ŠIAULIŲ UNIVERSITETAS
MATEMATIKOS IR E. STUDIJŲ INSTITUTAS
PROGRAMŲ SYSTEMŲ KATEDRA

Donatas Kavaliauskas

Informatikos specialybės II kurso dieninio skyriaus studentas

Dirbtinės bičių kolonijos algoritmai ir jų taikymų analizė optimizavimo uždaviniams spręsti.

The Analysis of Artificial Bee Colony Algorithms and their Application to the Solving of Optimization Problems

MAGISTRO DARBAS

Darbo vadovas:
Doc. dr. G. Felinskas
Recenzentas:
Doc. Dr. L. Kaklauskas

Šiauliai, 2015

„Tvirtinu, jog darbe pateikta medžiaga nėra plagijuota ir paruošta naudojant literatūros sąraše pateiktus informacinius šaltinius bei savo tyrimų duomenis“

Darbo autoriaus _____

(vardas, pavardė, parašas)

Darbo tikslai ir uždaviniai

Tikslas:

- Išanalizuoti dirbtinės bičių kolonijos algoritmus, sukurti programinę realizaciją optimizavimo uždaviniams spręsti ir atlikti algoritmų taikymo analizę.

Uždaviniai:

- Išnagrinėti įvairius dirbtinių bičių spiečių optimizavimo algoritmus.
- Išanalizuoti maršrutų parinkimo bei jų optimizavimo uždavinių formuluotes, interpretacijas ir taikymus;
- Suprojektuoti dirbtinės bičių kolonijos algoritmų funkcijas ir šių algoritmų procedūrų praktines interpretacijas;
- Sukurti dirbtinių bičių kolonijų algoritmų programinę realizaciją ir pritaikyti optimizavimo uždaviniams spręsti;
- Atlikti sukurtos programinės įrangos testavimą, pateikti skaičiavimo rezultatus.
- Atlikti sukurtų algoritmų efektyvumo analizę.

Darbo vadovas: G. Felinskas

Tvirtinu: _____

Turinys

ĮVADAS

Viena iš aktualių informatikos mokslo sričių yra optimizavimo metodai ir algoritmai. Spręsti vis sudėtingesnius optimizavimo uždavinius, nebepakanka klasikinių algoritmų. Šių algoritmų sprendimo radimui yra pasitelkiama nestandartinius optimizavimo algoritmus, kurie būtų pagrįsti ne pavieniais sprendiniais. Vienas iš tokių algoritmų paradigmų yra dirbtinių bičių spiečiaus algoritmai.

Dirbtinių bičių spiečiaus algoritmai yra palyginti naujas narys spiečiaus intelekto šakoje. Šie algoritmai imituoja natūralų bičių elgesį modeliuojant mechanizmus kaip: maisto paieška, poravimas ar karalienės valdymas. Šiuo metu yra sukurta nevienas algoritmas, kuris remiasi bitėmis, tačiau nėra susistemintos medžiagos apie dirbtinių bičių algoritmų pranašumus vienas prieš kitą pagal panaudojimo sritį.[24].

Darbo rezultatų publikavimas ir aprobavimas

Tyrimo rezultatai buvo paskelbti šiuose moksliniuose konferencijose:

1. 5- oji Lietuvos jaunųjų mokslininkų konferencija. (2013 m. Šiauliai) „*Operacijų tyrimas ir taikymai*“;
2. 6- oji Lietuvos jaunųjų mokslininkų konferencija (2014 m. Vilnius) „*Operacijų taikymas versle ir inžinerijoje*“.

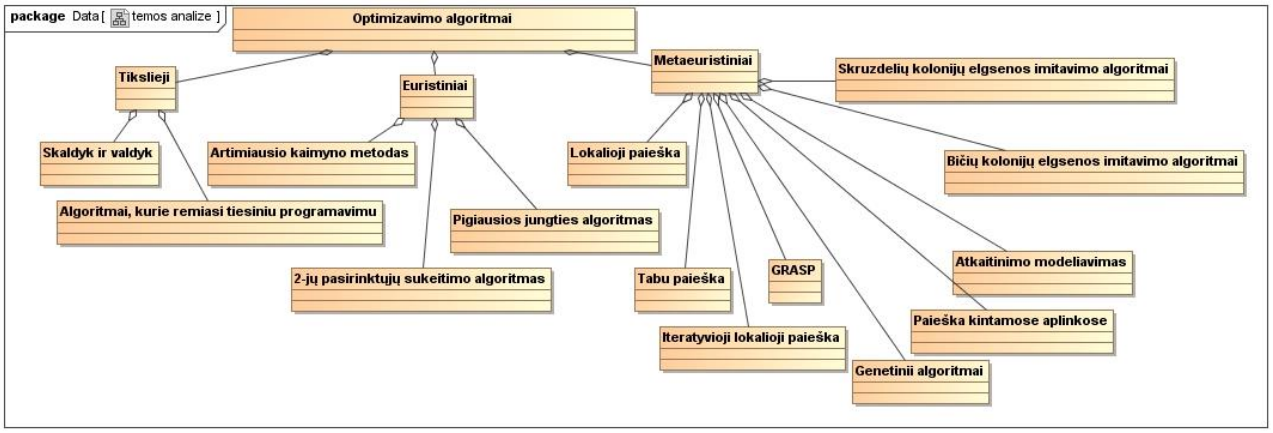
Darbo rezultatai buvo pateikti straipsnyje:

1. Donatas Kavaliauskas, Gražvydas Felinskas, „Dirbtinės bičių kolonijos algoritmai ir ų taikymai maršrutų optimizavimo uždaviniams spręsti“. „Šiaulių universiteto jaunųjų mokslininkų žurnalas“, 2014 m. NR. 1, 117-123. (ISSN 1648-8776).

1. Analitinė dalis

1.1 Temos analizė

Optimizavimas - paieška apibrėžtos aibės elemento, kuriam tam tikro kriterijaus (vadinamo **tikslo funkcija**) reikšmė būtų maksimali (kitais atvejais - minimali). Spręsti optimizavimo uždavinius yra sukurta daug algoritmų, kurie yra skirstomi į grupes (Žr. Pav. nr. 1). Tiksliųjų optimizavimo grupė garantuoja tikslų sprendinį. Tačiau šioje algoritmų grupėje galima atlikti skaičiavimus su mažu duomenų kiekiu, nes patikrinti kiekvieno duomens sprendinį laikas išauga eksponentiškai.[1],[2],[24].



Pav. 1 Optimizavimo algoritmai [24].

Euristiniai algoritmai pranoksta tiksliųjų algoritmų grupę sugaišto laiko atžvilgiu gaunant sprendinį. Tačiau gautas sprendinys gali skirtis nuo optimalaus sprendinio su leistina paklaida (asmens, kuris sprendžią uždavinį nustatyta paklaidos kriterijais). Algoritmai remiasi taisyklių rinkiniais, kurie vadinami euristikomis. Taigi, euristikos pritaiko algoritmą prie konkretaus optimizavimo uždavinio su apribojimais ir prielaidomis [1],[2],[3],[24].

Metaeuristiniai metodais yra aprašomi kurios nors klasės uždavinių sprendimo idėja, principas. Taigi, metaeuristiniai algoritmai yra virš euristinių algoritmų – kas yra išvystyta, modernizuota tradicinių euristinių algoritmų atžvilgiu. Optimizavimo uždaviniams spręsti yra naudojami metaeuristiniai algoritmai: lokalioji paieška, tabu paieška, genetiniai algoritmai, godžiosios randomizuotos adaptyvios paieškos procedūros (GRASP), skruzdėlių kolonijų elgsenos imitavimo algoritmą, paieška kintamose aplinkose, bičių kolonijų elgsenos imitavimo algoritmai ir kt. (Pav. 1).

Visi šie algoritmai yra skirti spręsti uždavinius: tvarkaraščių optimizavimui, keliaujančio pirklio uždavinys, tinklo paskirstymo optimizavimas, šviesoforų šviesų sinchronizavimui ir kt. [24].

Optimizavimas yra naudojamas spręsti šias problemas: tvarkaraščių sudarymo, maršruto sudarymas, perrikiavimo šviesoforų signalų sinchronizacija, taip pat įvairiu kitų srautų valdymas ir daugybė kitokių gyvenimiškų uždavinių.

Optimizavimo uždaviniai aktualūs įvairiose žmonių veiklos srityse. Verslininkai ir vadybininkai stengiasi darbą organizuoti taip, kad būtų gautas maksimalus pelnas, panaudojus ko mažesnius kaštus. Sunkvežimio vairuotojas, gavęs krovinį, stengiasi pasirinkti maršrutą taip, kad kelionei sugaištų kiek galima mažiau laiko ir sutaupytų kurą. Turistas, besipakuojantis kuprinę išvykai, stengiasi susidėti kuo vertingesnių daiktų pasirinktai kelionei, sutaupant vietą kuprinėje. Taigi, optimizavimas neapima vienos bendros srities. Jo panaudojimo sritys galima suskirstyti į: Maršrutų uždaviniai, perrikiavimo uždaviniai, tvarkaraščių sudarymo uždavinius, geriausių elgesio strategijų uždavinius, variantų uždavinius, sprendimų paieškos uždavinius ir kitus. [1].

1.1.1 Maršrutų uždavinys – keliaujančio pirklio uždavinys

Keliaujančio pirklio uždavinys (sutr. KPU; ang. „Traveling salesperson problem“ - TSP) yra grafų teorijos uždavinys, kai pilnajame svoriniame grafe ieškoma mažiausio svorio Hamiltono ciklo. Šio uždavinio principas yra apkeliauti visas grafo viršūnes (aplankant viršūnę tik vieną kartą), įveikiant kuo trumpesnę distanciją. [24].

Keliaujančio pirklio uždaviniai yra naudojami daugelyje sričių:

- Tvarkaraščių sudarymas;
- Transporto organizavime. Pvz. autobuso maršruto suplanavimas, kai autobusas turi apvažiuoti tam tikras stoteles;
- GPS maršruto apskaičiavimas iš taško A į tašką B;
- Pilnajame svoriniame grafe surasti mažiausio svorio Hamiltono[1] ciklą. [24].

1.1.2 Perrikiavimo uždavinys Kuprinės uždavinys

Dvejetaini kuprinės uždavinys (angl. „ binary knapsack problem“) yra gerai žinoma ir svarbi kombinatorinio optimizavimo problema. Šio uždavinio principas yra sudėti kuo daugiau

[1] Grafas, kuris yra pilnasis, yra Hamiltono grafas.

vertingesnių daiktų į kuprinę. Kiekvienas daiktas turi savo svorį ir vertę, o į kuprinę galima įdėti tik tam tikra svori daiktų. Taigi, dėdami daiktus į kuprinę, turime atsižvelgti kiekvieno daikto svorį taip pat kaip ir į jo vertę. Bandant įvairius variantus bandoma surasti tokį daiktų rinkinį, kuriuo vertė būtų didžiausia. [22].

1.1.3 Tvarkaraščių (angl. Flow-shop) uždavinys

Flow-Shop priklauso tvarkaraščio optimizavimo uždavinių klasei. Šiame uždavinyje reikia parinkti tokią vienodos sekos, tačiau skirtingos trukmės darbų eilės tvarką, kad bendras atlikimo laikas būtų trumpiausias. Uždavinyje yra du svarbūs parametrai: Darbų skaičius ir Mašinų skaičius. Kiekvienam darbui atlikti reikia vienodos mašinų sekos. Tačiau mašinos atliekamas etapas gali skirtis įvairiuose darbuose. Šiame uždavinyje tikslo funkcija yra visų darbų pabaiga - reikia kuo greičiau atlikti visus darbus. [23].

Dažniausiai tvarkaraščių uždaviniai turi papildomus kriterijus, kaip darbas „x“, negali būti pradėtas tol, kol dar nėra atliktas darbas „y“ arba „z“ darbuotojas gali atlikti greičiau darbą „v“, negu darbuotojas „j“.

