VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
FACULTY OF INFORMATION TECHNOLOGY

ÚSTAV INTELIGENTNÍCH SYSTÉMŮ
DEPARTMENT OF INTELLIGENT SYSTEMS

PŘEHLED SOUČASNÝCH PŘÍSTUPŮ K OPTIMALIZACÍM

OVERVIEW OF ACTUAL APPROACHES TO OPTIMIZATION

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE **PATRÍCIA HUDECOVÁ**
AUTHOR

VEDOUCÍ PRÁCE **Doc. Ing. FRANTIŠEK V. ZBOŘIL, CSc.**
SUPERVISOR

BRNO 2020

Vysoké učení technické v Brně
Fakulta informačních technologií

Ústav inteligentních systémů (UITS) Akademický rok 2019/2020

Zadání bakalářské práce

Studentka: **Hudecová Patrícia**
Program: Informační technologie
Název: **Přehled současných přístupů k optimalizacím**
Overview of Actual Approaches to Optimization
Kategorie: Umělá inteligence

Zadání:

1. Prostudujte zadanou literaturu.
2. Vyberte několik různých optimalizačních algoritmů a navrhněte program pro demonstraci jejich činnosti.
3. Navržený program implementujte.
4. Proveďte experimenty s vybranými optimalizačními úlohami.
5. Porovnejte a zhodnoťte výsledky dosažené jednotlivými algoritmy.

Literatura:

- Venter, G.: Review of Optimization Techniques, Encyclopedia of Aerospace Engineering, John Wiley & Sons, Ltd., 2010
- Hassanien, A.E.: Swarm Intelligence, CRC Press, Taylor & Francis Group, 2016

Pro udělení zápočtu za první semestr je požadováno:

- První dva body zadání.

Podrobné závazné pokyny pro vypracování práce viz https://www.fit.vut.cz/study/theses/

Vedoucí práce: **Zbořil František V., doc. Ing., CSc.**
Vedoucí ústavu: Hanáček Petr, doc. Dr. Ing.
Datum zadání: 1. listopadu 2019
Datum odevzdání: 28. května 2020
Datum schválení: 31. října 2019

Abstrakt

Cieľom tejto práce bolo preštudovať niektoré z optimalizačných algoritmov inšpirovaných prírodou a otestovať ich úspešnosť pri hľadaní extrému funkcie na rôznych funkciách. Boli vybrané štyri algoritmi a to netopierí algorimus, algoritmus svätojánskych mušiek, algoritmus opelenia kvetov a algoritmus čiernych dier. Ako testovacie funkcie na hľadanie extrému funkcie boli zvolené Griewankova funkcia, Rastringinova funkcia a Rosenbrockova funkcia. Práca obsahuje popis jednotlivých algoritmov, popis testovacích funkcií a popis daných experimentov a vyhodnotenie úspešnosti daných algoritmov.

Abstract

This work aimed to study some of the optimization algorithms inspired by nature and to test their success in finding the extreme of a function on various functions. Four algorithms were selected, namely the bat algorithm, the firefly algorithm, the flower pollination algorithm, and the black hole algorithm. The Griewank function, the Rastringin function, and the Rosenbrock function were chosen as test functions for finding the extreme of the function. The work contains a description of individual algorithms, a description of test functions and a description of the experiments, and an evaluation of the success of the algorithms.

Kľúčové slová

optimalizačné algoritmy, netopierí algoritmus, algoritmus svätojánskych mušiek, algoritmus opelenia kvetov, algoritmus čiernych dier, hľadanie extrému funkcie, Griewankova funkcia, Rastringinova funkcia, Rosenbrockova funkcia, python

Keywords

optimization algorithms, bat algorithm, firefly algorithm, flower pollination algorithm, black hole algorithm, searching for extreme of a function, Griewank function, Rastringin function, Rosenbrock function, python

Citácia

HUDECOVÁ, Patrícia. *Přehled současných přístupů k optimalizacím.* Brno, 2020. Bakalářská práce. Vysoké učení technické v Brně, Fakulta informačních technologií. Vedoucí práce Doc. Ing. František V. Zbořil, CSc.

Přehled současných přístupů k optimalizacím

Prehlásenie

Prehlasujem, že som túto prácu vypracovala samostatne pod vedením pána Doc. Ing. Františka V. Zbořila CSc a uviedla som všetky literárne zdroje, z ktorých som čerpala.

.....
Patrícia Hudecová
4. júna 2020

Poďakovanie

Touto cestou by som sa chcela poďakovať pánovi Doc.Ing. Františkovi V. Zbořilovi CSc. za jeho čas, trpezlivosť a cenné rady pri vypracovávaní tejto bakalárskej práce. Rada by som tiež poďakovala mojej rodine a priateľom za podporu počas celého môjho štúdia a taktiež mojej spolubývajúcej, za to, že vo mňa verila aj keď ja sama nie.

Obsah

Zoznam použitých skratiek

BA	Bat Algorithm
BBA	Binary Bat Algorithm
BH	Black Holes
CBA	Chaotic Bat Algorithm
CF	Constant Frequency
ELD	Economic Load Dispatch
FFA	Firefly algorithm
FM	Frequency Modulation
FPA	Flower Pollination Algorithm
HS	Harmony Search
PSO	Particle Swarm Optimization
SI	Swarm Intelligence

Kapitola 1

Úvod

Cieľom tejto bakalárskej práce bolo zvoliť si niekoľko optimalizačných algoritmov inšpirovaných prírodou, naštudovať si ich problematiku, následne navrhnúť program pre demonštrovanie činnosti zvolených algoritmov a potom na ňom vykonať experimenty a zhodnotiť výsledky.
Na začiatku boli vybrané štyri optimalizačné algoritmy z troch rôznych kategórií. Netopierí algoritmus a algoritmus svätojánskych mušiek z kategórie algoritmov založených na inteligencii skupiny, algoritmus opelenia kvetín z algoritmov inšpirovaných biológiou a algoritmus čiernych dier z algoritmov inšpirovaných fyzikálnymi a chemickými javmi. Ich podrobný popis je uvedený v druhej kapitole spolu s pseudokódmi jednotlivých algoritmov, ktoré sa neskôr stali inšpiráciou pri implementácii programu.
Tretia kapitola obsahuje popis problému hľadania extrému funkcie a popis testovacích funkcií na ktorých boli zvolené algoritmy testované.
V štvrtej kapitole je popísaná implementácia navrhnutého programu, programovací jazyk a knižnice, ktoré boli pri tvorbe programu využité.
Piata kapitola popisuje návrh, priebeh, výsledky a zhrnutie experimentov, ktoré boli vykonané pomocou jednotlivých funkcií, konkrétne Griewankovej, Rastringinovej a Rosenbrockovej testovacej funkcie na vyššie spomínaných algoritmoch.
V závere práce sú zhrnuté výsledky testovania jednotlivých algoritmov a vyhodnotenie ich úspešnosti.

Kapitola 2

Optimalizačné algoritmy inšpirované prírodou

V dnešnej dobe sa optimalizácie využívajú takmer v každej oblasti, či už sa jedná o priemyselnú výrobu, oblasť finančníctva, dopravu, rôzne rozhodovacie systémy a podobne. Existujú rôzne dôvody, pre ktoré sa o optimalizácie usilujeme, či už sa jedná o zníženie nákladov, energie, zvýšenie výkonu, ziskov, efektivity práce a mnoho iných.

Pričom pod pojmom optimalizácia si predstavujeme snahu nájsť ideálne riešenie. Ideálne riešenie nemusí byť vždy najlepšie možné, vzhľadom na časovú a priestorovú náročnosť niektorých optimalizačných algoritmov je pre nás vhodnejšie uspokojiť sa s riešením, ktoré sa najlepšiemu približuje. Aby mala optimalizácia žiadúci význam je veľmi dôležité, si vhodne zvoliť optimalizačný algoritmus.

Vzhľadom na široké využitie optimalizácie, existuje mnoho optimalizačných algoritmov, ktoré je možné deliť do rôznych kategórií podľa rôznych parametrov, podľa podstaty algoritmov ich môžeme deliť na stochastické a deterministické.

Deterministické algoritmy striktne dodržiavajú vopred dané pravidlá, keď si na začiatku zvolia cestu ktorou sa budú uberať v priebehu riešenia ju nezmenia, to znamená, že pri rovnakom vstupe vždy nájdu rovnaký výsledok ich predstavitelom je napríklad horolezecký algoritmus (angl. hill climbing).

Na rozdiel od nich stochastické algoritmy vždy pracujú s istou dávkou náhodnosti, medzi ne patrí napríklad optimalizácia rojom častíc (angl. particle swarm optimization).

Taktiež je možné algoritmy deliť podľa zdroja inšpirácie na algoritmy inšpirované prírodou, algoritmy inšpirované biológiou a meta-heuristické algoritmy.

