91. KUMAR, R. Blending Roulette Wheel Selection & Rank Selection in Genetic Algorithms. *International Journal of Machine Learning and Computing*. 2012, Vol. 2, No. 4, 365–370.
92. EIBEN, A. E., RAUE, P.-E., RUTTKAY, Z. Genetic algorithms with multi-parent recombination. In Y. Davidor, H. P. Schwefel, R. Männer (eds.), *Parallel Problem Solving from Nature — PPSN III, International Conference on Evolutionary Computation, The Third Conference on Parallel Problem Solving from Nature, Proceedings. Lecture Notes in Computer Science*. Berlin-Heidelberg: Springer, 1994, Vol. 866, 78–87.
93. CAPONETTO, R., FORTUNA, L., FAZZINO, S., XIBILIA, M. G. Chaotic sequences to improve the performance of evolutionary algorithms. *IEEE Transactions on Evolutionary Computation*. 2003, Vol. 7(3), 289–304.

AUTORĖS MOKSLINIŲ PUBLIKACIJŲ DISERTACIJOS TEMA SĄRAŠAS

Mokslinės informacijos instituto duomenų bazės „ISI Web of Science“ leidiniuose, turinčiuose citavimo indeksą, paskelbti straipsniai:

1. MISEVIČIUS, A., STANEVIČIENĖ, E. A New Hybrid Genetic Algorithm for the Grey Pattern Quadratic Assignment Problem. *Information Technology and Control.* 2018, Vol. 47(3), 503–520.
2. STANEVIČIENĖ, E., MISEVIČIUS, A., OSTREIKA, A. Experimental Analysis of Hybrid Genetic Algorithm for the Grey Pattern Quadratic Assignment Problem. *Information Technology and Control.* 2019, Vol. 48(2), 335–356.

Kitų tarptautinių duomenų bazių leidiniuose paskelbti straipsniai:

1. MISEVIČIUS, A., BLONSKIS, J., BUKŠNAITIS, V., STANEVIČIENĖ, E., ŽELVYS, T. Aukštos kokybės skaitmeninių spalvų atspalvių kompiuterinis generavimas. *Computational Science and Techniques.* Klaipėda: Klaipėda University, 2013, Vol. 1, no. 2, 126–140.
2. MISEVIČIUS, A., GUOGIS, E., STANEVIČIENĖ, E. Computational Algorithmic Generation of High-quality Colour Patterns. *Information and software technologies: 19th international conference, ICIST 2013, Kaunas, Lithuania, October 10–11, 2013: proceedings / [edited by] Tomas Skersys, Rimantas Butleris, Rita Butkiene. Communications in computer and information science.* Berlin, Heidelberg: Springer, 2013, Vol. 403, 285–296.
3. STANEVIČIENĖ, E., MISEVIČIUS, A., KUZNECOVAITĖ, D., DRĄSUTĖ, V. Hybrid Genetic Algorithms for Image Processing: a Short Review. *ALTA'15: Pažangios mokymosi technologijos: konferencijos pranešimų medžiaga*. Kaunas: Kauno technologijos universitetas, 2015, 45–49.
4. STANEVIČIENĖ, E., MISEVIČIUS, A. Empirical Study of Local Optimization Methods. *ALTA'17: Pažangios mokymosi technologijos: išmanusis mokymasis: tarptautinė konferencija, skirta IT idėjų sklaidai: konferencijos pranešimų medžiaga*. Kaunas: Kauno technologijos universitetas, 2017, 91–99.

A PRIEDAS

Priede pateikiamas eksperimentų vykdymo laiko vidurkis (vykdyta 10 kartų).