1.2 Darbinės srities modelis

1.2.1 Kolektyvinis intelektas

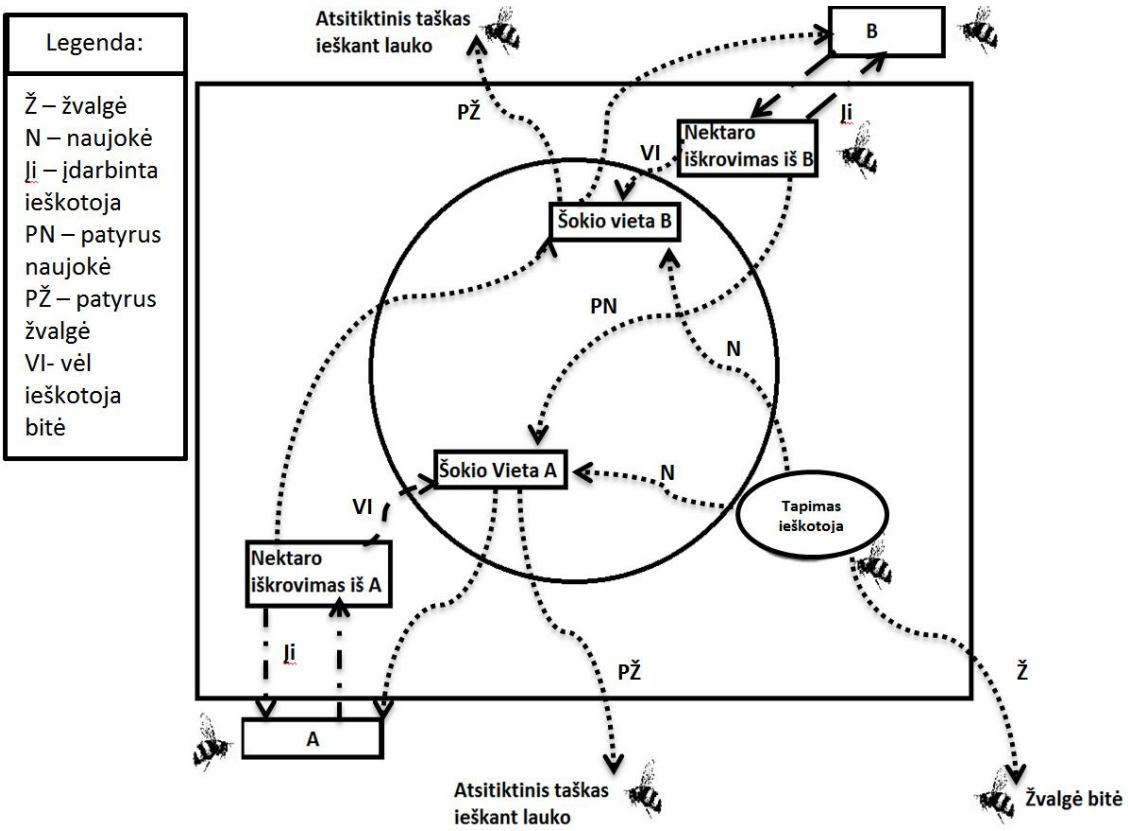
Algoritmai, kurie yra grindžiami manipuliavimu tarp sprendinių rinkinių ar grupių, šiuolaikinėje literatūroje yra vadinama kolektyvinio intelekto (angl. swarm intelligence) algoritmais. Kolektyviniu intelekto algoritmams yra priskiriama: dirbtinis bičių spiečiaus algoritmai (angl. Artificial bee colony algorithms), skruzdėlių kolonijos metodas (angl. Ant colony optimization), krioklio algoritmas (angl. Intelligent Water Drops algorithm), minios metodas (angl. Multi-swarm optimization) ir kt. Šių algoritmų pagrindas yra agentų bendradarbiavimas vienas su kitu ir supančia aplinka. Tai yra gaunant sprendinį yra komunikuojama tarp agentų, dalijimąsi informacija ir pagal tai yra priimami bendri sprendimai. Tai yra svarbiausi veiksniai, kurie leidžia siekti geriausių įmanomų rezultatų sprendžiant uždavinius [1],[3],[6],[24].

1.2.2 Bičių elgesys gamtoje

Bitės gamtoje veikia, kaip bendruomenė, kurios renka informaciją iš aplinkos ir pritaiko savo elgesį atsižvelgiant į gautą informaciją. Vienu metu vyksta įvairus darbai, kaip renkama informacija, prisitaikymas prie aplinkos, renkamas maistas ir kiti. Dėl šios priežasties yra pasiskirstomi darbai spiečiuje, ir kiekviena bitė atlieka sau suteiktą darbą. Tai įtakoja kiekvieno vabzdžio elgseną. Pagrindiniai bičių kolonijos komponentai (Žr. Pav. Nr. 2).:

- Maisto šaltinis[2] (angl. food sources). Maisto šaltinio vertė priklauso nuo įvairių parametrų, kaip jo artumą prie lizdo, maisto kiekį; (Informatikoje maisto šaltiniu laikysime sprendiniu).
- Ieškotoja (angl. foragers). Ieškotojos bitės yra skirstomos pagal savo darbus į rūšis:
    - Neįdarbintos ieškotojos (angl. unemployed foragers): vabzdys neturintis informacijos apie maisto šaltinio vietą. Yra dvi neįdarbintų bičių rūšys:
        - Žvalgė bitė[3] (angl. scout bee): tai bitė, kuri pradeda ieškoti maisto šaltinio atsitiktinai skraidydama aplinkoje;
        - Naujokė bitė (angl. recruit): tai bitė, kuri neturėdama darbo skrenda į šokio vietą[4], kad gautų iš kitų bičių informacijos apie maisto šaltinius. Šis vabzdys nusikopijuoja informacija apie maisto šaltinio vietą.
    - Įdarbinta ieškotoja (angl. employed foragers): Vabzdys, kuris žino kelią iki maisto šaltinio ir iš jo gabeną maistą į avilį. [24].

[2] Maisto šaltinio samprata informatikos terminais suprantamas, kaip sprendinys.
[3] Informatikoje žvalgės bitės sąvoka yra suprantama kaip atsitiktinis sprendinys.
[4] Šokio vietos terminas informatikos terminais reiškia: geriausio tarpinio rezultato priskyrimas atsitiktiniam sprendimui.



Pav. 2 Tipiškas bičių maitinimosi elgesys [24].

Pagal bičių elgseną gamtoje yra kuriami įvairūs optimizavimo algoritmai. Per pastarąjį dešimtmetį smarkiai pagausėjo tyrimų su bičių algoritmais literatūroje. Bičių algoritmai yra suskirstyti pagal jų elgsenas į:

- Maitinimosi elgsena, kuriuos sudaro:
  - Bičių sistema (angl. Bee System);
  - Dirbtinių bičių spiečiaus algoritmas (angl. Artificial Bee Colony Algorithm);
  - Virtualus bičių algoritmas (angl. Virtual Bee Algorithm);
  - Bičių avilys (angl. Beehive) ;
  - Bičių algoritmas (angl. Bees Algorithm) ;
  - Bičių kolonijos optimizavimo (angl. Bee Colony Optimization);
  - Ir kiti.
- Santuokos elgsena;
  - Bičių poravimosi optimizavimo algoritmas (angl. Honey-bee mating optimization algorithm) ;
  - Bičių santuokos optimizacija (angl. Marriage in Honey Bees Optimization) ;
  - Ir kiti.

- Bičių karalienė.
    - Bičių kryžminimas (angl. Bee Crossover);
    - Ir kiti. [24].

Darbe bus aptarti keli iš išvardytų algoritmų.

1.2.3 Bičių kolonijos optimizavimo algoritmas

Bičių kolonijos optimizavimo (angl. Bee Colony Optimization) idėja yra sukurti kolektyvinę bičių sistemą, galinčią spręsti sudėtingus kombinatorinias problemas. Algoritmas sudaro du pakaitomis einantys etapai: priekinio paso (angl. forward pass) ir atgalinio paso (angl backward pass). Bičių kolonijos optimizavimo algoritmą galima suformuoti taip:

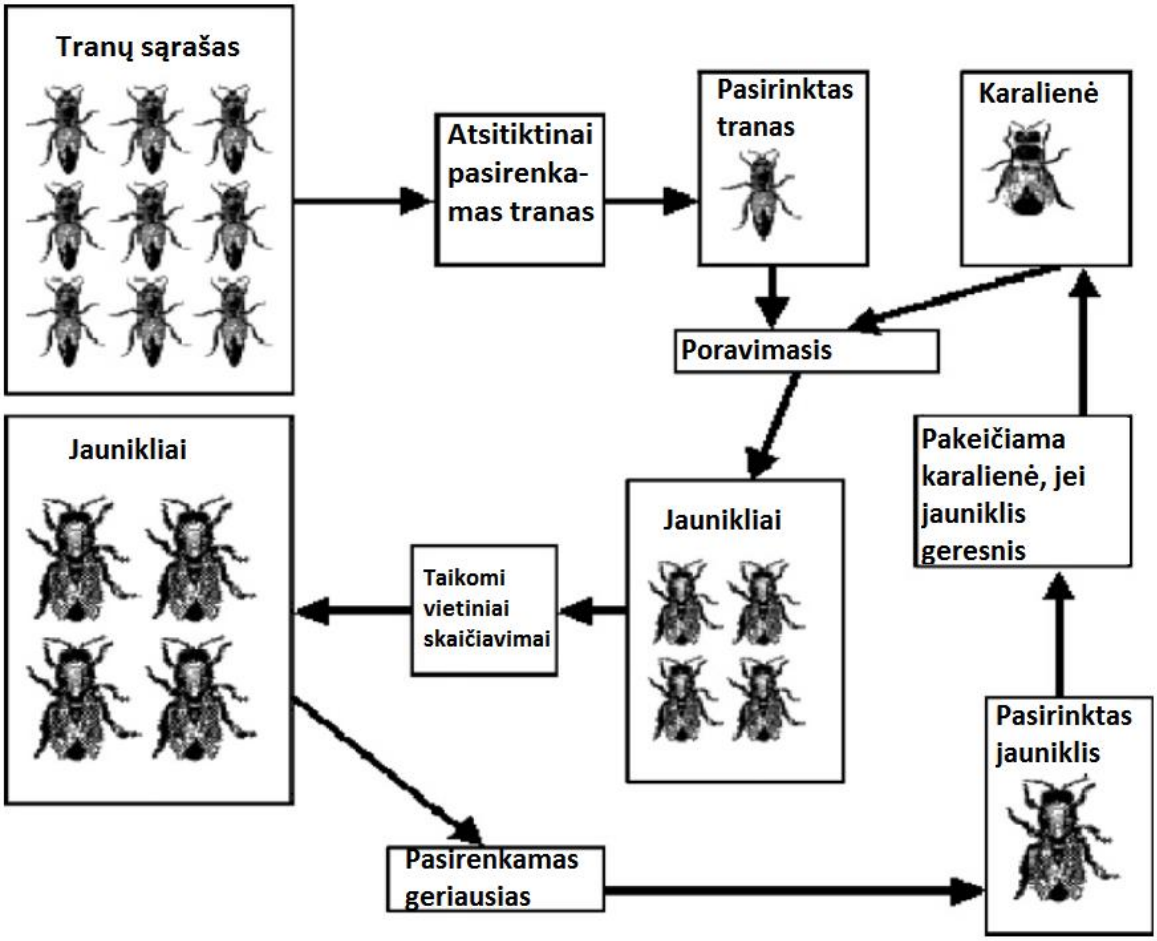
1. Inicializacija: kiekvienai bitei nustatomas tuščias sprendinys;
2. Kiekviena bitė atlieka priekinį pasą:
    a. Įvertina visus galimus kelius.
    b. Pagal įvertinimą pasirenkamas kelias.
3. Visos bitės grįžta į lizdą (atgalinis pasas);
4. Surūšiuojamos bitės pagal jų sprendinio reikšmę;
5. Kiekviena bitė atsitiktinai nusprendžia ar tęsti savo tyrinėjimą ir tapti darbininke ar tapti žiūrovėmis ir tyrinėti kitos bitės pasiektą geresnį sprendinį;
6. Grįžtama į antrą žingsnį, kol neįgyvendinama pabaigimo sąlyga [7],[24].

1.2.4 Bičių poravimosi optimizavimo algoritmas

Bičių poravimosi optimizavimo algoritmas (angl. Honey-bee mating optimization algorithm) yra meta-euristinis algoritmas taikomas: komercija, inžinerija, moksliniai darbai. Šio algoritmo idėja remiasi tikrų bičių poravimosi principu. HBMO algoritmas yra panašus į genetinį algoritmą, nes Tranai su karaliene atlieka kryžminimo funkciją HBMO algoritmas susideda iš šių etapų:

1. Algoritmas prasideda poravimosi skrydžiu, kur karalienė (geriausias sprendinys) ir atsitiktinai pasirinktas tranas (atsitiktinis sprendinys) suporuojami, kad siekiant gauti jauniklių;
2. Sukuriami nauji jaunikliai (sprendiniai, paremti atsitiktinio sprendinio ir geriausio sprendinio), kryžminant trano genus su karalienes (atliekamas kryžminimo algoritmas);
3. Darbininkės atlieka vietinius skaičiavimus su jaunikliais (apskaičiuojamas geriausias sprendinys);
4. Darbininkių bičių tinkamumas pritaikymas, grindžiamas pasiektų jauniklių tobulinimo suma;
5. Pakeičiamas karalienės prastesnis rezultatas geresniu jauniklio rezultatu. [8],[9],[24].