Meta-heuristické algoritmy je možné považovať za algoritmy vyšších metód, ktoré pri hľadaní riešenia využívajú "výberový" mechanizmus a zdieľanie informácií, medzi ne môžeme zaradiť napríklad metódu náhodného vyhľadávania (angl. random search).

Algoritmy inšpirované prírodou čerpajú inšpiráciu z prírody môžeme medzi ne zaradiť napríklad genetické algoritmy (angl. genetic algorithms), algoritmy inšpirované biológiou taktiež čerpajú inšpiráciu z prírody ale skôr sú špecializované na biologické systémy ako napríklad včelia kolónia, preto ich môžeme považovať za podkategóriu algoritmov inšpirovaných prírodou. Algoritmy inšpirované prírodou môžu byť zároveň aj meta-heuristickými algoritmami.

Veľká časť algoritmov v dnešnej dobe berie inšpiráciu z prírody a to na základe pozorovania správania zvierat, ľudí či rôznych biologických, chemických a fyzikálnych javov ktoré sa odohrávajú v našej prírode, prípadne algoritmy inšpirované hudbou. Tieto algoritmy sú delené do štyroch základných kategórií a to na algoritmy inšpirované inteligenciou skupiny (angl. swarm intelligence) ako napríklad netopierí algoritmus (angl. bat algorithm), algoritmus vlčej svorky (angl. grey wolf optimization) atď., algoritmy inšpirované biológiou, ale nie inteligenciou skupiny, sem zaraďujeme algoritmus párenia včiel (angl. marriage in honey bees), opeľovania kvetov (angl. flower pollination algorithm) a mnohé iné, algoritmy inšpirované fyzikálnymi alebo chemickými javmi ako napríklad algoritmus čiernych dier (angl. black hole), simulovaného žíhania a podobne a ostatné algoritmy. Prvým trom kategóriám sa budem venovať podrobnejšie v nasledujúcich kapitolách.

Do kategórie ostatných algoritmov zaraďujeme algoritmy, ktoré na základe ich vlastností nie je možné zaradiť medzi zvyšné tri skupiny, patrí sem algoritmus ligy majstrov (Championship league algorithm), sociálno-emočný optimalizačný algoritmus (angl. social emotional optimization), algoritmus gramatickej evolúcie (angl. grammatical evolution) a podobne, táto kategória algoritmov je najmenej rozšírená. [16]

2.1 Algoritmy založené na inteligencii skupiny

Inteligencia skupiny (angl. swarm intelligence) sa zameriava na kolektív jedincov, pričom sleduje správanie viacerých jedincov, ktorý spolu interagujú na základe istých pravidiel. Každý jedinec (agent) je sám považovaný za neinteligentného, pričom skupina viacerých jedincov ukazuje známky samo-organizovaného správania, čo pôsobí ako kolektívna inteligencia. Hlavné vlastnosti algoritmov inšpirovaných inteligenciou skupiny je možné zhrnúť nasledovne:

- Zdieľanie informácií medzi viacerými agentmi
- Agenti sú samo-organizovaný a spoločne sa učia
- Je veľmi efektívny pre ich schopnosť vzájomného učenia
- Môže byť jednoducho paralelizovaný pre praktické problémy a problémy v reálnom čase (real-time problémy) [16]

Patria sem algoritmy ako napríklad netopierí algoritmus, algoritmus mačacieho hejna (angl. cat swarm), algoritmus svätojánskych mušiek (angl. firefly algorithm) a mnohé iné.

2.1.1 Netopierí algoritmus

Netopiere môžeme považovať za veľmi zaujímavé živočíchy a to z viacerých dôvodov. Sú to jediné cicavce, ktoré vedia lietať. Taktiež sú to spoločenské živočíchy, ktoré žijú vo väčších skupinách pričom samice a samci môžu žiť oddelene. Ich ďalšou zaujímavou vlastnosťou je schopnosť hibernácie (zimného spánku), kedy ich telesné funkcie klesnú na minimum, čo veľakrát pôsobí akoby ani nežili, celú zimu prežijú na jednom mieste, kde v stave hibernácie žijú zo zásob ktoré získali v priebehu roka. Ďalšou ich zaujímavou vlastnosťou je schopnosť echolokácie, ktorá sa stala inšpiráciou pre BA a bude vysvetlená podrobnejšie v ďalšej časti.

Zároveň sú druhou najväčšou skupinou cicavcov hneď po hlodavcoch, pričom poznáme niečo medzi 900 až 1200 druhmi netopierov. Samotné netopiere tvoria asi pätinu zo všetkých známych druhov cicavcov a vyskytujú sa vo všetkých oblastiach sveta okrem Antarktídy a Arktídy. Ich ďalšou zaujímavou vlastnosťou je, že väčšina z nich sú nočné živočíchy, ktoré celý deň prespia niekde v tme a po zotmení vyrážajú na lov. [3] Netopiere je možné deliť do rôznych kategórií, pre účely tejto práce ich budeme deliť na mikronetopiere (angl. microbats) a meganetopiere (angl. megabats).

Mikronetopiere sú netopiere malých rozmerov, ktoré sa živia vo väčšine prípadov hmyzom, v niektorých prípadoch krvou vtákov, rýb a žiab. Aj napriek mnohým poverám netopiere pre ľudí nie sú nebezpečné, naopak na koľko lovia hmyz sú pre nás užitočné a ich zánik by mal pre nás vážne následky. Existuje len veľmi málo zaznamenaných prípadov, kedy netopier zaútočil na človeka okrem prípadov kedy sa cítil ohrozený a bránil sa. Keďže sa tieto netopiere živia malými živočíchmi a lovia hlavne v noci majú schopnosť echolokácie, ktorú využívajú pri detekovaní a love koristi. Schopnosť echolokácie z nich robí taktiež lepších letcov ako sú vtáky. Narozdiel od meganetopierov majú malé oči a veľké uši, pričom ich uši obsahujú tragus[1]. [12]

Meganetopiere sú netopiere väčších rozmerov ich rozmedzie krídel môže nadobudnúť až 1.7 metra, tieto netopiere sa živia výhradne rastlinou stravou a to väčšinou ovocím prípadne nektárom. Taktiež majú na rozdiel od mikronetopierov dobrý zrak a čuch pomocou ktorého vyhľadávajú potravu, v dôsledku čoho u nich nie je vyvinutá echolokácia. [11]

Echolokácia používaná netopiermi

Ako už bolo spomínané echolokáciu využívajú až na pár výnimiek len mikronetopiere a to nielen pri hľadaní potravy ale taktiež aj na zlepšenie kvality letu.

Echolokácia je jav pri ktorom netopiere zisťujú polohu, vzdialenosť a veľkosť jednotlivých objektov, funguje na princípe odrážania sa zvukov od daných objektov. Netopier vydá zvuk určitej vlnovej dĺžky, pričom vlnová dĺžka závisí od druhu potravy, ktorou sa živí. Tento zvuk sa od daného predmetu odrazí a vráti sa späť k netopierovi, ktorý si vie vďaka nemu určiť veľkosť, vzdialenosť a polohu daného objektu. Zvukový signál obvykle vydáva ústami alebo nosom niektoré netopiere tento zvukový signál vysielajú mávnutím krídel. Spätný signál je prijímaný ušami, pričom na základe jeho intenzity, rozdielu času medzi prijatím signálu do oboch uší získame informácie o horizontálnom uhle objektu od ktorého sa daný signál odrazil, čo môžeme vidieť na obrázku 2.1. Informácie o vertikálnom uhle objektu sú získavané pomocou tragusu.[6]