A1 lentelė. Eksperimentų su komponentais **`HGA-BV`**, **`PP-1`**, **`PP-2`**, **`STA-1`**, **`STA-2`** vidutinis vykdymo laikas ($n = 256$)

m	*Vykdymo laikas*					**Laikas‡ (s)**
	HGA-BV	**PP-1**	**PP-2**	**STA-1**	**STA-2**	
95	65,1	33,4	85,8	56,5	176,1	*83*
96	70,6	32,8	249,6	89,8	122,0	*113*
97	81,8	31,9	129,3	60,6	101,2	*81*
98	82,3	31,5	211,5	84,9	85,7	*99*
99	81,0	31,0	210,4	88,3	99,9	*102*
100	85,6	30,7	348,5	81,4	122,5	*134*
101	85,7	30,0	309,9	79,4	93,5	*120*
102	74,6	29,3	316,6	72,2	86,6	*116*
103	66,8	28,5	298,4	65,2	106,7	*113*
104	66,9	28,5	286,6	64,8	84,9	*106*

‡ – vidutinis vykdymo laikas (sekundėmis)

A2 lentelė. Eksperimentų su komponentais **`HGA-BV`**, **`PP-1`**, **`PP-2`**, **`STA-1`**, **`STA-2`** vidutinis vykdymo laikas ($n = 1024$)

m	*Vykdymo laikas*					**Laikas (s)**
	HGA-BV	**PP-1**	**PP-2**	**STA-1**	**STA-2**	
50	175,8	357,8	414,7	177,9	229,9	*271*
60	2469,6	771,6	2277,1	2420,5	3131,0	*2214*
70	2769,9	802,0	3609,3	2793,7	6322,5	*3259*
80	2433,1	809,6	3201,6	2422,8	2321,3	*2238*
90	2370,5	804,6	3217,9	2465,7	6239,3	*3020*
100	2236,2	743,3	2829,2	2198,7	2662,5	*2134*
110	2194,9	711,0	2114,9	2315,5	6357,0	*2739*
120	2375,8	710,3	2001,6	2407,1	6149,5	*2729*
130	2368,1	709,1	1893,0	2640,7	4170,3	*2356*
140	2098,2	688,8	1869,0	2203,3	4179,9	*2208*

A3 lentelė. Eksperimentų su komponentais **HGA-BV**, **LP-1**, **LP-2**, **LP-3**, **LP-4** vidutinis vykdymo laikas ($n = 256$)

m	*Vykdymo laikas*					Laikas‡ (s)
	HGA-BV	**LP-1**	**LP-2**	**LP-3**	**LP-4**	
95	65,1	44,4	125,4	123,0	79,1	*87*
96	70,6	57,0	116,2	179,1	106,3	*106*
97	81,8	49,0	116,4	165,4	100,3	*103*
98	82,3	54,1	122,0	179,1	102,3	*108*
99	81,0	53,3	136,7	156,9	95,3	*105*
100	85,6	55,6	126,3	153,9	91,4	*103*
101	85,7	55,2	108,2	145,8	89,6	*97*
102	74,6	52,1	91,8	119,5	89,7	*86*
103	66,8	49,7	84,8	110,9	77,9	*78*
104	66,9	49,5	86,0	105,6	81,1	*78*

A4 lentelė. Eksperimentų su komponentais **HGA-BV**, **LP-1**, **LP-2**, **LP-3**, **LP-4** vidutinis vykdymo laikas ($n = 1024$)

m	*Vykdymo laikas*					Laikas (s)
	HGA-BV	**LP-1**	**LP-2**	**LP-3**	**LP-4**	
50	175,8	122,1	193,7	296,6	296,4	*217*
60	2469,6	1440,2	3712,3	6338,7	2951,9	*3383*
70	2769,9	1492,1	4007,3	7012,2	3395,1	*3735*
80	2433,1	1475,6	3861,5	5392,7	3058,8	*3244*
90	2370,5	1539,1	3651,9	6578,0	2968,7	*3422*
100	2236,2	1277,9	2940,4	5668,2	2785,0	*2982*
110	2194,9	1361,9	3214,2	5300,3	2765,1	*2967*
120	2375,8	1381,7	3419,7	6908,9	2794,6	*3376*
130	2368,1	1399,2	3657,1	6853,8	2782,0	*3412*
140	2098,2	1248,2	3298,3	5775,2	2735,8	*3031*