Visas algoritmas pavaizduotas paveikslėlyje (Žr. Pav. Nr. 3).



Pav. 3 HBMO algoritmas. [24].

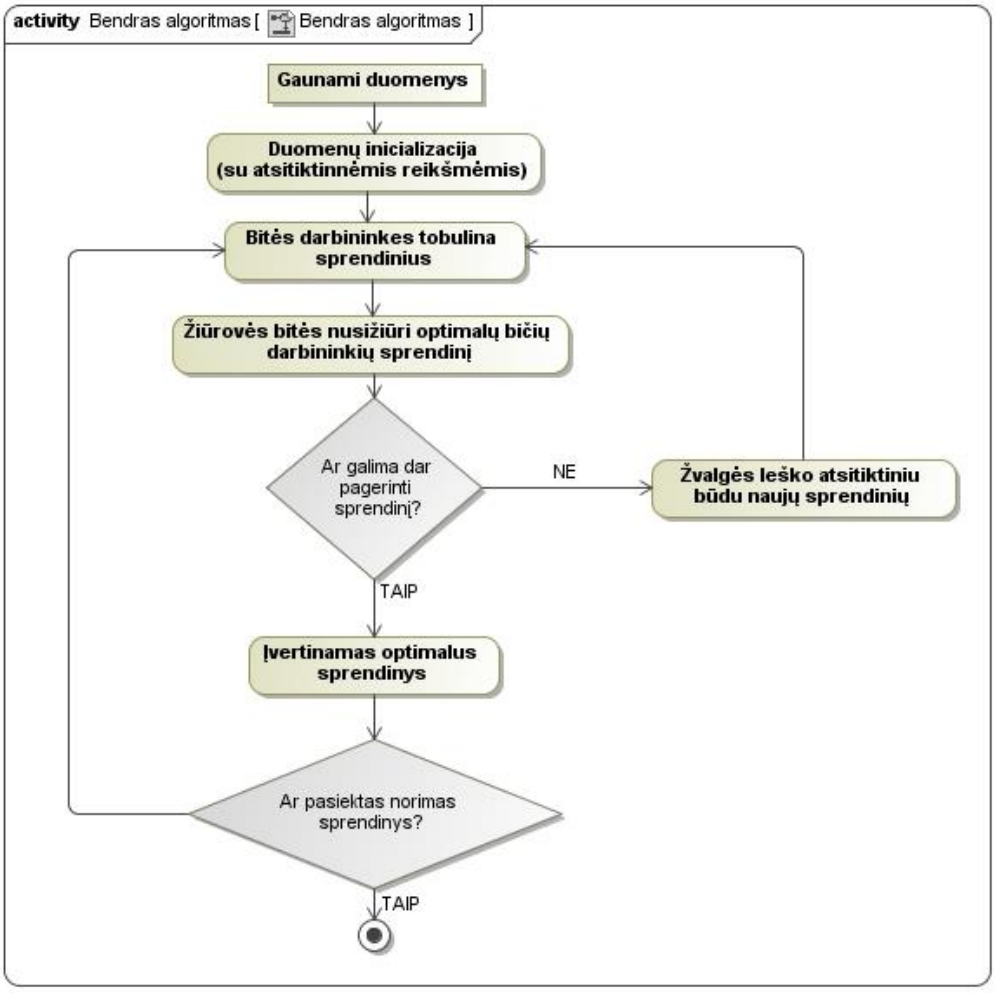
1.2.5 ABC algoritmas

Dirbtinių bičių spiečiaus algoritmas (ABC) yra optimizavimo algoritmas grindžiamas tam tikru manipuliavimu sprendinių rinkiniais, grupėmis. Šis algoritmas priklauso metaeuristninių algoritmams grupei, todėl negarantuoja, kad gautas sprendinys yra optimalus. [4],[24].

Bičių spiečiaus algoritmas sudaro trys bičių grupės: darbininkės bitės (angl. employed bees), žiūrovės (angl. onlookers) ir žvalgės (angl. scouts).

- Darbininkės bitės koreguoja žinomą sprendinį elementariais pakeitimais, stengiantis atrasti naują sprendinį, kuris duotų geresnį rezultatą (Tarpiniui sprendiniui taikomi elementarieji skaičiavimai). [4],[5,[11],[12],[6],[24].
- Žiūrovės bitės laukia darbininkių bičių informacijos, sužinotų maisto šaltinio vietą ir jo kiekį. Išanalizuoja visų šokusių bičių maisto šaltinius ir atsirenka geriausią šaltinį, kuri nusikopijuoja (Tarpiniui sprendiniams priskiriama kita reikšmė). [4],[5,[11],[12],[6],[24].
- Bitės darbininkės, kai nebegali pagerinti kelio prie maisto, gali nuspręsti tapti žvalgėmis. Tada bando ieškoti atsitiktinai naujo kelio prie neatrastų maisto šaltinių.

(Tarpiniui sprendiniui suteikiama atsitiktinė reikšmė). Pagal tai galima sudaryti ABC algoritmą (Žr. Pav.4) [4],[5,[11],[12],[6],[24].



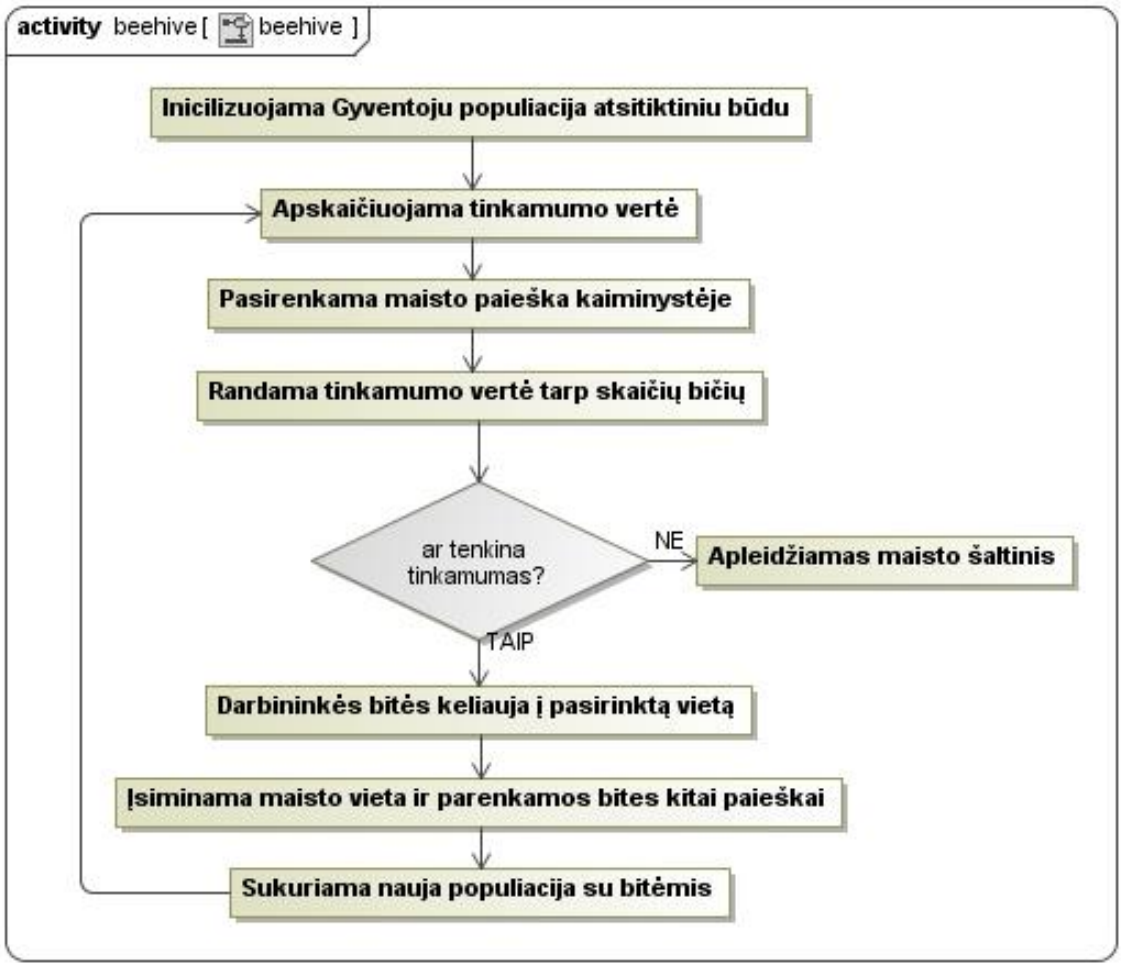
Pav. 4 ABC algoritmas.

1.2.6 Bičių avilys

Bičių avilys (angl. Beehive) yra algoritmas, kuris imituoja maitinimosi elgseną bičių kolonijoje. Šis algoritmas yra taikomas nesuvaržytiems optimizavimo uždaviniams. Bičių avilys algoritmo įgyvendinimas susideda iš šių etapų:

1. Inicializuojama populiacija atsitiktiniai sprendiniais.
2. Apskaičiuojama bičių tinkamumo reikšmės.
3. Pasirenkama maisto vieta kaimynystėje taikant skautes bites.
4. Surandamas tinkamiausia maisto vieta pas skautes bites. Jei tinkamumas netenkinamas, tai maisto šaltinis bus atsisakytas ir darbininkes bitės bus perskirstytos skaučių bičių būriui ieškoti naujų maisto šaltinių.

5. Priešingu atveju darbininkės bites pasirenka maisto šaltinį iš skaučių bičių ir pradeda rinkti maistą.
6. Išsaugoti maisto vieta bičių atmintyje.
7. Kartoti 3-6 žingsnius kol atsiranda stabdymo sąlyga.[13],[14]
   Bičių avilio algoritmas pavaizduotas paveikslėlyje (Žr. Pav. nr. 5).



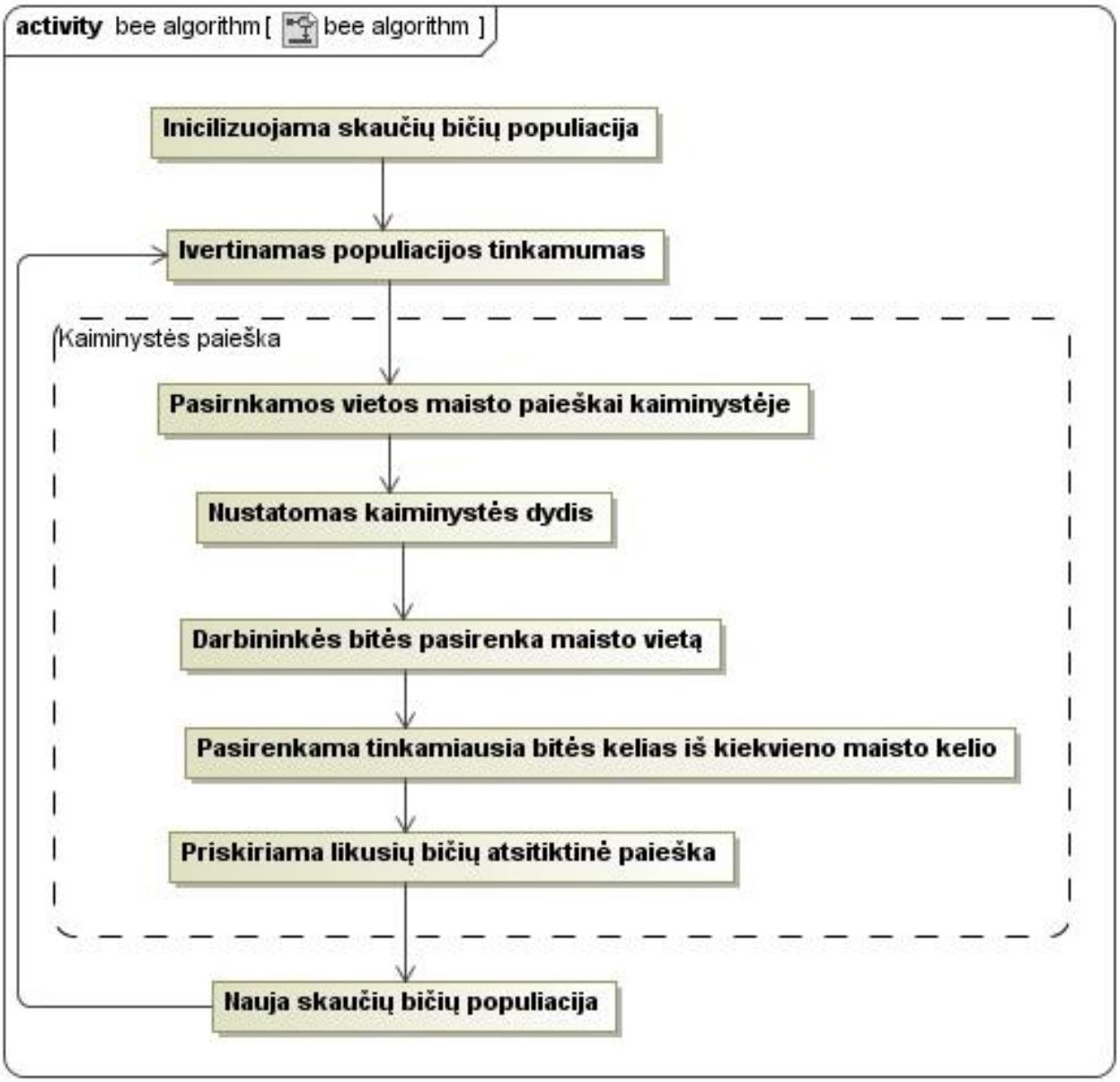
Pav. 5 Bičių avilio algoritmas.