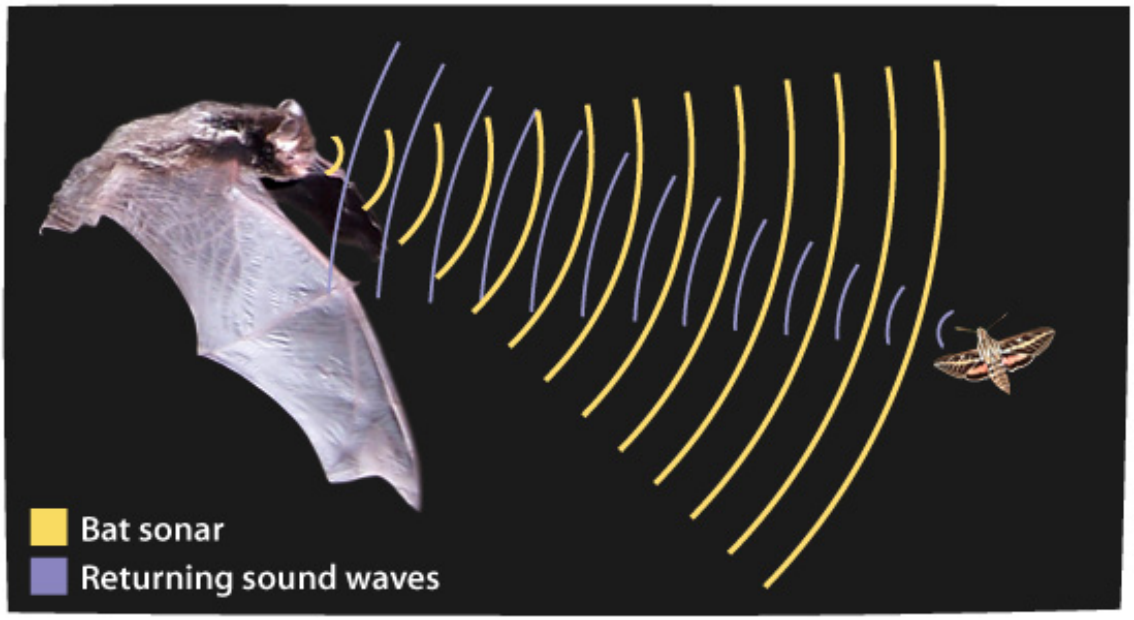
Odrazený signál sa nazýva ozvena (angl. echo). Frekvencia signálu a čas v akom odoslanie signálu vysiela závisí od prostredia v ktorom netopier žije, potravy akou sa živí a účelu za akým vlny vysiela. Je rozdiel v intenzite vĺn pokiaľ netopier zháňa potravu alebo keď sa potrebuje len informovať o prekážkach v teréne. Pričom väčšina signálov ktoré netopiere vydávajú je v rozsahu ultrazvukových vĺn, ktoré sú pre nás nepočuteľné. Ultrazvukové vlny sú vlny s frekvenciou vyššou ako 20 kHz. Mikronetopiere môžeme ďalej rozdeliť na netopiere ktoré pri echolokácii používajú frekvenčnú moduláciu (angl. frequency modulation FM) a tie ktoré používajú konštantnú frekvenciu (angl. constant frequency CF). Ktorý druh vĺn

[1] Tragus je súčasťou vonkajšieho ucha, ktorá je využívaná pri echolokácii.

netopiere využívajú taktiež závisí od typu potravy, pričom obe majú svoje výhody.

Hlavnou výhodou FM vĺn je veľmi presné určenie cieľa, netopiere využívajúce tento typ signálu sú schopné rozlíšiť od seba objekty vzdialené menej ako milimeter od seba, pretože FM signál umožňuje lepšie rozlíšenie času medzi volaním a vracajúcou sa ozvenou, na druhej strane vzdialenosť pri ktorej je možné detekovať ciele je obmedzená, pretože dané ozveny je možné vyhodnotiť iba na veľmi krátky čas nakoľko frekvencia sa môže meniť.

Výhodou netopierov využívajúcich CF signál je možnosť určiť rýchlosť objektu a rýchlosť kmitania krídel obete na základe Dopplerovho efektu [2], čo im umožňuje rozlíšiť lietajúci objekt od stacionárneho. Keďže signál je sústredený do úzkeho frekvenčného pásma, je možné ciele detekovať aj na väčšie vzdialenosti. [2]



Obr. 2.1: Znázornenie echolokácie [28]

Algoritmus

Nakoľko využitie echolokácie netopiermi je rôzne a bolo by komplikované zohľadniť všetky vlastnosti, algoritmus pracuje s tromi idealizovanými pravidlami a to:

1. Všetky netopiere využívajú echolokáciu na vnímanie vzdialeností a zároveň vedia nejakým spôsobom určiť rozdiel medzi jedlom, obeťou a prekážkami v pozadí.

2. Netopiere lietajú náhodne s rýchlosťou v_i na pozícii x_i so stálou frekvenciou f_{min} premenlivou vlnovou dĺžkou λ a hlasitosťou A_0. Môžu automaticky zmeniť vlnovú dĺžku alebo frekvenciu svojich vysielaných impulzov (angl. emitted pulses) a upraviť rýchlosť vysielania impulzov v závislosti na vzdialenosti cieľa. Vyžarovanie impulzov označujeme $r \epsilon [0, 1]$.

[2]Dopplerov jav je zmena vlnovej dĺžky, frekvencie elektromagnetických alebo akustických vĺn vyvolaná pohybom zdroja a pozorovateľa.[5]

3. Aj napriek tomu, že hlasitosť sa môže meniť rôznymi spôsobmi, predpokladáme, že sa mení z veľkej kladnej hodnoty A_0 až po minimálnu konštantnú hodnotu A_{min}.

Ďalej si musíme definovať pravidlá podľa ktorých budeme určovať zmenu ich polohy x_i, rýchlosti v_i v d-dimenzionálnom poli v čase t a to na základe nasledujúcich vzťahov:

$$f_i = f_{min} + \beta(f_{max} - f_{min}) \tag{2.1}$$

$$v_i(t) = v_i(t-1) + f_i(x_i(t) - x_*) \tag{2.2}$$

$$x_i(t) = x_i(t) + v_i(t) \tag{2.3}$$

, kde β je náhodný vektor získaný z rovnomerného rozdelenia. Hodnota x_* je aktuálna najlepšia hodnota polohy získaná porovnaním všetkých riešení spomedzi všetkých n netopierov. Hodnoty f_{min} a f_{max} predstavujú minimálnu a maximálnu frekvenciu potrebnú v závislosti na probléme a premennou v_i vyjadrujeme rýchlosť.

Akonáhľe spomedzi všetkých riešení zvolíme najlepšie, pre oblasť lokálneho vyhľadávania je nové riešenie generované náhodne pre každého netopiera pomocou náhodnej prechádzky[3] (angl. random walk) a to nasledovne:

$$x_{old}(t) = x_{new} + \varepsilon A(t) \tag{2.4}$$

,kde ε je náhodné číslo z intervalu $[0, 1]$ a $A(t)$ je priemerná hlasitosť všetkých netopierov v danom čase.

Ďalej je potrebné aktualizovať hlasitosť A_i a rýchlosť vysielania impulzov r_i v priebehu iterácií. Hlasitosť klesá keď netopier nájde obeť, naopak rýchlosť vysielacej frekvencie s nájdením obete stúpa. Pričom hodnota hlasitosti A_0 môže byť ľubovoľná hodnota, ktorá nám vyhovuje a vypočítame ju pomocou nasledujúceho vzťahu:

$$A_i(t+1) = \alpha A(t) \tag{2.5}$$

Zmenu vysielania impulzov r_i vypočítame nasledovným spôsobom:

$$r_i(t+1) = r_i(0)[1 - exp(-\gamma t)] \tag{2.6}$$

, kde α a γ sú konštanty. [26]

[3] Náhodná prechádzka je stochastický náhodný proces, ktorý popisuje cestu, ktorá pozostáva z postupnosti náhodných krokov v niektorom matematickom priestore. [14]

Pseudokód [26]

Algoritmus je možné napísať pomocou nasledovného pseudokódu:
input: Q frekvenčné pásmo
f_i frekvencia impulzov
r_i rýchlosť vysielania impulzov
A_i hlasitosť
ε kontrolný parameter pre zvukovú amplitúdu
γ kontrolný parameter pre rýchlosť vysielania impulzov
output: Optimálna pozícia netopiera a odpovedajúca fitness funkcia;
Inicializácia populácie n netopierov na náhodných pozíciách a nájdenie najlepšieho riešenia na základe fitness funkcie g∗;
while Nie sú splnené koncové kritéria **do**
foreach netopiera i **do**
Generujeme nové riešenia na základe nastavenia novej frekvencie x_i^{new}, ktoré získame z rovníc 2.1,2.2 a 2.3 a nastavíme nové hodnoty pozícií a frekvencií;
if náhodná hodnota $> r_i$ **then**
Vyberieme spomedzi všetkých riešení najlepšie riešenie;
Generujeme nové lokálne riešenia okolo najlepšieho riešenia;
end
if $rand < A_i$ a zároveň fitness funkcia nového riešenia je lepšia ako pôvodná **then**
Aktualizujeme polohu $netopiera_i$ na novú pozíciu x_i^{new};
Zvýšime rýchlosť vysielania impulzov r_i;
Znížime hlasitosť A_i;
end
Aktualizujeme najlepšie riešenie;
end

Varianty

Daný algoritmus sa vyskytuje v rôznych variantách upravených podľa potreby pre riešenia rôznych úloh ako napríklad.

- Diskrétny netopierí algoritmus (angl. discrete bat algorithm)
- Binárny netopierí algoritmus (angl. binary bat algorithm BBA)
- Chaotický netopierí algoritmus (angl. chaotic bat algorithm CBA)
- Paralelný netopierí algoritmus (angl. parallel bat algorithm)

a iné.

Taktiež je možné vytvárať rôzne hybridné algoritmy ako napríklad:

- Netopierí algoritmus a algoritmus optimalizácie pomocou hejna častíc
- Netopierí algoritmus a algoritmus hľadania kukučky (angl. cuckoo search)

- Netopierí algoritmus a algoritmus simulovaného žíhania

a ďalšie.