A5 lentelė. Eksperimentų su komponentais **HGA-BV**, **PA-1**, **PA-2**, **PA-3**, **PA-4** vidutinis vykdymo laikas ($n = 256$)

m	*Vykdymo laikas*					Laikas (s)
	HGA-BV	**PA-1**	**PA-2**	**PA-3**	**PA-4**	
95	65,1	69,8	40,3	106,3	43,9	*65*
96	70,6	93,4	47,2	182,4	45,0	*88*
97	81,8	80,1	46,5	146,5	47,9	*81*
98	82,3	86,6	45,9	169,6	46,8	*86*
99	81,0	80,2	45,0	162,0	45,1	*83*
100	85,6	85,6	44,2	147,7	43,9	*81*
101	85,7	76,5	42,8	149,1	42,1	*79*
102	74,6	71,0	40,8	108,8	40,5	*67*
103	66,8	71,4	39,2	100,1	39,3	*63*
104	66,9	70,0	39,4	99,7	40,2	*63*

A6 lentelė. Eksperimentų su komponentais **`HGA-BV`**, **`PA-1`**, **`PA-2`**, **`PA-3`**, **`PA-4`** vidutinis vykdymo laikas ($n = 1024$)

m	*Vykdymo laikas*					**Laikas (s)**
	`HGA-BV`	**`PA-1`**	**`PA-2`**	**`PA-3`**	**`PA-4`**	
50	175,8	176,8	177,8	298,2	176,3	*201*
60	2469,6	2471,7	1117,5	5805,6	1086,4	*2590*
70	2769,9	2814,4	1108,1	6952,2	1142,1	*2957*
80	2433,1	2241,5	1155,1	4990,2	1171,5	*2398*
90	2370,5	2529,7	1127,6	5372,7	1137,0	*2508*
100	2236,2	2298,0	1032,2	5151,1	1045,1	*2353*
110	2194,9	2178,4	1015,1	5007,3	987,3	*2277*
120	2375,8	2365,6	963,3	6314,4	983,6	*2601*
130	2368,1	2481,1	968,0	6081,2	956,4	*2571*
140	2098,2	2050,8	987,9	5074,9	950,1	*2232*

A7 lentelė. Eksperimentų su komponentais **`HGA-BV`**, **`KP-1`**, **`KP-2`**, **`KP-3`**, **`KP-5`**, **`KP-6`**, **`KP-7`** vidutinis vykdymo laikas ($n = 256$)

m	*Vykdymo laikas*							**Laikas (s)**
	`HGA-BV`	**`KP-1`**	**`KP-2`**	**`KP-3`**	**`KP-5`**	**`KP-6`**	**`KP-7`**	
95	65,1	103,2	57,6	58,2	261,0	64,7	105,4	*102*
96	70,6	162,3	92,7	181,7	325,3	80,9	154,9	*153*
97	81,8	151,7	59,9	140,5	292,9	78,7	141,6	*135*
98	82,3	175,2	92,5	167,1	306,2	88,1	180,0	*156*
99	81,0	155,2	85,1	161,8	248,1	87,0	170,8	*141*
100	85,6	158,1	89,9	142,9	252,1	81,6	180,7	*142*
101	85,7	141,9	88,2	143,2	232,0	78,2	189,1	*137*
102	74,6	120,3	80,1	117,2	193,1	75,3	180,4	*120*
103	66,8	102,6	75,5	104,4	164,1	70,6	180,4	*109*
104	66,9	106,7	72,9	101,9	169,5	69,7	183,5	*110*