Bičių avilio algoritmas buvo naudojamas per pastarąjį dešimtmetį šiuose srityse:

- Tinklo maršruto parinkimas (angl. Routing in networks).
- „Qos unicas“ maršrutų schema (angl. Qos unicas routing scheme).
- Skaitliuko grėsmių apsaugojimas bičių avylyje. (angl. Counter security threats of beehive).
- Saugumas (angl. Security).
- Maršrutai (angl. routing).
- Interneto paieškos (angl. Web search).[20].

1.2.7 Bičių algoritmas

Bičių algoritmas (angl. Bees Algorithm) yra optimizavimo algoritmas įkvėptas natūralių bičių maitymosi elgsenos. Šiuo algoritmu yra bandoma rasti optimalų uždavinio sprendinį. Paveikslėlyje (Žr. Pav. nr. 6) yra pavaizduotas algoritmo pseudo kodas paprasčiausia forma. [15],[16].



Pav. 6 Bičių algoritmas [16].

Bičių algoritmu yra sukurta daug inžinerinių programų. Pavyzdžiui:

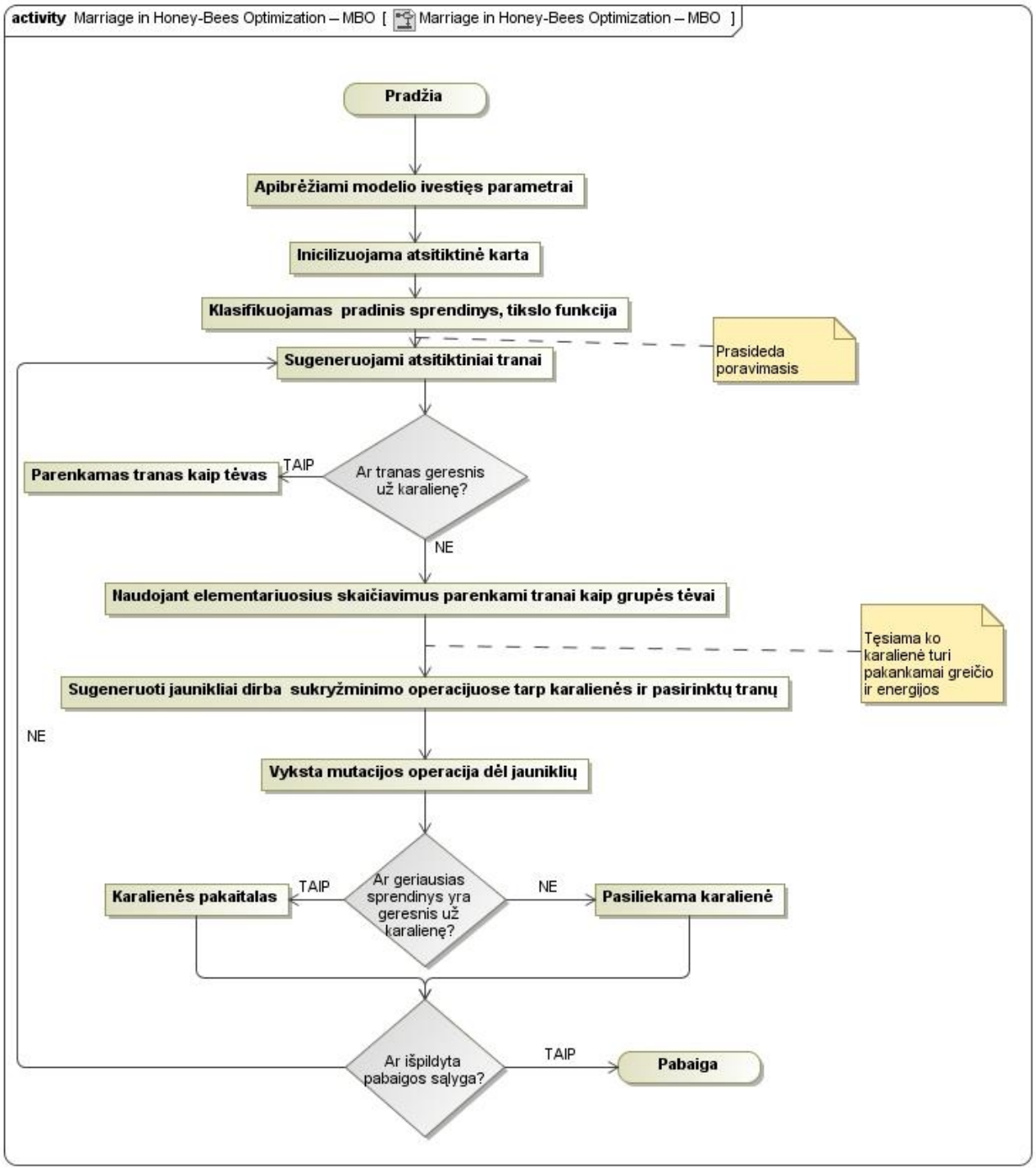
- Mokymo neuroninio tinklo atpažinime.

- Formuojant gaminančias ląsteles.
- Planuojant darbo vietos gamybos mašiną.
- Spręsti inžinerijos optimizavimą.
- Rasti kelis tinkamus sprendimus preliminarią projektavimo problemos.
- Duomenų grupavimas.
- Kelių tikslų optimizavimas.
- Robotikoje.
- Vaizdo analizėje. [16],[20].

1.2.8 Bičių santuokos optimizacija

Bičių santuokos optimizacijoje (angl. Marriage in Honey Bees Optimization) kiekviena populiacijos bitė turi seką užduočių pagrįstų savo genetika, ekologinės aplinkos ir socialinių taisyklių. Karalienė yra svarbiausia avilio narė, nes gamina naujas kartas. Karalienės tarnai su keliais tarnais gyvena keletą dienų, tai yra viena karta. Kiekvienas tranas perduoda savo geną karalienei. Padėdama karalienė naujus kiaušinius panaudoja gautus tranų genus naujiems neapvaisintiems kiaušiniams. Tranai yra kolonijoje tėvai ir kyla iš neapvaisintų kiaušinių. Jų pagrindinis tikslas yra suteikti karalienei savo geną. Kai tai atlieka jie miršta. Darbininkės daugiausiai rūpinasi naujais kiaušinėliais. [17],[18].

Dirbtinis algoritmas prasideda nuo atsitiktinio geno generavimo jaunikliuose, nes karalienė visada yra geriausia turinti sprendinį bitė, iš jauniklių. Po karalienės atrankos prasideda poravimas. Poravimą galima vaizduoti kaip karalienės judėjimą tarp skirtingų tranų. Visa tai yra pavaizduota paveikslėlyje (Žr. Pav. nr. 7). [17],[18].



Pav. 7 Bičių santuokos optimizacijos algoritmas [17].

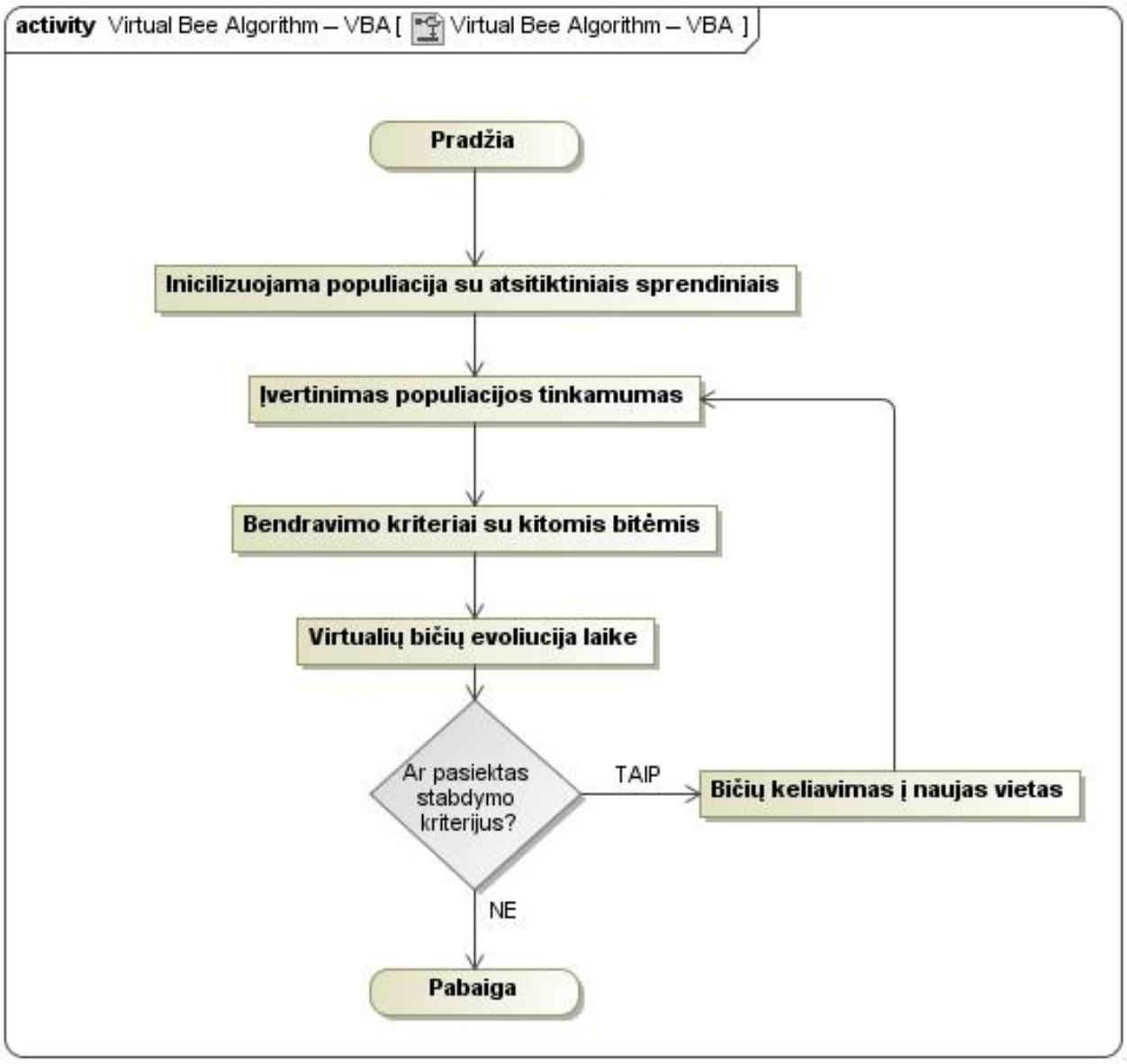
1.2.9 Virtualus bičių algoritmas

Virtualus bičių algoritmas (angl. Virtual Bee Algorithm – VBA) – optimizavimo algoritmas įkvėptas natūraliai besimaitinančių bičių, kad surastų optimalų sprendinį. Gamtos

įkvėpti algoritmai turi tam tikrų panašumų su genetiniais algoritmais, tačiau jie turi kelis agentus, kurie dirba savarankiškai. Todėl VBA yra daug veiksmingesnis nei genetiniai algoritmai, nes yra daug nepriklausomų bičių. Virtualių bičių algoritmo schema prasideda nuo grupės virtualių bičių, kuri kiekviena atsitiktinai ieško maisto šaltinio. Pagrindiniai VBA etapai, kad funkcija optimizuotųsi yra:

1. Sukuriama virtualių bičių populiacija, kur kiekviena bitė turi atskirą atmintį.
2. Išanalizuojami uždavinys ar optimizacijos funkcija ir paverčiama į virtualų maistą.
3. Apibrėžiami bendravimo kriterijai ir atstumai.
4. Atnaujinama arba keliavimas kiekvieno populiacijos individo pozicija į virtualų maistą, maisto žymėjimas ir krypties nustatymas.
5. Po tam tikro laiko ar aplankytu vietų skaičiaus bitės jau turi tiksliausią atsakymą.
6. Iškoduojamas rezultatas į problemos sprendimą. [19].