Štandardný BA je používaný na riešenie spojitých optimalizačných problémov, ale nie je ho možné použiť pri riešení kombinačných optimalizačných problémov. Na riešenie týchto problémov bol algoritmus modifikovaný do diskrétnej formy, kde boli pridané niektoré metódy na prehľadávanie susedstva (angl. neighborhood search) a to:

- Výmena (angl. swap)
- Vloženie (angl. insert)
- Obrátenie (angl. inverse)
- Prekríženie (angl. crossover)

Jednou z najväčších slabostí klasického BA, je predčasná alebo klamlivá konvergencia (v niektorých prípadoch oboje), toto je spôsobené hlavne náhodnou inicializáciou parametrov, tento problém je možné vyriešiť pomocou CBA, ktorý inicializáciu rieši výpočtom pomocou Gaussových máp [4] alebo stanových máp [5] (angl. tent map).

Paralelný netopierí algoritmus na rozdiel od klasického BA rozdeľuje model na submodely, kde každý submodel má n netopierov a svoje vlastné najlepšie riešenie, v istom časovom intervale sa porovnajú najlepšie riešenia všetkých submodelov a vyberie sa najlepšie riešenie z najlepších. Toto riešenie sa v ďalšom kroku nastaví ako najlepšie riešenie vo všetkých submodeloch a postup sa opakuje. [16]

Využitie

Netopierí algoritmus má široké využitie, bol úspešne aplikovaný na mnoho oblastí ako napríklad na podporu rozhodovania či spracovanie obrázkov. Konkrétne bol použitý napríklad na prahovanie obrázkov, trénovanie neurónových sietí, na výpočet optimalizácie ekonomických nákladov (angl. economic load dispatch ELD), lokalizáciu uzlov v bezdrôtovej sieti a podobne.[16]

2.1.2 Algoritmus svätojánskych mušiek

Svätojánska muška alebo svetluška je označenie pre skupinu hmyzu s viac ako 2000 druhmi, pričom sa väčšinou vyskytujú v miernom a tropickom podnebí, konkrétne v okolí močiarov alebo zalesnených oblastiach. Väčšina lariev sú dravce, ktoré sa živia inými larvami alebo slimákmi. Pri dospelých jedincoch sa mnoho druhov živí peľom rastlín alebo nektárom, ale sú medzi nimi aj dravé druhy ktoré v niektorých prípadoch konzumujú aj svetlušky iného druhu.

Svoje pomenovanie získali na základe schopnosti bioluminiscencie[6], ktorá je zároveň aj

[4] Gaussova mapa je funkcia, ktorá v BA slúži na inicializáciu frekvencie a hlasitosti.

[5] Stanová mapa je funkcia, ktorej graf má tvar stanu, v BA sa využíva na inicializáciu rýchlosti vysielania impulzov

[6] Bioluminiscencia je schopnosť produkovať svetlo živým organizmom, svetlo môže byť produkované baktériami alebo samotnými zvieratami.

inšpiráciou pre daný algoritmus. Svetlušky produkujú takzvané studené svetlo, t.j. svetlo bez infračervených alebo ultrafialových frekvencií. Mušky vyžarujú svetlo v žltých, zelených alebo červených odtieňoch. Väčšina svetlušiek sú nočné živočíchy, ale existujú aj niektoré denné druhy, pričom všetky nočné druhy buď žiaria alebo si produkujú svetlo pomocou špeciálneho orgánu, ktorý sa nachádza v spodnom bruchu, denné druhy svetlo väčšinou nevyžarujú iba niektoré z nich v prípade, že žijú v tienistých oblastiach.

Svetlo vydávajú hlavne z troch dôvodov a to:

- Prilákanie partnera za účelom párenia
- Pri zháňaní koristi
- Obranný mechanizmus

Pri väčšine druhov sa nachádza obdobie párenia v letných mesiacoch a prebieha nasledovne, samičky mnohých druhov nelietajú iba vyžarujú silné svetlo, aby ich samčekovia mohli ľahšie nájsť, samčekovia lietajú ale produkujú len slabé a prerušované svetlo, následne si vyberú samičku a priletia za ňou. Nie všetky svetlušky používajú svetlo pri párení, ukázalo sa, že niektoré druhy využívajú feromóny a poradia si aj bez svetla.

Pri zháňaní potravy niektoré druhy svetlušiek napodobňujú spôsob svetelnej signalizácie menších druhov svetlušiek, aby prilákali samčekov a potom ich jedia .

Svetlo ako obranný mechanizmus je využívané hlavne pri larvách, kde slúži ako výstražný signál pre predátorov, nakoľko obsahujú chemikálie, ktoré sú pre predátorov toxické.

Ďalšou ich zaujímavou vlastnosťou je rovnako ako pri netopieroch schopnosť hibernácie, pričom na rozdiel od netopierov dokážu hibernovať aj niekoľko rokov, zahrabané niekde pod zemou.[7]

Algoritmus

Na koľko existuje mnoho druhov svätojánskych mušiek a použitie a spôsob bioluminescencie sa pri nich líši algoritmus pracuje s tromi idealizovanými pravidlami:

1. Pri svetluškách nás nezaujíma pohlavie a teda sú vzájomne medzi sebou priťahované bez ohľadu na pohlavie.
2. Atraktívnosť jednotlivých svetlušiek závisí od jasu, to znamená, že svetluška s nižším jasom sa bude pohybovať smerom k jasnejšej. Atraktívnosť klesá so stúpajúcou vzdialenosťou medzi jednotlivými jedincami. Ak sa v blízkosti nenachádza žiadna žiarivejšia svetluška, pohybuje sa v priestore náhodne.
3. Jas svetlušky závisí na objektívnej funkcii, pre problém maximalizácie môže byť jas jednoducho úmerný hodnote objektívnej funkcie.

Na začiatku je potrebné si definovať zmenu intenzity svetla (angl.light intensity) a výpočet príťažlivosti (angl. atractivness). Zmenu intenzity svetla je možné vypočítať pomocou nasledujúceho vzťahu:

$$I(r) = I_0 e^{-\gamma r 2} \tag{2.7}$$

, kde I_0 je počiatočná hodnota svetelnej intenzity a γ je koeficient absorbcie svetla, r je vzdialenosť medzi jednotlivými svetluškami.

Príťažlivosť je možné vypočítať pomocou nasledujúceho vzťahu:

$$\beta(r) = \beta_0 e^{-\gamma r 2} \tag{2.8}$$

, kde β_0 je príťažlivosť v $r = 0$. Vzdialenosť r je možné vypočítať pomocou nasledovného vzťahu:

$$r_{ij} = ||x_i - x_j|| \tag{2.9}$$

, kde x_i a x_j sú súradnice jednotlvých svetlušiek. Pohyb svetlušiek počítame pomocou nasledujúceho vzťahu:

$$x_i = x_i + \beta_0 e^{-\gamma r^2_{ij}}(x_j - x_i) + \alpha(rand - \frac{1}{2}) \tag{2.10}$$

, kde α je náhodný parameter, $rand$ je náhodné číslo získané pomocou rovnomerného rozloženia, ktoré leží na intervale $[0, 1]$. [25]

Pseudokód [25]

Algoritmus je možné implementovať pomocou nasledujúceho pseudokódu.

input: Nastavíme si kontrolné parametre μ,$b, rand$
Nastavíme si veľkosť populácie n
Nastavíme maximálny počet iterácií
Nastavíme hodnotu parametru γ
Nastavíme hodnotu parametru α
output: Optimálna pozícia svetlušky a jej fitness hodnota
Inicializujeme populáciu n svetlušiek na náhodných pozíciách
Nájdeme najlepšie riešenie na základe fitness hodnoty
while Nie sú splnené počiatočné podmienky **do**
    foreach $Svetluka_i$ **do**
    foreach $Svetluka_j$ **do**
        if $Svetluka_j$ je lepšia ako $Svetluka_i$ **then**
            Presuň $Svetluka_i$ smerom k $Svetluka_j$
            podľa vzorca 2.10;
    end
    Vyhodnoť polohu jednotlivých svetlušiek;
end

Varianty

Algoritmus je možné nájsť v rôznych variantách pre riešenie rôznych typov úloh ako napríklad:

- Diskrétny algoritmus svätojánskych mušiek
- Binárny algoritmus svätojánskych mušiek
- Chaotický algoritmus svätojánskych mušiek
- Paralelný algoritmus svätojánskych mušiek

a iné.
Algoritmus sa tiež vyskytuje v rôznych kombináciách s rôznymi inými algoritmami ako napríklad:

- Algoritmus svätojánskych mušiek a algoritmus harmonizovaného vyhľadávania
- Algoritmus svätojánskych mušiek a algoritmus mravčej kolónie
- Algoritmus svätojánskych mušiek a evolučný algoritmus

a iné.
K hybridizácii s ostatnými algoritmami dochádza najmä aby sa predišlo uviaznutiu v lokálnom minime, k minimalizácii počtu vyhodnocovania funkcií či na zlepšenie populačnej rozmanitosti.[16]