A8 lentelė. Eksperimentų su komponentais **`HGA-BV`**, **`KP-1`**, **`KP-2`**, **`KP-3`**, **`KP-5`**, **`KP-6`**, **`KP-7`** vidutinis vykdymo laikas ($n = 1024$)

m	*Vykdymo laikas*							**Laikas (s)**
	`HGA-BV`	**`KP-1`**	**`KP-2`**	**`KP-3`**	**`KP-5`**	**`KP-6`**	**`KP-7`**	
50	175,8	174,1	195,6	176,1	301,8	177,4	195,3	*199*
60	2469,6	4739,5	2527,8	4040,3	691,5	1921,7	4049,0	*2920*
70	2769,9	5598,3	3008,4	5162,7	1845,6	2826,7	6137,7	*3907*
80	2433,1	4116,3	2000,8	2811,2	653,6	2438,1	4565,5	*2717*
90	2370,5	4948,7	2729,9	4598,5	88,0	2740,0	5635,5	*3302*
100	2236,2	4409,0	2334,5	4237,4	8226,2	2239,0	4531,0	*4030*
110	2194,9	4443,5	2354,9	4311,0	8058,3	2166,1	4522,9	*4007*
120	2375,8	4794,0	2428,6	3954,2	1130,5	2024,5	3345,9	*2865*
130	2368,1	4746,2	2558,9	3927,8	8605,0	1843,6	2998,4	*3864*
140	2098,2	4585,9	2386,8	4050,0	8305,3	2022,6	3260,0	*3816*

A9 lentelė. Eksperimentų su komponentų rinkiniais: **STA-2**, **KP-6**, **KS-1**, **LP-2**, **PR-2**; **STA-2**, **KP-5**, **KS-1**, **LP-2**, **PR-2**; **STA-2**, **KP-5**, **KS-3**, **LP-2**, **PR-2** vidutinis vykdymo laikas ($n = 256$)

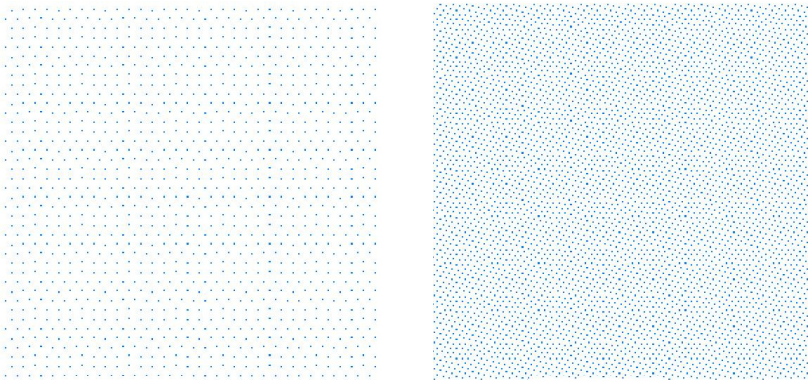
m	*Vykdymo laikas*				Laikas (s)
	HGA-BV	STA-2, KP-6, KS-1, LP-2, PR-2	STA-2, KP-5, KS-1, LP-2, PR-2	STA-2, KP-5, KS-3, LP-2, PR-2	
95	65,1	189,0	191,2	602,3	*262*
96	70,6	649,4	386,0	693,8	*450*
97	81,8	244,4	246,8	438,9	*253*
98	82,3	205,6	206,7	557,9	*263*
99	81,0	574,5	574,9	1425,9	*664*
100	85,6	643,2	558,0	1543,3	*708*
101	85,7	610,5	582,0	1621,2	*725*
102	74,6	559,8	504,9	3397,9	*1134*
103	66,8	513,2	464,1	2290,2	*834*
104	66,9	487,5	471,9	1741,0	*692*