Visa tai atsispindi paveikslėlyje (Žr. Pav. nr. 8).



Pav. 8 Virtualus bičių algoritmas [19].

1.2.10 Dirbtinių bičių spiečiaus algoritmų domėjimasis Lietuvoje

Lietuvoje bičių spiečiaus algoritmais imtasi domėtis prieš kelis metus. Yra keli darbai apie dirbtinius bičių spiečiaus algoritmus. Vieni iš pirmųjų šia tema paraše mokslinį straipsnį Kauno technologinio universiteto mokslininkai. Jie išnagrinėjo ABC algoritmą ir remiantis algoritmu realizavo programinę įrangą sprendžiančią kvadratinio paskirstymo uždavinį. Atlikus programinės įrangos testavimą buvo pastebėta, jog naudojant dirbtinio bičių spiečiaus algoritmą yra gaunamas didelis efektyvumas. [21].

1.2.11 Dirbtinių bičių spiečiaus algoritmų panaudojimo galimybės

Kognityvinis radijo ryšio tinklas (angl. Cognitive Radio Network) – tai yra pažangi technologija leidžianti radijo prietaisų naudojimosi spektru (t. y. radijo dažniu) į visiškai naujų ir sudėtingesniu būdu. Kognityviniai imtuvai turi galimybę stebėti, fiksuoti ir aptikti jų veiklos aplinką, sąlygas ir dinamiškai iš naujo konfigūruoti savo ypatybes geriausiai atitiktų šias sąlygas. [25].

Viena iš specifinių technikų pritaikomų apsaugoti kognityvinius radijo ryšio tinklus yra naudojami dalelių spiečiaus optimizavimo algoritmas (angl. Particle swarm optimization PSO). Kiekviena kognityvinio radiją tinkle sudaro dalelė, kuri turi savo nuosavą hipoteze apie tai, kas yra geriausias elgesys konkrečioje situacijoje. Savo elgesį jie atrenka ne iš savo asmeninės nuomonės, tačiau tai yra svertinis vidurkis visų hipotezių tinkle. [25].

Vienas iš dalelių spiečiaus optimizavimo pavyzdžių yra pradinio vartotojo pamėgdžiojimo (angl. Primary User Emulation) ataka prieš dinaminės spektro prieigos (angl. Dynamic Spectrum Access) sistemas. Čia kiekvienas įrenginys gali priimti savo hipotezę apie konkrečius perdavimo būdus iš pirminio vartotojo ir daugumos šios grupės narių nuomonė priimti sprendimą. Tikrai bus priimtas sprendimas tenkinantis daugumą, kurį priima prietaisai remiantis hipotezių svertiniu vidurkiu. [25].

Šis požiūris taip pat gali būti naudojamas prisitaikanti radiją. Pavyzdžiui, anksčiau minėtame scenarijuje, naudojamas radijo įvairūs mokymosi algoritmai, siekiant nustatyti, kaip jų indėlį paveikė jų tikslo funkciją. Užpuolikas gali manipuliuoti, kad procesas sukelia pakitusi elgsena radijuje. Jei dalelių spiečiaus optimizavimas buvo naudotas šiame mokymosi etape, radijo bangos būtų patiriamais kas nors panašaus į grupinį mokymusi, o ne individualų mokymuisi, todėl sunkiau įsilaužti į sistemą. [25].

HMBO algoritmas [26] yra lyginamas su PSO metodu. Abu metodai sprendžia vieną uždavinį ir HMBO algoritmas gauna artimesnius optimaliam sprendiniui rezultatus negu PSO. Tai įrodo, kad tam tikri dirbtinių bičių spiečiaus algoritmai gali būti naudojami vietoj PSO metodo.

Taip pat ABC algoritmas yra lyginamas su PSO algoritmu sprendžiant tą patį uždavinį [27],[28]. Gauti rezultatai įrodo, kad ABC algoritmas suteikia geresnius rezultatus negu PSO metodas sprendžiant tą pati uždavinį.

Taigi, galima daryti prielaidą, kad visi dirbtinių bičių spiečiaus algoritmai gali spręsti tuos pačius uždavinius kaip ir PSO metodas ir gauti tokius pat ar net geresnius sprendinius. Būtų galima sukurti dirbtinių bičių spiečiaus realizaciją, kuri spręstų apsaugos klausimą kognityvinio radijo ryšio tinkluose ir palyginti gaunamus rezultatus su PSO metodu.

2. PROJEKTINĖ DALIS

2.1 Įrankių ir priemonių pasirinkimas

Įrankio pasirinkimą nulėmė, tai, jog magistro darbe yra tęsiama bakalauro darbo tema. Kadangi bakalauro darbe programa buvo realizuota JavaScript kalba ir programos realizacijai pakako JavaScript kalbos galimybių, buvo nuspręsta ir magistro darbe tęsti darbus su šia kalba. Ši kalba leidžią programą realizuoti interneto svetainėse ir nereikalauja papildomų plėtinių. Taip pat JavaScript kalbą palaiko visos interneto naršyklės, tad vartotojas galės naudotis programa su bet kuria operacinę sistema. JavaScript kalbos kodas yra vykdomas kliento pusėje. Tai reiškia, kad kodas yra vykdomas vartotojo procesoriuje, o ne svetainės serveryje. Tai sutaupo pralaidumą ir įtampą serveryje. Taipogi, šioje kalboje yra realizuoti masyvai, If sakiniai, For ciklai, funkcijų kūrimas, kurie bus panaudojami įgyvendinant programos realizaciją.

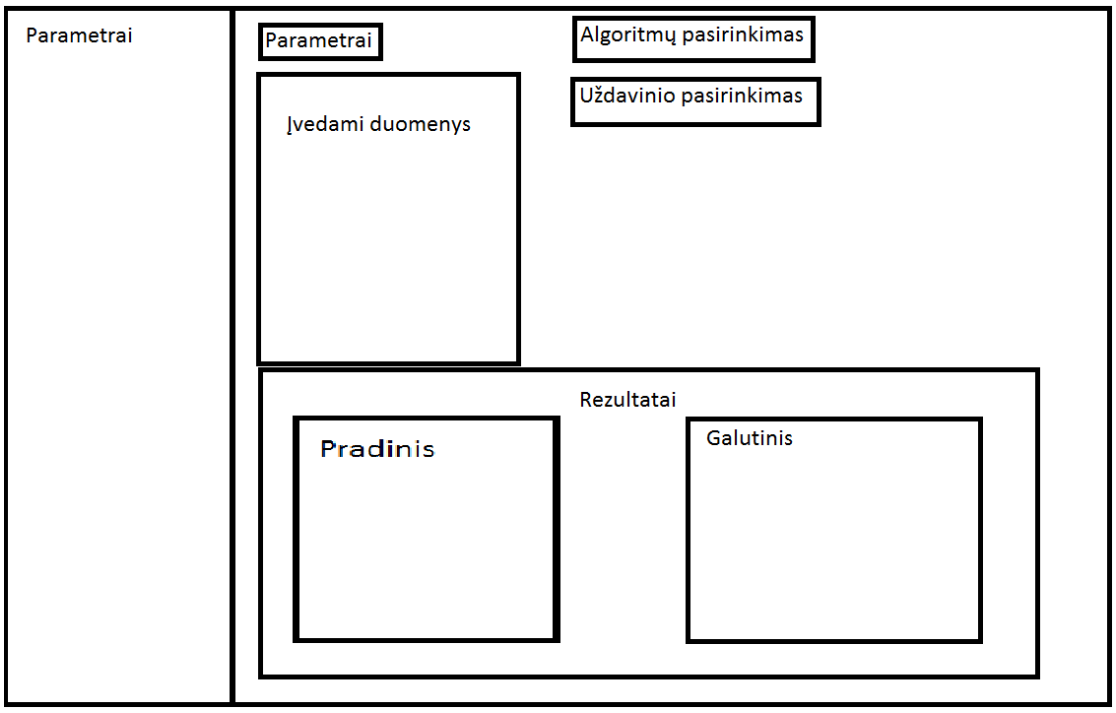
Notepad++ v6.7.7 teksto redaktorių pasirinkau kaip darbo atlikimo priemonę. Pasirinkimą nulėmė tai, jog rašant kodą JavaScript kalba, nėra reikalinga kokia nors programavimo aplinka, kaip pvz. Microsoft Visual Studio 2013. Taip pat Notepad++ išskiria kalbos bazinius žodžius , todėl kodą yra lengviau skaityti ir suprasti.

2.2 Pradinis projektas

2.2.1 Bendras programos projektavimas

Atlikus temos ir darbinės srities modelio analizę, toliau vykdomas projektavimas ir realizavimas.

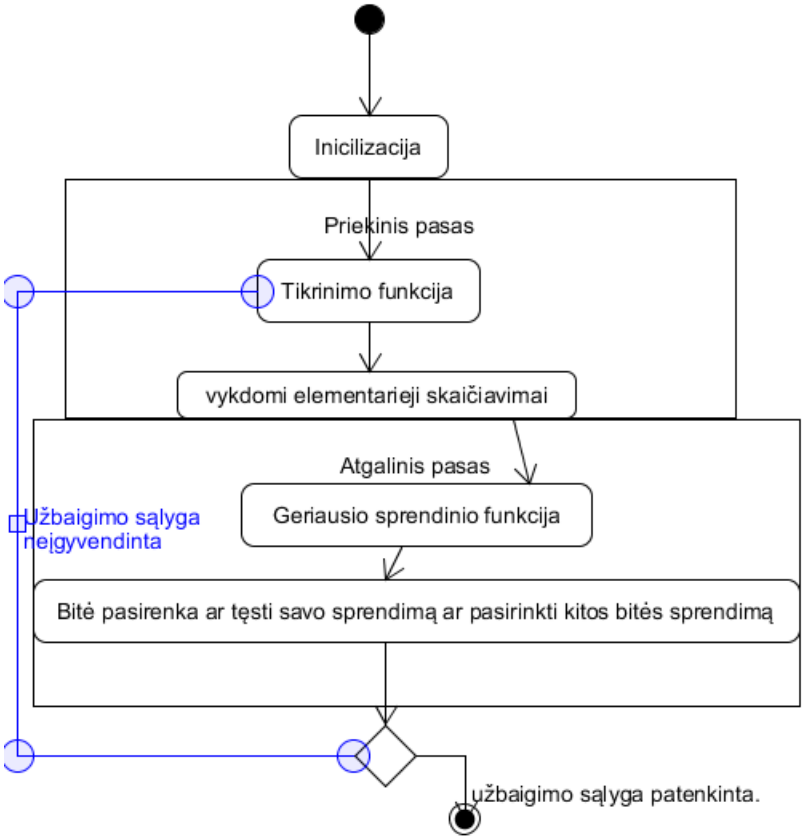
A. Pradiniai duomenys pateikiami iš tekstinio failo, nes bičių spiečiau algoritmai yra skirti dirbti su dideliu kiekiu duomenų;
B. Pradiniai duomenys dirbtinių bičių spiečiaus algoritmams bus sugeneruojami atsitiktiniu būdu.
C. Programinės įrangos gautus sprendinius išvesime į tą patį naršyklės langą, šalia pateiktų duomenų.
D. Sprendiniai bus atvaizduojami tekstu.
E. Vartotojui bus leidžiama modifikuoti parametrus.
F. Grafinės sąsajos modelis (Žr. Pav. nr. 9):



Pav. 9 Grafinės sąsajos modelis.

2.2.2 Bičių kolonijos optimizavimo algoritmas

Elementariuose skaičiavimuose bus naudojama perrikiavimo funkcija. Sudarytas programos algoritmas (Žr. Pav. nr. 10):



Pav. 10 Bičių kolonijos algoritmas.

2.2.3 ABC algoritmas

Atlikti kodo optimizavimą. Realizuotą programą pritaikyti spręsti tvarkaraščių ir perrinkimo uždaviniams.