Využitie

Algoritmus svätojánskych mušiek sa v praxi využíva pri riešení rôznych problémov ako napríklad vyhľadávanie obrázkov, na výpočet ELD, pri riešení problému obchodného cestujúceho (angl. travelling salesman problem), problémov poradí (angl. queuing problem) a iných.[16]

2.2 Algoritmy inšpirované biológiou

Táto kapitola sa zaoberá algoritmami, ktoré čerpajú inšpiráciu z biológie ale nie inteligencie skupiny. Algoritmy inšpirované inteligenciou skupinu sú v podstate podmnožinou algoritmov inšpirovaných prírodou. Do tejto skupiny zaraďujeme algoritmy, ktoré sa zameriavajú na vlastnosti jedinca ako takého a nezaoberajú sa jeho úlohou v kolektíve, v niektorých prípadoch môže byť komplikované určiť, či daný algoritmus spadá do algoritmov, ktoré sa inšpirovali inteligenciou skupiny alebo sú zamerané na inteligenciu jedinca. [16]

Patrí sem napríklad algoritmus inšpirovaný termiťou kolóniou (angl. termite colony optimization), kdežto algoritmus inšpirovaný správaním mravcov (angl. ant colony optimization), ktorý funguje na podobnom princípe zaraďujeme medzi algoritmy inšpirované inteligenciou skupiny. Ďalej sem zaraďujeme algoritmus, ktorý využíva echolokáciu delfínov (angl. dolphine echolocation) na rozdiel od neho netopierí algoritmus, ktorý taktiež pracuje so schopnosťou echolokácie zaraďujeme medzi algoritmy inšpirované inteligenciou kolektívu. Zaraďujeme sem taktiež genetické algoritmy, algoritmus párenia včiel, algoritmus inšpirovaný

volaním japonských stromových žiab (angl. japanese tree frogs calling), tento algoritmus bol použitý pri riešení problému farbení máp a mnohé iné. V tejto kapitole sa podrobnejšie budeme venovať algoritmu opelenia kvetov (angl. flower pollination algorithm).

2.2.1 Algoritmus opelenia kvetov

Pod pojmom opeľovanie si predstavujeme dej, pri ktorom sa samčie rastlinné bunky, ktoré nazývame peľ, prenášajú na samičie orgány kvetu, ktoré nazývame piestik, vďaka čomu neskôr dochádza k oplodneniu, teda sa jedná o rozmnožovanie rastlín.
Spôsoby oplodnenia a tvorby semien rastlín sa líšia pri nahosemenných a krytosemenných rastlinách, avšak pre účely tejto práce informácie o oplodnení a tvorbe semena nie sú dôležité a preto sa im nebudem podrobnejšie venovať. Opeľovanie kvetov môžeme rozdeliť do dvoch základných kategórií a to na biotické (angl. biotic) a abiotické (angl. abiotic). Rozdiel medzi abiotickým a biotickým opelovaní je v tom, že kým pri biotickom opelení musí byť peľ prenesený živým organizmom tak pri abiotickom opelení dochádza k prenosu peľu pomocou vetru, vody či dažďa. Biotický spôsob opelenia je využívaný zhruba v osemdesiatich percentách prípadov. Existujú prípady, v ktorých rastliny pri opelovaní využívajú obidva spôsoby v závislosti na aktuálnych podmienkach.
Aj napriek tomu, že abiotický spôsob je menej využívaný prináša isté výhody, keďže rastlina nepotrebuje kvôli opeleniu prilákať živé organizmy, sústredí sa len na tvorbu peľu pričom nepotrebuje dávať energiu do tvorby kvetov či nektáru. Najviac používaným prostriedkom pri abiotickom opeľovaní je vietor a najmenej využívaným je dážď.
Tieto rastliny sa danému spôsobu opelenia prispôsobili napríklad svojou výškou, polohou kvetu, tyčiniek, samčích orgánov a piestika, čo podporuje šírenie a prenos peľu.
Opelenie vodou funguje na princípe toho, že voda slúži na prenos pelu z jedného kvetu do druhého, spôsob akým sa pel prenesie závisí od konkrétnej rastliny, tento druh opelenia využívajú hlavne vodné rastliny.
Opelenie dažďom je využívané vo veľmi malej miery, väčšinou v prípadoch keď dochádza k silným dažďom, čo odraďuje živočíchy od opelenia. Rastliny opeľované dažďom sú na to fyzicky prispôsobené nachádza sa v nich niečo ako dažďový odtok, pomocou ktorého sa peľ dostanem k samičím pohlavným orgánom a dochádza k oplodneniu.
Biotické opelovanie využíva na prenos pelu takzvaných opeľovačov (angl. pollinator), do opelenia sa zapája sto až dvesto tisíc živočíchov pričom väčšinu z nich tvorí hmyz, zvyšok tvoria niektoré druhy vtákov a netopierov (meganetopierov). Do opelenia sa zapájajú aj niektoré druhy opíc, lemurov a hlodavcov.
Biotické opeľovanie môžeme rozdeliť do dvoch skupín a to na základe opeľovačov, na rastliny opeľované hmyzom (angl. entomophily) a rastliny opeľované živočíchmi (angl. zoophily).
Rastliny, ktoré využívajú pri opelovaní hmyz majú farebné lupienky, silnú vôňu a tvoria nektár, tento druh opeľovania bol využívaný už v období dinosaurov. Opeľovanie živočíchmi a stavba daných kvetov sa líši v závislosti od druhu opeľovačov, kým rastliny opeľované netopiermi majú väčšinou bielu farbu, kvitnú v noci a majú silnú vôňu, kvety opelované s pomocou vtákov kvitnú cez deň, tvoria nektár a majú výraznejšiu farbu. Kvety opeľované živočíchmi majú obvykle väčšie rozmery ako kvety opelované hmyzom.
Ďalej si opelovanie môžme rozdeliť do kategórii podľa mechanizmu opelenia a to na samoopelenie (angl. self-pollination) a cudzoopelenie (angl. cross-pollination).
Samoopelenie je proces, pri ktorom dochádza buď k opeleniu toho istého kvetu na danej rastline alebo k opeleniu iného kvetu na tej istej rastline. Tento druh opelenia sa vyskytuje pri rastlinách s krátkou dobou života.

Cudzoopelenie nastáva vtedy, keď sa peľ z jednej rastliny prenesie na inú rastlinu toho istého druhu. [13]

Algoritmus

Vyššie spomínané informácie o opeľovaní rastlín si pre potreby algoritmu zhrnieme do nasledujúcich pravidiel:

1. Biotické opelenie a cudzoopelenie považujeme za globálny opeľovací proces pri ktorom opeľovače presúvajú peľ pomocou Levyho letov [7] (angl. Levy flight).
2. Abiotické opelenie a samoopelenie považujeme za lokálne opelenie.
3. Stálosť kvetov môžeme považovať za pravdepodobnosť reprodukcie ktorá je úmerná podobnosti dvoch zúčastnených kvetov.
4. Lokálne a globálne opeľovanie je regulované pravdepodobnosťou prepnutia $p \epsilon [0, 1]$, vzhľadom na fyzickú blízkosť a ostatné faktory ako napríklad vietor môže mať lokálne opelovanie významný podiel na celkovom opelovaní.

V prírode má väčšina rastlín viacej kvetov a každý kvet má veľké množstvo gamét [8], na rozdiel od skutočných rastlín algoritmus pracuje s rastlinami, ktoré majú len jeden kvet a ten má len jednu gamétu, vďaka čomu pri riešení problému nemusíme rozlišovať, či sa jedná o gamétu, kvet či rastlinu. V našom prípade x_i označuje kvet alebo gamétu prípadne oboje.

Na základe uvedených pravidiel môžeme algoritmus rozdeliť na dve časti a to globálne a lokálne opelovanie. V prípade globálneho opeľovania, je peľ prenášaný hmyzom na väčšie vzdialenosti, čo nám zaručuje, že k opeleniu sa dostane len najlepší jedinec, ktorého získame pomocou fitness funkcie g_*, čo môžeme zapísať pomocou nasledujúceho vzťahu:

$$x_i(t+1) = x_i(t) + L(x_i(t) - g_*) \tag{2.11}$$

, kde $x_i(t)$ je vektor riešenia i v čase t, g_* je najlepšie riešenie v danom kroku nájdené prehľadaním všetkých súčasných riešení v rámci jednej iterácie alebo v rámci jednej generácie. Parameter L nám určuje "silu opelenia", čo je v podstate veľkosť kroku, ktorú získavame pomocou Lévyho letu a to nasledovne:

$$L \sim ((\lambda\Gamma(\lambda) sin(\pi\lambda \div 2)) \div \pi) * (1 \div s^{1+\gamma}) \tag{2.12}$$

, pričom $\Gamma(\lambda)$ je štandardná gamma funkcia, táto Lévyho distribúcia je vhodná hlavne pre veľké kroky.