A10 lentelė. Eksperimentų su komponentų rinkiniais: **STA-2**, **KP-6**, **KS-1**, **LP-2**, **PR-2**; **STA-2**, **KP-5**, **KS-1**, **LP-2**, **PR-2**; **STA-2**, **KP-5**, **KS-3**, **LP-2**, **PR-2** vidutinis vykdymo laikas ($n = 1024$)

m	*Vykdymo laikas*				Laikas (s)
	HGA-BV	STA-2, KP-6, KS-1, LP-2, PR-2	STA-2, KP-5, KS-1, LP-2, PR-2	STA-2, KP-5, KS-3, LP-2, PR-2	
50	175,8	886,0	885,2	2355,2	*1076*
60	2469,6	2972,7	524,9	2459,5	*2107*
70	2769,9	4237,0	2763,7	388,9	*2540*
80	2433,1	6148,7	2826,1	4910,5	*4080*
90	2370,5	3813,5	1584,6	439,2	*2052*
100	2236,2	3399,5	2236,3	1696,0	*2392*
110	2194,9	7453,9	2397,9	8145,3	*5048*
120	2375,8	7743,8	709,1	5739,5	*4142*
130	2368,1	5632,6	8060,6	5281,8	*5336*
140	2098,2	4312,7	4591,1	3856,0	*3714*

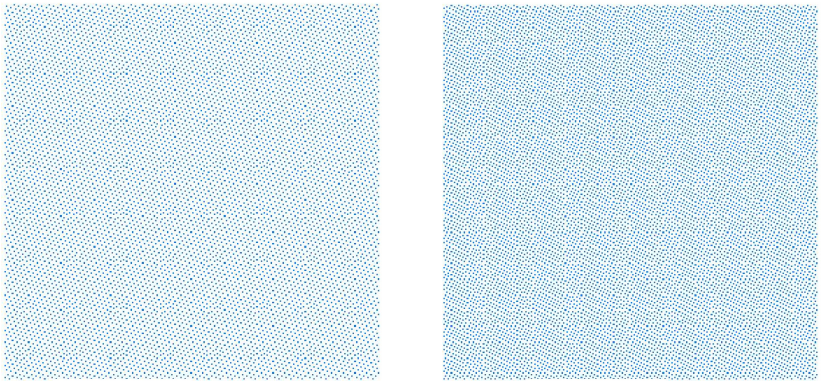
B PRIEDAS

Rastų galimai optimalių sprendinių (perstatymų) grafinis vaizdas, kai spalvos intensyvumas $m = 20, 75, 130, 185, 240, 295, 350, 405, 460, 512$, PŠF uždavinio dydis $n = 1024$, fono spalva mėlyna, taškų spalva juoda, kiekvienas tinklelis pakartotas 8 kartus vertikaliai ir 8 kartus horizontaliai, pateikiamas B1–B6 paveikslėliuose.



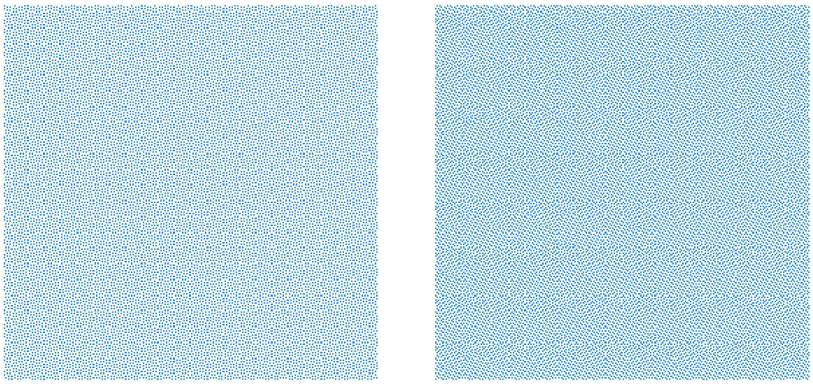
(a) (b)

B1 pav. (Pseudo)optimalių pilkųjų šablonų pavyzdžiai (32×32): a) $m = 20$; b) $m = 75$