2.2.4 Likusieji dirbtinių bičių algoritmai

Algoritmams mus naudojami algoritmų prototipai aprašyti 1.2. skyriuje. Algoritmų pabaigimui bus naudojama užbaigimo sąlyga, kuri sustabdys algoritmą po pasirinkto iteracijų skaičiaus. Algoritmuose kur reikalingi elementarieji skaičiavimai bus naudojamas poslinkio arba apkeitimo metodai.

3. REALIZACINĖ DALIS

3.1 Galutinis projektas

Realizuoti algoritmų interpretacijas buvo naudojamas funkcinis programavimas. Tai yra aprašant sprendimo sąlygas naudojamos funkcijos.

3.1.1 Bičių poravimosi optimizavimo algoritmas

Kryžminimo funkcijai įgyvendinti buvo pasirinktas genetinio algoritmo ciklinio kryžminimo metodas.

Ciklinio kryžminimo esminis principas yra tas, jog stengiamasi išsaugoti abiejuose tėvuose esamą informaciją. T.Y. visos palikuonio reikšmės yra gautos iš vieno iš tėvų. Ciklinio kryžminimo etapai:

1. Visos reikšmės rastos tose pačiose pozicijose abiejuose tėvuose, priskiriamos atitinkamoje pozicijoje palikuoniui.
2. Pradedant nuo pirmos pozicijos, elementas atsitiktinai parenkamas vienas iš tėvų. Atliekamas papildomas veiksmas, užtikrinantis, jog neįvyktų atsitiktiniu mutacijų. [29].

Paminėti etapai yra pavaizduoti paveikslėlyje (Žr. Pav. Nr. 11):

3	5	8	2	9	1	4	6	7	Tėvas 1
8	5	6	9	4	1	2	3	7	Tėvas 2
	5				1			7	Žingsnis nr. 1: elementai 5,1,7 pavelėti iš tėvų
3		8					6		Žingsnis nr. 2: pradinė pozicija 3, elementas 8
			9	4		2			Žingsnis nr. 3: pradinė pozicija 4, elementas 9
3	5	8	9	4	1	2	6	7	Palikuonis

Pav. 11 Ciklinis kryžminimas [29].

3.1.2 Bičių santuokos optimizacija

Kryžmino funkcijai įgyvendinti buvo pasirinktas genetinio algoritmo pozicijos pagrindu paremtas kryžminimo metodas. Šio metodo etapai:

A. Atsitiktinai yra pasirenkama pozicijos tėvuose.
B. Imamas atsitiktinai vienas iš tėvų ir į palikuonį perrašomos reikšmės iš atsitiktinai pasirinktų pozicijų.

C. Imamas antras tėvas ir iš jo perrašomi likę reikšmės.[29].

Pavyzdžiui turime du tėvus (Žr. Pav. Nr. 12):

(1 2 3 4 5 6 7 8) ir (2 4 6 8 7 5 3 1),

Pav. 12 Tėvai [29].

Ir tarkime atsitiktinai pasirenkame antrą, trečia, ir šeštą pozicijas. Gauname palikuonis (Žr. Pav. Nr. 13):

(1 4 6 2 3 5 7 8) ir (4 2 3 8 7 6 5 1).

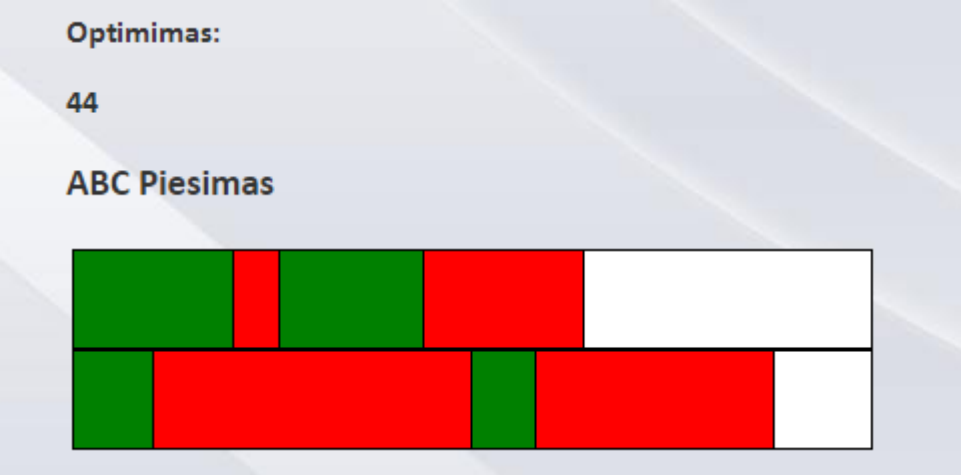
Pav. 13 Palikuonys [29].

3.1.3 Programos grafinė sąsaja

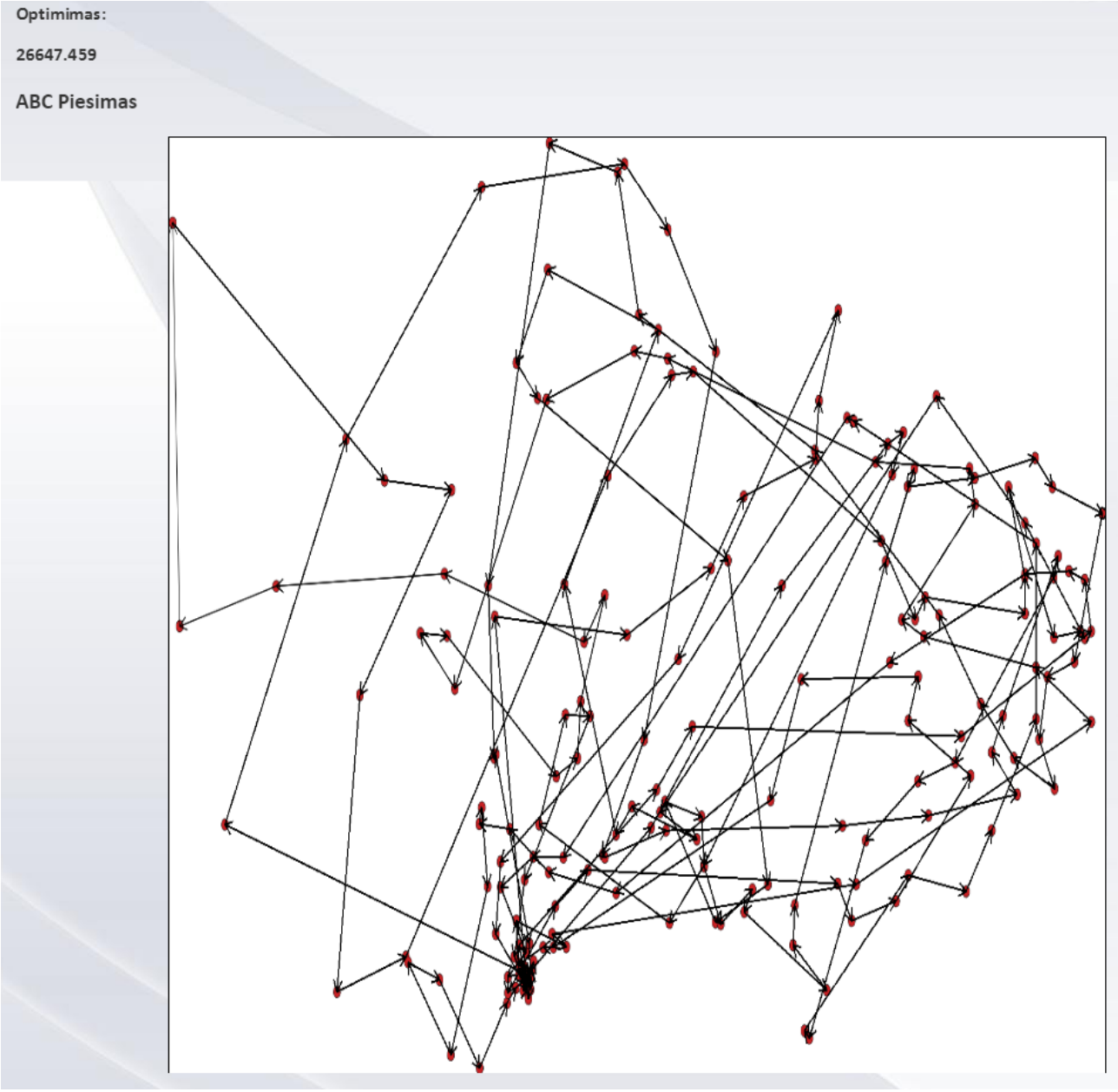
Programos grafinė vartotojo sąsaja pavaizduota (Žr. Pav. Nr. 14):

Pav. 14 Vartotojo grafinė sąsaja.

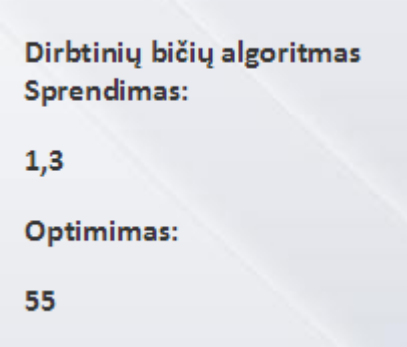
Rezultatų atvaizdavimo pavyzdžius galite matyti paveikslėliuose (Žr. Pav. Nr. 15, 16, 17):



Pav. 15 Tvarkaraščio sudarymo rezultatų atvaizdavimas.



Pav. 16 Keliaujančio pirklio rezultatų atvaizdavimas.



Pav. 17 Kuprinės uždavinio rezultatų atvaizdavimas.

3.2 Problemų sąrašas

Kuriant tvarkaraščių uždavinių realizacija buvo susidurta su problema, kad JavaScript kalba neturi struktūrinių kintamųjų, kaip C#, C++ ir kitos kalbos. Ši problema buvo apeita naudojant masyvus masyvuose. Tačiau taip sudėtingėja duomenų paėmimas iš masyvo ir kodo skaitymas yra sudėtingesnis.

3.3 Darbo rezultatų analizė

Visi testai buvo atlikti kompiuteryje, kuriuo parametrai yra:

Procesorius - Intel® Core™ i7 3610QM Processor;

RAM – 16 GB SDRAM;

Kietasis diskas – 2,5 SATA 500GB 7200;

Vaizdo plokštė - NVIDIA® GeForce® GT 650M with 2GB DDR3 VRAM;

Operacinė sistema: Windows 7.

3.3.1 Keliaujančio pirklio uždavinys

Atlikti keliaujančio pirklio uždavinio rezultatų analizę buvo pasiimti tęstiniai duomenys iš: http://www.math.uwaterloo.ca/tsp/data/ puslapio. Visiems algoritmas buvo testuojamas su įvairiais parametrais, nustatyti parametrus, kurie suteikia geriausius rezultatus. Šiam testui atlikti buvo atlikti po 10 sprendinių su kiekvienu parametru. Paimtas visų rezultatų vidurkis ir išvesti numatytieji parametrai, kurie testavimo metu teikė geriausius rezultatus.

Atliekant eksperimentą su keliaujančio pirklio uždaviniu visiems algoritmas užbaigimo sąlyga buvo taikoma vienodą, tai yra 100 iteracijų. To reikėjo tam, kad galėtume stebėti kiekvieno algoritmo gaunamą rezultatą po tam tikro iteracijų skaičių ir juos palyginti. Atlikus skaičiavimus su keliaujančio pirklio uždaviniu (Žr. diagrama Nr.1) buvo pastebėta, jog MiHBO algoritmas suteikia geriausius rezultatus. Tačiau vidutinis algoritmo atlikimo laikas yra ilgiausias (žr. lentelė Nr. 1). BA algoritmas sugaišta vidutiniškai trumpiausią laiką ir gauna geresnį rezultatą negu algoritmai BCO ar BH algoritmai.