Na výpočet lokálneho opelenia používame nasledovný vzťah:

$$x_i(t+1) = x_i(t) + \varepsilon(x_j(t) - x_k(t) \tag{2.13}$$

[7] Lévyho let je náhodná prechádzka pri ktorej je dĺžka krokov určená pomocou rozdelenia pravdepodobnosti [10]

[8] gaméta = pohlavná bunka

, kde $x_j(t)$ a $x_k(t)$ predstavujú peľ z rôznych kvetov toho istého druhu rastliny, parameter ε získame z rovnomerného rozdelenia, ktoré leží na intervale $[0, 1]$. U väčšiny kvetov môže dôjsť k obom spôsobom opelenia lokálnemu aj globálnemu. V praxi dochádza častejšie k lokálnemu opeľovaniu kvetov, ktoré ležia blízko pri sebe, aby sme to docielili využívame štvrté pravidlo, vďaka ktorému dochádza k prepínaniu medzi globálnym a lokálnym opelením. [27]

Pseudokód

Vyššie opísaný algoritmus je možné zapísať formou nasledujúceho pseudokódu:

input: prehľadávací priestor
fitness funkcia
počet kvetov alebo gamét
n počet iterácií
p pravdepodobnosť prepnutia
Inicializujeme populáciu o veľkosti n kvetov respektíve gamét s náhodnými riešeniami;
V počiatočnej populácii nájdeme najlepšie riešenie $g*$;
Nastavíme čas t=0;
while $t < n$ **do**
foreach Kvet i v skupine riešení i **do**
if $rand < p$ **then**
Vyberieme spomedzi všetkých riešení najlepšie riešenie;
Generujeme nové lokálne riešenia okolo najlepšieho riešenia;
end
if $rand < A_i$ a zároveň fitness funkcia nového riešenia je lepšia ako pôvodná **then**
Vypíš d-dimenzionálny vektor L, ktorý pracuje s Lévyho distribúciou;
Aplikuj vzťah určený na výpočet globálneho opelenia 2.11;
else
Vypíš hodnotu ε z rovnomerného rozdelenia;
Náhodne vyber j, k riešenia z aktuálnej sady všetkých riešení;
Aplikuj vzťah určený na výpočet lokálneho opelenia 2.13;
end
Vyhodnotenie nového riešenia, ak je riešenie lepšie nahradí sa ním riešenie i;
end for
Aktualizácia najlepšieho riešenia;
end

Varianty

Algoritmus bol upravený pre riešenie rôznych problémov vďaka čomu sa vyskytuje v rôznych variantách ako napríklad:

- Binárny algoritmus opelenia kvetov (angl. binary FPA)
- Chaotický algoritmus opelenia kvetov (angl. chaotic FPA)

a iné.

Taktiež je možné vytvárať rôzne hybridné algoritmy ako napríklad:

- Algoritmus opelenia kvetov a algoritmus optimalizácie pomocou hejna častíc
- Algoritmus opelenia kvetov a algoritmus harmonického vyhľadávania

a ďalšie.

Chaotický FPA je realizovaný pomocou chaotických máp ako napríklad kruhové mapy, Gaussove mapy, Chebyshevove mapy a podobne. Hlavným cieľom tejto modifikácie je vylepšiť parametre FPA a zlepšiť tak jeho presnosť. Parameter, ktorý má najväčší vplyv na výpočet je parameter p, ktorý slúži na prepínanie medzi lokálnym a globálnym spôsobom opelenia. [16]

Využitie

Algoritmus opelenia kvetov, sa používa na riešenie rôznych problémov ako napríklad, spracovanie obrázkov, v strojárstve a podobne. Bol použitý tiež na segmentáciu obrázkov, na výpočet ELD, problémy s návrhom transformátora a iné. [16]

2.3 Algoritmy inšpirované fyzikálnymi a chemickými javmi

Optimalizačné algoritmy čerpajú inšpiráciu nielen z prírody ale aj z chemických a fyzikálnych javov, ako napríklad gravitácia, riečne systémy, elektrické náboje a podobne. Patria sem algoritmy ako napríklad algoritmus simulovaného žíhania, cyklu vody (angl. water cycle algorithm), špirálovej optimalizácie (angl. spiral optimization), veľkého tresku (angl. big bang-big crunch), harmonického vyhľadávania (angl. harmony search), tento algoritmus bol inšpirovaný hudbou, a mnoho iných. V nasledujúcej časti sa budeme bližšie venovať algoritmu čiernych dier. [16]

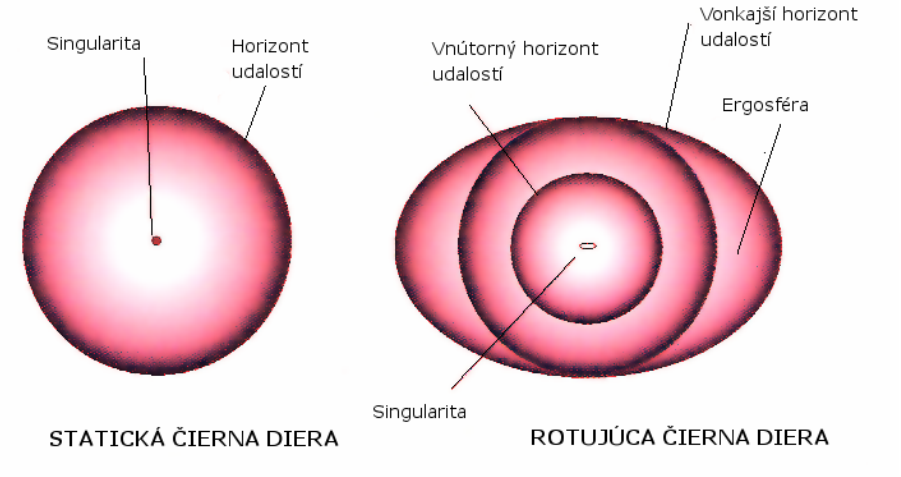
2.3.1 Algoritmus čiernych dier

Čierna diera je hmota respektíve vesmírne teleso veľkých rozmerov, ktoré má tak silnú gravitáciu, že keď sa nejaké teleso dostane do jej blízkosti pohltí ho a nie je možné z nej uniknúť, von sa nedostane ani svetlo.
Prvé zmienky o čiernych dierach siahajú do 18.storočia kedy anglický vedec John Mitchell prvýkrát prišiel s myšlienkou, že vo vesmíre by mohli existovať neviditeľné telesá s tak silnou

gravitáciou, že z nich neunikne ani svetlo. Väčší záujem o čierne diery môžeme pozorovať od druhej polovice 20. storočia, kedy im bol aj pridelený názov Čierna diera.
Čierne diery vznikajú vďaka procesu ktorý nazývame gravitačné zrútenie, vo väčšine prípadov k nemu dochádza u hviezd. Gravitačné zrútenie nastane vtedy, keď vnútorný tlak objektu nie je dostatočne veľký aby odolal vlastnej gravitácii, pri hviezdach k nemu môže dôjsť napríklad, keď má hviezda príliš málo paliva na udržanie svojej teploty. Akonáhle sa proces gravitačného zrútenia spustí nie je možné ho zastaviť žiadnym spôsobom. K vzniku čiernej diery je potrebný gravitačný kolaps hviezdy, ktorá je minimálne trikrát väčšia ako Slnko. Pričom čierne diery po vzniku majú zhruba štvrtinu pôvodnej veľkosti hviezdy z ktorej vznikli. Čierna diera môže po vytvorením rásť absorbovaním okolitých objektov. Môže dôjsť aj k zlúčeniu viacerých čiernych dier dohromady, takto vznikajú supermasívne čierne diere, ktoré sú väčšinou umiestnené v strede galaxií. Čierne diery nie je veľmi možné pozorovať je ich možné nájsť, buď podľa hviezd alebo iných objektov, ktoré okolo nich obiehajú aj keď to pôsobí, že tam nič nie je alebo pri pozorovaní hmoty, ktorá smeruje k čiernej diere a nakoniec je ňou absorbovaná.
Čierne diery môžeme rozdeliť na rotujúce (angl. spinning black holes) a statické čierne diery (angl. non-spinnig black holes). Statické čierne diery sa skladajú z horizontu udalostí (angl. event horizont) a singularity (angl. singularity). Horizont udalostí je najvrchnejšia vrstva čiernej diery je to hranica časopriestoru cez ktorú môže hmota a svetlo prechádzať dovnútra ale v dôsledku silného gravitačného poľa sa už von nedostane. Pre pozorovateľa nie je možné určiť či udalosť, ktorá sa odohrala na hranici horizontu udalosti sa naozaj stala alebo nie, preto dostal toto pomenovanie. V blízkosti horizontu udalostí dochádza k nárastu takzvanej dilatácie (spomalenia) času (angl. gravitational dilatation time), to znamená, že pozorovateľovi sa môže zdať, že objektu trvá nekonečne dlho priblížiť sa k horizontu udalostí. Vzdialenosť v ktorej sa je možné vyhnúť vtiahnutiu do čiernej diery je určená Schwarzschilldovým polomerom, ak je vdialenosť menšia objekt sa už nemôže vyhnúť vtiahnutiu do čiernej diery túto vzdialenosť je možné vypočítať na základe vzťahu:

$$r_S = \frac{2Gm}{c^2} \tag{2.14}$$

, kde G je gravitačné zrýchlenie, m je hmotnosť telesa a c je konštanta ktorá udáva rýchlosť svetla. Predpokladá sa, že uprostred čiernej diery leží singularita, čo je oblasť v ktorej je časopriestorové zakrivenie nekonečné. Singularita je len bod uprostred, ktorý má nulový objem a nekonečnú hustotu, teleso po prekročení horizontu udalostí je priťahované k tomuto bodu, po dosiahnutí singularity sú objekty rozdrvené na nekonečnú hustotu o ktorú sa objem danej čiernej diery zväčší. Rotujúce čierne diery sa skladajú vonkajšieho horizontu udalostí (angl. outer event horizont), ergosféry (angl. ergosphere), vnútorného horizontu udalostí (angl. inner event horizont) a singularity. Algoritmus čerpá inšpiráciu zo statických čiernych dier preto si rotujúce čierne diery nebudeme podrobnejšie popisovať, rozdiely v ich rozložení môžete vidieť na obrázku 2.2. [4]



Obr. 2.2: Zloženie čiernej diery [15]

Algoritmus

BH algoritmus patrí do skupiny algoritmov založených na populácii (angl. population-based). Pre potreby algoritmu si definujeme nasledujúce pravidlá:

1. Na začiatku si generujeme určitý počet hviezd, ktorý sa v priebehu riešenia nemôže zmeniť.
2. Hviezdy náhodne umiestnime v prehľadávacom priestore na rôznych pozíciách.
3. Na základe fitness funkcie zvolíme najlepšie hodnotenú hviezdu za čiernu dieru.
4. Ak je hviezda od čiernej diery v menšej vzdialenosti ako udáva Schwarzschildov polomer, zmizne a namiesto nej sa vygeneruje nová hviezda na náhodné miesto v rámci prehľadávacieho priestoru.

Pohltenie hviezd čiernou dierou získame pomocou nasledujúceho vzťahu:

$$x_i(t+1) = x_i(t) + rand * (x_{BH} - x_i(t)) \tag{2.15}$$

, kde $x_i(t)$ udáva súradnice i-tej hviezdy v čase t, a $x_i(t+1)$ sú jej súradnice v čase $t+1$, $rand$ je náhodné číslo z intervalu $[0, 1]$ a x_{BH} nám udáva pozíciu čiernej diery.

V prípade, že hviezda prekročí horizont udalostí je čiernou dierou pohltená a namiesto nej sa vygeneruje ďalšia hviezda. Hviezda sa na horizont udalostí dostane v prípade, že prekročí Schwarzschildov polomer, čo zistíme na základe nasledujúceho vzťahu:

$$r = f_{BH} \div \sum_{i=1}^{N} f_i \tag{2.16}$$

, kde f_{BH} je fitness funkcia čiernej diery, a f_i je fitness funkcia i-tej hviezdy. V prípade, že vzdialenosť medzi hviezdou a čiernou dierou je menšia ako r, hviezda je čiernou dierou pohltená. [17]

Pseudokód [17]

Inicializujeme populáciu hviezd s náhodnými umiestneniami v prehľadávacom priestore
Loop
foreach Hviezdu vyhodnotíme hodnoty objektívnej funkcie;
Zvolíme najlepšie ohodnotenú hviezdu podľa fitness funkcie a nastavíme ju ako čiernu dieru;
Zmeníme polohu všetkých hviezd podľa rovnice;
if Hviezda má lepšie hodnotenie ako čierna diera **then**
Nastavíme hviezdu ako novú čiernu dieru;
if Hviezda prekročila horizont udalostí **then**
Vygeneruj novú hviezdu s náhodným umiestnením;
if Sú splnené koncové kritériá **then**
Vystúp z cyklu;
End loop

Využitie

BH algoritmus je používaný pri riešení problémov zhlukovania (angl. clustering), jeho úspešnosť pri testovaní bola porovnávaná s ďalšími algoritmami ako napríklad PSO, K-means, algoritmus gravitačného vyhľadávania. Pri testovaní bol použitý napríklad Friedmanov test[9] a Holmova metóda[10], na základe testovania sa ukázalo, že algoritmus BH bol v takmer väčšine prípadov úspešnejší ako ostatné algoritmy. [17]

[9] Friedmanov test je neparametrický štatistický test.[8]
[10] Holmova metóda sa využíva v štatistike pri riešení problémov viacnásobného porovnávania. [9]

Kapitola 3

Riešené optimalizačné úlohy

3.1 Hľadanie extrému funkcie

Extrémom funkcie rozumieme hodnotu funkcie, ktorá nadobúda buď najnižšiu hodnotu spomedzi všetkých funkčných hodnôt, nazývame minimum funkcie, alebo najvyššiu hodnotu, nazývame maximum funkcie. Pričom rozlišujeme globálne a lokálne extrémy.
Globálne extrémy nám určujú maximum alebo minimum funkcie na celom definičnom obore, kým lokálne extrémy nám určujú maximum alebo minimum len na určitom intervale z definičného oboru.
Pričom pod pojmom hľadanie extrému funkcie rozumieme snahu nájsť minimum alebo maximum funkcie na intervale, respektíve sa čo najviac priblížiť jeho hodnote, nakoľko úlohou optimalizačných algoriotmov častokrát kvôli veĺkej náročnosti nie je nájsť najlepšie riešenie, ale priblížiť sa mu v rámci istej tolerancie.

3.2 Popis jednotlivých funkcií

Na hľadanie extrémov som si vybrala Griewankovu, Rastringinovu a Rosenbrockovu funkciu, hlavne kvôli ich častému použitiu pri testovaní optimalizačných algoritmov.

Griewankova funkcia

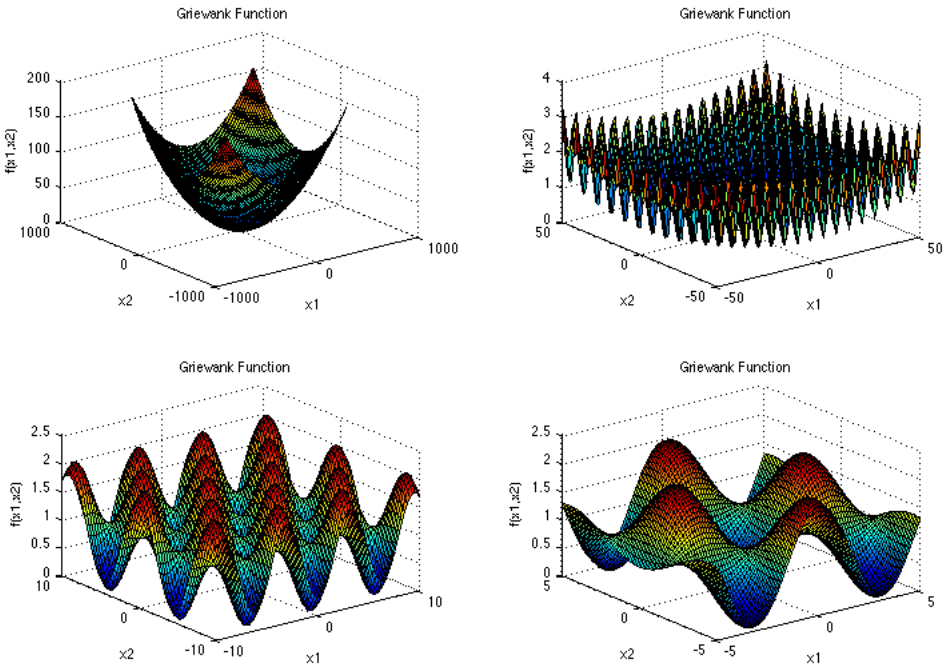
Griewankova funkcia je charakteristická tým, že má mnoho lokálnych miním, ktoré sú pravidelne distribuované, čo môžeme vidieť na obrázku 3.1.
Predpis Griewankovej funkcie je nasledovný: [21]

$$f(x_1 \cdots x_n) = 1 + \frac{1}{4000} \sum_{i=1}^{n} x_i^2 - \prod_{i=1}^{n} cos(\frac{x_i}{\sqrt{i}}) \tag{3.1}$$