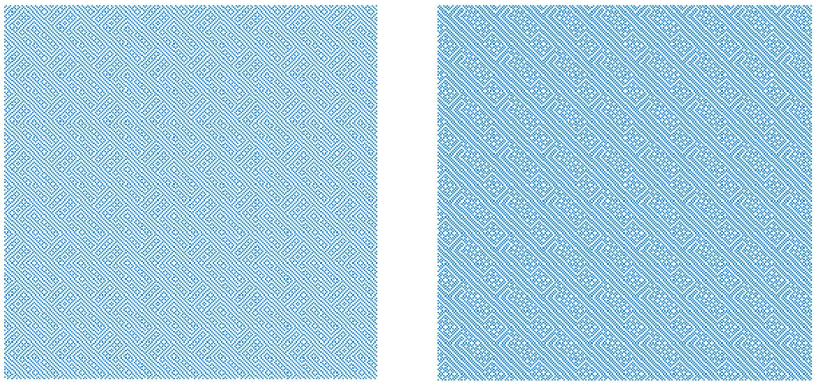
(a) (b)

B2 pav. (Pseudo)optimalių pilkųjų šablonų pavyzdžiai (32×32): a) $m = 130$; b) $m = 175$



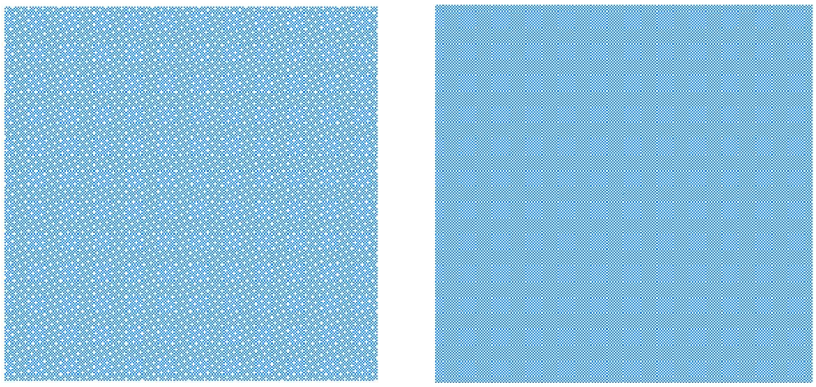
(a) (b)

B3 pav. (Pseudo)optimalių pilkųjų šablonų pavyzdžiai (32×32): a) $m = 240$; b) $m = 295$



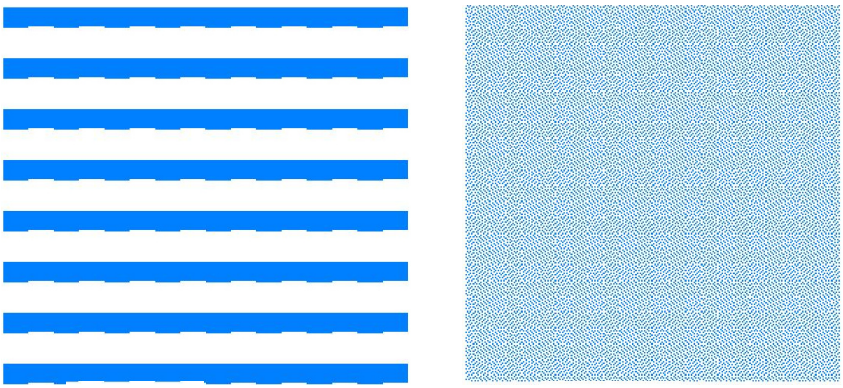
(a) (b)

B4 pav. (Pseudo)optimalių pilkųjų šablonų pavyzdžiai (32×32): a) $m = 350$; b) $m = 405$



(a) (b)

B5 pav. (Pseudo)optimalių pilkųjų šablonų pavyzdžiai (32×32): a) $m = 460$; b) $m = 512$



(a) (b)

B6 pav. Pilkųjų šablonų pavyzdžiai (32×32), kai $m = 300$: a) blogas sprendinys; b) rastas (pseudo)optimalus sprendinys

SL344. 2019-07-02, 11 leidyb. apsk. l. Tiražas 14 egz. Užsakymas 149.
Išleido Kauno technologijos universitetas, K. Donelaičio g. 73, 44249 Kaunas
Spausdino leidyklos „Technologija“ spaustuvė, Studentų g. 54, 51424 Kaunas