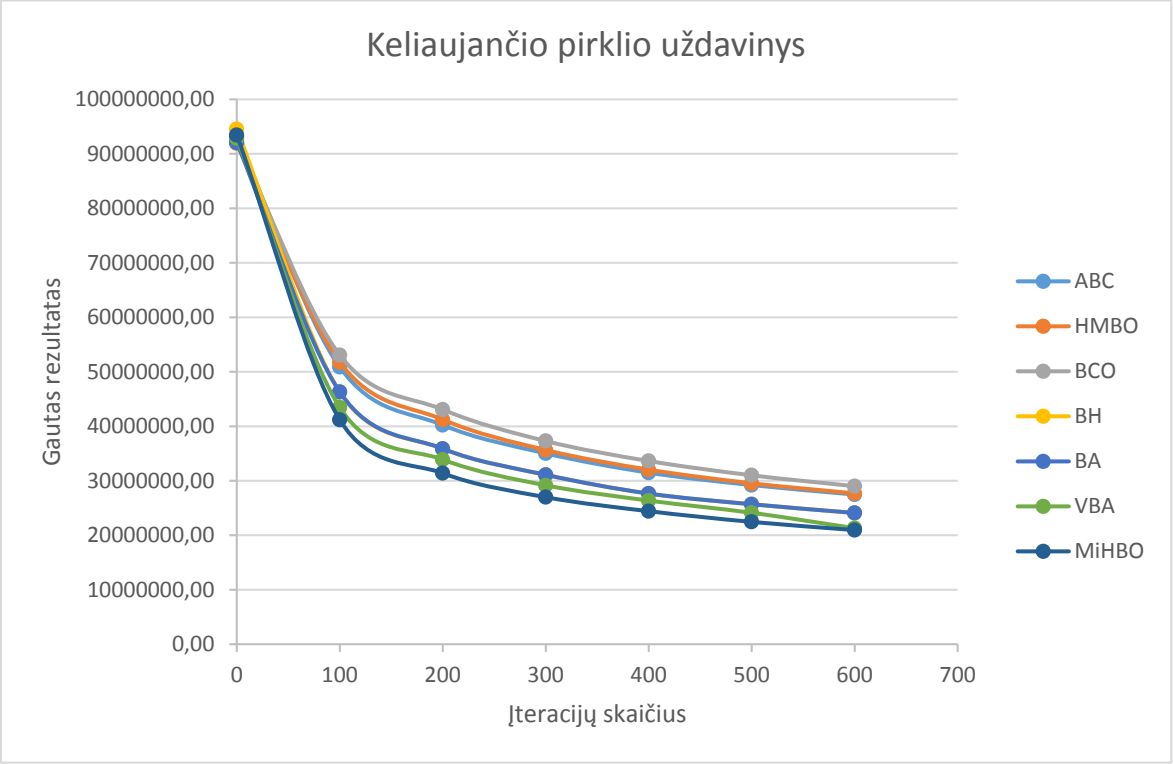


Diagrama. 1 Keliaujančio pirklio uždavinio rezultatų diagrama.

Algoritmas	Vidutiniškai sugaištas laikas (ms)
ABC	471,00
HMBO	482,00
BCO	1644,25
BH	379
BA	1281,5
VBA	1000
MiHBO	2409,40

Lentelė 1 Keliaujančio pirklio vidutiniškai sugaišto laiko lentelė.

3.3.2 Dvejetainės kuprinės uždavinys

Šiam uždaviniui spręsti buvo naudojamas vienas tęstinis uždavinys visiems algoritmams. Kuprinės uždaviniui spręsti visiems algoritmams buvo nustatyta vienoda užbaigimo sąlyga, bei

lygiai toks pats svoris kuprinės. Kuprinės svorio limitas nustatytas 120. Iš diagramos (Žr. diagrama Nr. 2) matome, kad visi algoritmai gauna panašius rezultatus. Diagramos rezultatai kyla beveik tiesine linija. Tai gali būti todėl, kad buvo pasirinktas tęstinis uždavinys turintis mažai duomenų.

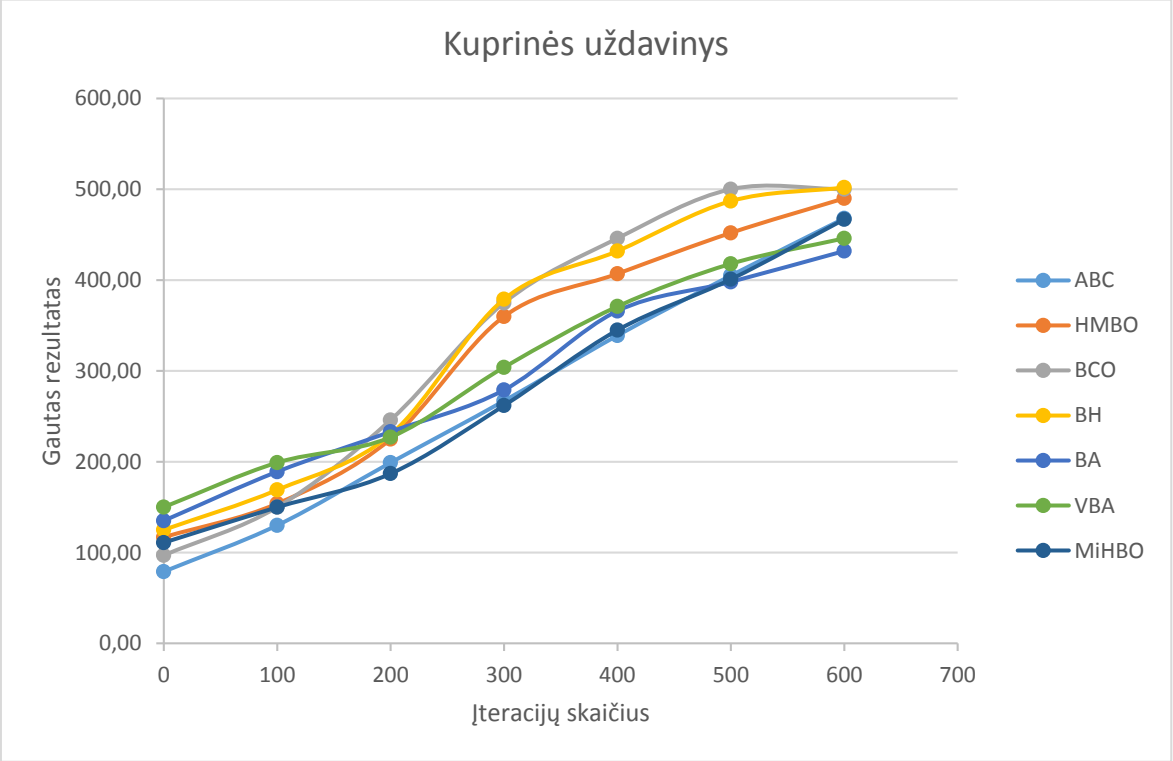


Diagrama. 2 Kuprinės uždavinio rezultatų diagrama.

Algoritmų atlikimo laikas skiriasi nuo keliaujančio pirklio uždavinio daugiau nei keturis kartus. Tai parodo duomenų kiekį. Pagal gautus rezultatus (Žr. lentelė Nr. 2) pastebime, kad visi algoritmai spręsti šį uždavinį užtruko panašų laiko tarpą.

Algoritmas	Vidutiniškai sugaištas laikas (ms)
ABC	87.48
HMBO	95.36
BCO	105.16
BH	146.91
BA	108,5
VBA	98.52
MiHBO	151.7

Lentelė 2 Kuprinės uždavinio sugaištas laikas.

3.3.3 Tvarkaraščio uždavinys

Tvarkaraščio uždavinį eksperimentą atlikome tik su dvejomis mašinomis ir 169 darbais. Visiems algoritmams buvo nustatyta ta pati pabaigos sąlyga. Dėl to, kad šiame algoritme yra naudojami masyve dvimatis masyvas, jam nebuvo pritaikyta uždavinio tęstinumo funkciją. Atlikus eksperimentą ir susisteminus duomenis (Žr. diagrama Nr. 3) pastebime, jog BH algoritmas atlieką

didžiausią pokytį tarp pradinio sprendinio ir gauto rezultato. Tačiau, šis algoritmas užtrunka vidutiniškai ilgiausiai (Žr. lentelė Nr. 3). Trumpiausiai skaičiuoja algoritmas HMBO, tačiau jis tarp pradinio sprendinio ir gauto rezultato padaro mažą pokytį.

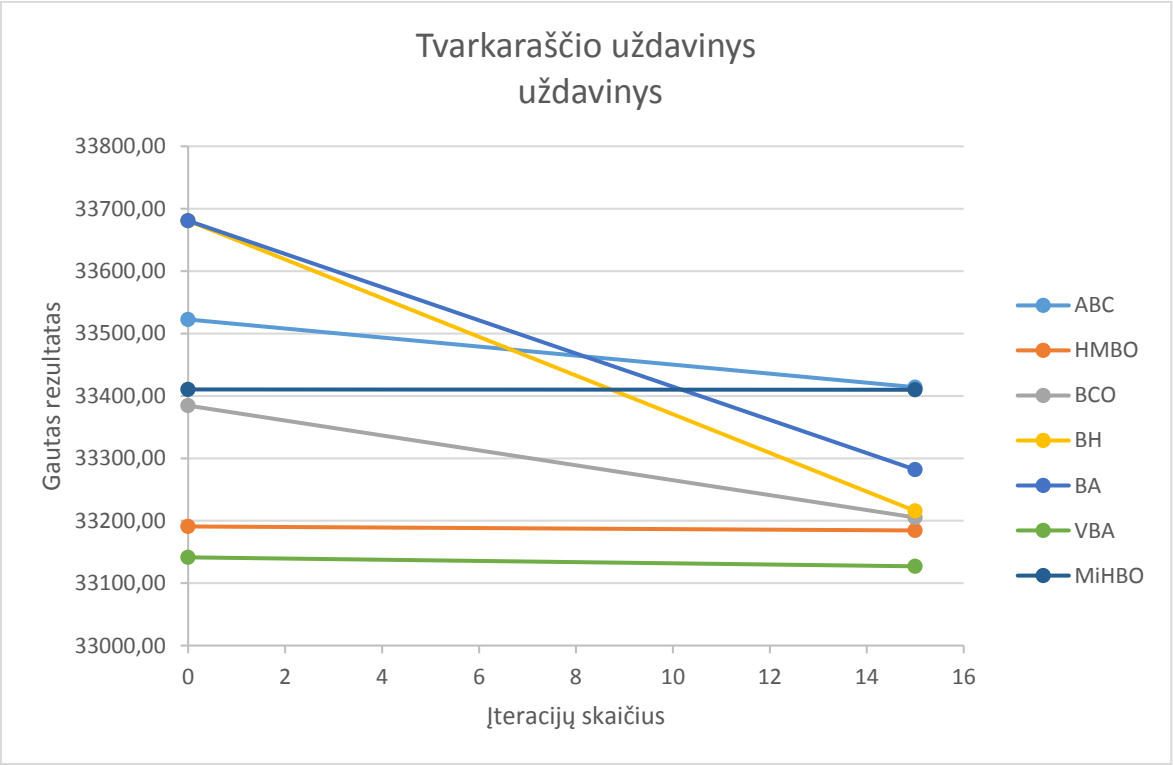


Diagrama. 3 Tvarkaraščio uždavinio rezultatų diagrama.

Algoritmas	Vidutiniškai sugaištas laikas (s)
ABC	2.66
HMBO	1.37
BCO	3.23
BH	3.15
BA	1.67
VBA	1.85
MiHBO	2.05

Lentelė 3 Tvarkaraščio uždavinio sugaištas laikas.

3.4 Rekomendacijos

- Reikia sukurti apsaugas nuo neteisingai įvestų parametrų. Tai užtikrintų teisingų ir korektiškų duomenų gavimą, taipogi, tai apsaugotų programą nuo klaidų, kuriuos įvyksta dėl neteisingų duomenų įvedimo.
- Suderinti vartotojo sąsaja su mobiliais įrenginiais, kad vartotojams būtų patogiau naudotis programa.

- Atlikti testus su MAC operacinę sistemą ir patikrinti ar turi rezultatams veiksnių operacinė sistema.
- Optimizuoti tvarkaraščio uždavinio algoritmus. Kadangi šiam algoritmui realizuoti yra naudojamas masyve dvimatis masyvas, tai sulėtina skaičiavimą ir negalima dirbti su dideliais parametrais.

IŠVADOS

- Išnagrinėjus dalelių spiečių sistemų algoritmus paaiškėjo, jog šie metaeuristiniai algoritmai taikomi įvairiems optimizavimo uždaviniams spręsti.
- Atlikus analizę apie įvairius maršrutų optimizavimo uždaviniais paaiškėjo, jog šie uždaviniai labai dažnai sutinkami praktikoje, su jais susiduria logistikos, turizmo ir įvairių paslaugų kompanijos.
- Bičių spiečiaus algoritmai praktikoje dažniausiai taikomi didelės dimensijos kombinatoriniams optimizavimo uždaviniams spręsti; jie efektyvūs, kai reikia taupyti skaičiavimo išteklius.
- Programai kurti buvo naudojama JavaScript kalba. Šios kalbos pranašumas dirbant su masyvais yra realizuota perkėlimo funkcija. Ji palengvino programos realizaciją ir sumažino kodo eilučių skaičių.
- Atlikus testinius skaičiavimus su kontroliniais duomenimis paaiškėjo, jog programa veikia be klaidų, sugeba rasti sprendinius, artimus optimumui, todėl šią programą galima naudoti ir didesnės dimensijos uždaviniams spręsti.
- Atlikus skaičiavimų rezultatų analizę galima teigti, kad visi algoritmai pagerina visuose uždaviniuose pradinį sprendinį.
- Spręsti keliaujančio pirklio uždavinį geriausią sprendinį gauna MiHBO algoritmas, tačiau tam sugaišta daugiausiai laiko. Jį rekomenduojama naudoti, jei turime didelius laiko išteklius ir reikia gauti kuo tikslesnį rezultatą.
- Kuprinės uždaviniui geriausią atsakymą suteikia BCO algoritmas, kuris laiko atžvilgiu nenusileidžia kitiems algoritmams.
- Tvarkaraščio uždavinio geriausius rezultatus pateikia BH algoritmas, kuris užtrunka vidutiniškai vienodą laiko, kaip ir kiti algoritmai.

LITERATŪROS IR INFORMACINIŲ ŠALTINIŲ SĄRAŠAS

1. Narimantas Listopadskis. *Kombinatorinio optimizavimo uždaviniai ir jų sprendimo algoritmai* .[interaktyvus] [žiūrėta 2013-04-15]. Prieiga internete: http://www.fmf.lt/ft/studiju-programos/taikomoji-matematika/S-18508/straipsnis/Kombinatorinio-optimizavimo-uzdaviniai-ir-ju-sprendimo-algoritmai?name=S-18508&l=5&p=1
2. A. Misevičius, J. Blonskis, V. Bukšnaitis, (2007). *Kombinatorinis optimizavimas ir metaeuristiniai metodai: teoriniai aspektai*. [interaktyvus] [žiūrėta 2013-04-15]. Prieiga internete: http://www.leidykla.eu/fileadmin/Informacijos_mokslai/42-43/213-219.pdf
3. Gražvydas Felinskas,(2007). *Euristinių metodų tyrimas ir taikymas ribotų išteklių tvarkaraščiams optimizuoti*. Daktaro disertacija. P. 58-72. [interaktyvus] [žiūrėta 2013-04-15]. Prieiga internete: http://www.mii.lt/files/mii_dis_07_felinskas.pdf
4. Felix T. S. Chan and Manoj Kumar Tiwari, (2007).*Swarm Intelligence Focus on Ant and Particle Swarm Optimization*. I-tech education and Publishing. Vienna, Austia.
5. Pei-Wei TSai, Jeng-Shyang Pan, Bin-Yih Liao, Shu-Chua Chu,(2009). *Enchanced artificial bee colony optimization*. [interaktyvus] [žiūrėta 2013-04-15]. Prieiga internete: http://iiirc.hitsz.edu.cn/course/CI/Section5/ENHANCED%20ARTIFICIAL%20BEE%20COLONY%20OPTIMIZATION.pdf
6. M. Faiza Abdulsalam, Azuraliza Abu Bakar, (2012). *A Cluster-Based Deviation Detection Task Using the Artificial Bee Colony (ABC) Algorithm*. [interaktyvus] [žiūrėta 2013-04-15]. Prieiga internete: http://docsdrive.com/pdfs/medwelljournals/ijscomp/2012/71-78.pdf
7. Dušan Teodorović. *Bee Colony Optimization (BCO)*. [interaktyvus] [žiūrėta 2013-04-15]. Prieiga internete: http://www.sf.bg.ac.rs/downloads/katedre/oi/1.BCO-Book-Chapter.pdf
8. A. Afshar, O. Bozorg Haddad, M.A. Marino, B.J. Adams (2006). *Honey-bee mating optimization (HBMO) algorithm for optimal reservoir operation*. [interaktyvus] [žiūrėta 2013-04-15]. Prieiga internete: http://robotics.csie.ncku.edu.tw/HCI_Project_2009/徐福瑜2.pdf
9. Hussein A. Abbass, Jason Teo (2001). *A True Annealing Approach to the Marriege in Honey-Bees Optimization Algorithm*. [interaktyvus] [žiūrėta 2013-04-15]. Prieiga internete:

http://pdf.aminer.org/000/905/860/a_true_annealing_approach_to_the_marriage_in_honey_bees.pdf

10. Pei-Wei TSai, Jeng-Shyang Pan, Bin-Yih Liao, Shu-Chua Chu,(2009). *Enchanced artificial bee colony optimization*. [interaktyvus] [žiūrėta 2013-04-15]. Prieiga internete: http://iiirc.hitsz.edu.cn/course/CI/Section5/ENHANCED%20ARTIFICIAL%20BEE%20COLONY%20OPTIMIZATION.pdf
11. Nadezda Stanarevic, Milan Tuba, Nebojsa Bacanin,(2011). *Modified artificial bee colony algorithm for constrained problems optimization*. [interaktyvus] [žiūrėta 2013-04-15]. Prieiga internete: http://www.naun.org/multimedia/NAUN/m3as/20-418.pdf
12. Nishant Pathak, Sudhanshu Prakash Tiwari,(2012). *Traveling Salesman Problem Using Bee Colony With SPV*. [interaktyvus] [žiūrėta 2013-04-15]. Prieiga internete: http://www.ijsce.org/attachments/File/v2i3/C0716052312.pdf
13. B. Padmanabhan1, J. Jasper M. E. and Siva Kumar R. S, (2011). .*Bee Hive Algorithm to Optimize Multi Constrained piecewise Non-Linear Economic Power Dispatch Problem In Thermal Units.* . [interaktyvus] [žiūrėta 2014-01-1]. Prieiga internete: http://ls3-www.cs.uni-dortmund.de/downloads/pdf/WFZ04.pdf
14. Horst F. Wedde, Muddassar Farooq, and Yue Zhang (2004). *BeeHive: An Efficient Fault-Tolerant Routing Algorithm Inspired by Honey Bee Behavior*. [interaktyvus] [žiūrėta 2014-01-1]. Prieiga internete: http://www.ijeei.org/docs-17146881664d9aba2bf1d23.pdf
15. D.T. Pham, A. Ghanbarzadeh, E. Koç, S. Otri , S. Rahim , M. Zaidi (2006). *The Bees Algorithm – A Novel Tool for Complex Optimisation Problems*. [interaktyvus] [žiūrėta 2014-01-1]. Prieiga internete: http://people.stfx.ca/x2010/x2010oxr/3.pdf
16. Fahimeh Aghazadeh and Mohammad Reza Meybodi (2011). *Learning Bees Algorithm For optimization*. [interaktyvus] [žiūrėta 2014-01-1]. Prieiga internete: http://ce.aut.ac.ir/~meybodi/paper/Aghazadeh-IPCSIT%202011-Singapur-2011.pdf
17. Hussein A. Abbass .*MBO: Marriage in Honey Bees Optimization A Haplometrosis Polygynous Swarming Approach*. [interaktyvus] [žiūrėta 2014-01-1]. Prieiga internete: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.14.6119&rep=rep1&type=pdf
18. Hussein A. Abbass and Jason Teo (2001). *A True Annealing Approach to the Marriage in Honey–Bees Optimization Algorithm* .[interaktyvus] [žiūrėta 2014-01-1]. Prieiga internete: http://pdf.aminer.org/000/905/860/a_true_annealing_approach_to_the_marriage_in_honey_bees.pdf

19. Laiq Khan, IkramUllah, Tariq Saeed, K.L. Lo (2010). *Virtual Bees Algorithm Based Design of Damping Control System for TCSC.* [interaktyvus] [žiūrėta 2014-01-1]. Prieiga internete: http://www.ajbasweb.com/ajbas/2010/1-18.pdf
20. Dervis Karaboga, Bahriye Akay (2009). *A survey: algorithms simulating bee swarm intelligence.* [interaktyvus] [žiūrėta 2014-01-1]. Prieiga internete: http://xa.yimg.com/kq/groups/16253916/194111798/name/fulltext.pdf
21. A. Misevičius, J. Blonskis, V. Bukšnaitis, (2011). *Bičių spiečių imitavimas sprendžiant optimizavimo uždavinius*. [interaktyvus] [žiūrėta 2014-01-1]. Prieiga internete: http://www.leidykla.eu/fileadmin/Informacijos_mokslai/2011-56/163-173.pdf
22. David Pisinger (1995). *Algorithms for Knapsack problems*.[interaktyvus] [žiūrėta 2014-01-1]. Prieiga internete: http://www.diku.dk/~pisinger/95-1.pdf
23. Samia kouki, Mohamed Jemni, Talel Ladhari (2011). *Solving the Permutation Flow Shop Problem with Makespan Criterion Using Grids.* [interaktyvus] [žiūrėta 2014-01-1]. Prieiga internete: http://www.sersc.org/journals/IJGDC/vol4_no2/5.pdf
24. Donatas Kavaliauskas, (2013). *Dirbtinės bičių kolonijos algoritmai ir jų taikymai maršrutų optimizavimo uždaviniams spręsti,* [interaktyvus] [žiūrėta 2014-01-1]. Prieiga internete: http://vddb.laba.lt/fedora/get/LT-eLABa-0001:E.02~2013~D_20130729_153102-94516/DS.005.0.01.ETD
25. T. Charles Clancy, Nathan Goergen, (2008). *Security in Cognitive Radio Networks: Threats and Mitigation.* [interaktyvus] [žiūrėta 2014-11-11]. Prieiga internete: http://www.ece.umd.edu/~goergen/docs/cr-crowncom08.pdf
26. Ming-Huwi Horng Ting-Wei Jiang and Jin-Yi Chen, (2009*). Multilevel Minimum Cross Entropy*
Threshold Selection based on Honey Bee Mating Optimization. [interaktyvus] [žiūrėta 2014-11-11]. Prieiga internete: http://www.iaeng.org/publication/IMECS2009/IMECS2009_pp815-818.pdf
27. Anant Baijal, Vikram Singh Chauhan and T Jayabarathi, (2011). *Application of PSO, Artificial Bee Colony and Bacterial Foraging Optimization algorithms to economic load dispatch: An analysis*. [interaktyvus] [žiūrėta 2014-11-11]. Prieiga internete: http://arxiv.org/ftp/arxiv/papers/1111/1111.2988.pdf
28. D. Karaboga, B. Basturk, (2008). *On the performance of artificial bee colony (ABC) algorithm.* [interaktyvus] [žiūrėta 2014-11-11]. Prieiga internete: http://sci2s.ugr.es/eamhco/pdfs/ASC-2008.pdf

29. Kęstutis katkus, (2006).*Hibridinis genetinis algoritmas komivojažieriaus uždaviniui.* [interaktyvus] [žiūrėta 2014-11-11]. Prieiga internete:.http://vddb.laba.lt/fedora/get/LT-eLABa-0001:E.02~2006~D_20060606_203011-23232/DS.005.0.02.ETD

ANOTACIJA

Autorius: Donatas Kavaliauskas

Tema: *Dirbtinės bičių kolonijos algoritmai ir jų taikymų analizė optimizavimo uždaviniams spręsti.*
Šiaulių universitetas 2015.

Šiame darbe yra analizuojami dirbtinių bičių spiečiau algoritmai, optimizavimo uždaviniai ir jų formuluotės, bei praktinės interpretacijos. Yra atliktas tyrimas nustatyti dirbtinių bičių algoritmų pranašumas vienas prieš kitą pagal panaudojimo sritį. Taip pat šiame darbe galima rasti dirbtinių bičių kolonijų algoritmo pritaikymą uždaviniams spręsti, bei sukurtos programos skaičiavimo rezultatų analizę.

SUMMARY

Author: Donatas Kavaliauskas

Subject: *The Analysis of Artificial Bee Colony Algorithms and their Application to the Solving of Optimization Problems.*
Siauliai University 2015

This paper consists of short description of swarm systems algorithms, optimisation problems like traveling salesman, knapsack and flow shop overview and longer description of artificial bee colony algorithms adaptation for these problems solving. Moreover, you can find an artificial bee colony algorithm's application and analysis of computational results.

Priedai

1. CD turinys:

- Kataloge „Magistro aprašymas“ yra Magistro darbo elektroninis aprašo versijos: Donatas_Kavaliauskas.docx ir Donatas_Kavaliauskas.pdf
- Kataloge „Programa“ yra įdėtas index.html failas, kuriame realizuota ABC algoritmas, papildomi failai dėl programos išvaizdos. Be to yra įdėtas tekstinis failai pvz.txt, pvz1.txt ir kt., kuriuose yra tęstiniai duomenys programai.
- Programos vartotojo vadovas, kuris paaiškina , kaip naudotis ABC algoritmo realizacija, yra įdėtas kataloge „Programa“.