Griewankova funkcia:

- Je spojitá
- Je nekonvexná
- Môže byť definovaná v n-dimenzionálnom priestore
- Unimodálna funkcia[1] [18]

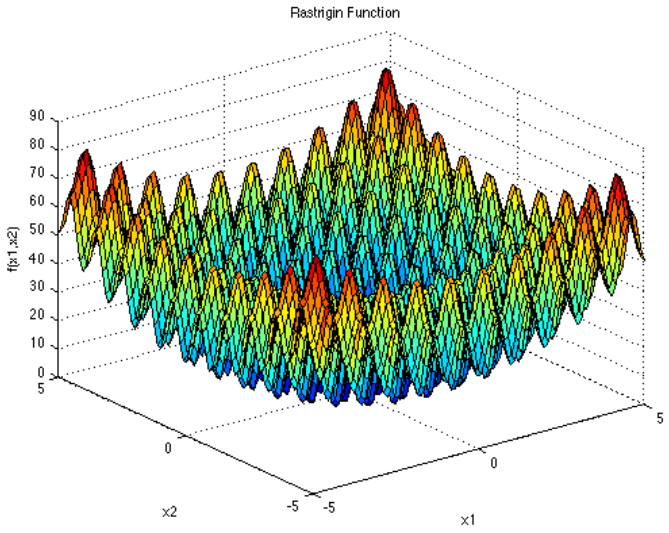
[1] Funkcia s jedným údolím alebo vrcholom



Obr. 3.1: Znázornenie Griewankovej funkcie[21]

Rastringininova funkcia

Rastringininova funkcia má niekoľko lokálnych miním, ktoré sú pravidelne distribuované, čo môžeme vidieť na obrázku 3.2.



Obr. 3.2: Znázornenie rastringinovej funkcie [22]

Predpis Rastringininovej funkcie je nasledovný:[22]

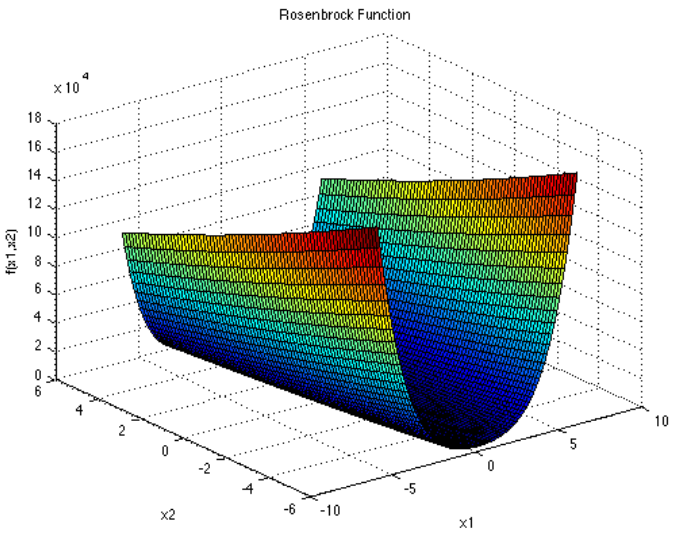
$$f(x_1 \cdots x_n) = 10n + \sum_{i=1}^{n}(x_i^2 - 10cos(2\pi x_i)) \tag{3.2}$$

Rastringinova funkcia:

- Je spojitá
- Je konvexná
- Môže byť definovaná v n-dimenzionálnom priestore
- Multimodálna funkcia[2]
- Funkcia je diferencovateľná [3]
- Funkcia je oddeliteľná[4] [19]

Rosenbrockova funkcia

Rosenbrockova funkcia je tiež označovaná ako banánová (angl. Banana function) alebo údolná funkcia (angl. Valley function), často sa využíva pri optimalizačných algoritmoch založených na gradiente (angl. gradient-based), môžeme ju vidieť na obrázku 3.3.



Obr. 3.3: Znázornenie Rosenbrockovej funkcie[23]

Predpis Rosenbrockovej funkcie je nasledovný:[23]

$$f(x_1 \cdots x_n) = \sum_{i=1}^{n-1}(100(x_i^2 - x_{i+1})^2 + (1 - x_i)^2) \tag{3.3}$$

[2]Funkcia s viacerými vrcholmi alebo údoliami.

[3]Diferencovatelná funkcia je taká funkcia, ktorá má v určitom bode diferenciál.[24]

[4]Funkcia 2 nezávislých premenných je oddeliteľná, ak sa dá vyjadriť ako súčin 2 funkcií, pričom každá z nich závisí len od jednej premennej.[1]

Rosenbrockova funkcia:

- Je spojitá
- Je konvexná
- Môže byť definovaná v n-dimenzionálnom priestore
- Multimodálna funkcia
- Funkcia je diferencovateľná
- Funkcia je neoddeliteľná[20]

Kapitola 4

Popis aplikácie

Táto kapitola sa zaoberá popisom implementácie aplikácie, vrátane programovacieho jazyka a knižníc, ktoré boli pri implementácii využité. Taktiež je popísané uživateľské rozhranie a spôsob akým uživateľ s týtmo rozhraním pracuje.

4.1 Implementácia aplikácie

Programovací jazyk

Aplikácia a všetky jej súčasti boli implementované pomocou programovacieho jazyku Python, ktorý bol vyvinutý v 90. rokoch minulého storočia pôvodne ako skriptovací jazyk. Jazyk Python má v súčastnosti veľmi široké použitie nielen pri písaní skriptov ale aj pri tvorbe rôznych aplikácií.
Jeho charakteristické vlastnosti sú:

- je to interpretovaný vysokoúrovňový programovací jazyk
- je objektovo orientovaný
- je dynamicky typovaný
- je prenositeľný
- dbá na čitatelnosť kódu
- na rozdiel od mnohých iných jazykov využíva odsadenie (angl. indentation) namiesto zložených zátvoriek

Použité knižnice

Pri implementácii boli použité rôzne matematické knižnice a to konrétne, knižnica *random* na generovanie pseudonáhodných čísel, knižnica *math*, ktorá bola použitá pri implementácii testovacích funkcií a optimalizačných algoritmov, z tejto knižnice boli použité najmä funkcie *sinus* a *cosinus*, taktiež bola pri implementácii algoritmov použitá knižnica *numpy* a *scipy*.
Pri implementácii grafického užívateľského rozhrania bola využitá knižnica *tkinter*.

Implementácia

Implementácia jednotlivých algoritmov prebiehala na základe pseudokódov uvedených v druhej kapitole, pri niektorých algoritmoch bolo potrebné počas implementácie vykonať zmeny pre spresnenie výpočtu algoritmov, konkrétne sa jednalo o algoritmus FFA, kde bolo potrebné upraviť vzorec, ktorý slúžil na pohyb jednotlivých svetlušiek, pri zmene som sa inšpirovala jedným z existujúcich riešení.
Ďalším algoritmom pri ktorom došlo k výraznejšej zmene oproti pseudokódu bol FPA, kde sa funkcia na výpočet Lévyho letu prevzala už z existujúceho riešenia, nakoľko existuje viacej možností pre výpočet Lévyho letu a vzorec na jeho výpočet uvedený v článku nebol pre algoritmus použiteľný. Obe zmeny sú označené v zdrojových kódoch spolu s odkazom na zdroj z ktorého som čerpala.

4.2 Užívateľské rozhranie

Hlavnou myšlienkou návrhu užívateľského rozhrania bolo spraviť ho užívateľský prívetivé (angl. user friendly), pri spustení aplikácie sa užívateľovi zobrazí jednoduché okno, kde si užívateľ zvolí jeden z optimalizačných algoritmov a funkciu, na ktorej ho chce otestovať, B.1 následne klikne na tlačidlo *Continue* B.2.
Po stlačení tohto tlačidla sa mu zobrazí zvolený algoritmus, B.3 kde si nastaví parametre, ktoré budú využité pri výpočte ako napríklad počet jedincov, maximálny počet iterácií a podobne, taktiež sa mu zobrazí graf zvolenej funkcie a hodnota jej globálneho minima.
Po zadaní všetkých parametrov užívateľ klikne na tlačidlo *Start* B.4, po kliknutí sa spustí výpočet B.5 a následne sa vypíše nájdená hodnota. B.6
V prípade, že užívateľ chce nanovo hľadať hodnotu minima s rovnakými parametrami, na rovnakej testovacej funkcii s pomocou rovnakého algoritmu stačí znovu stlačiť tlačidlo *Start*, čo výpočet spustí nanovo.
V prípade, že chce hľadať minimum pre tú istú funkciu pomocou toho istého algoritmu ale s inými parametrami zmení hodnoty parametrov a následne stlačí tlačidlo *Start*.
V prípade, že by chcel zmeniť testovaciu funkciu, optimalizačný algorimus prípadne oboje je nutné po tejto zmene znova klinúť na tlačidlo *Continue* a následne pokračovať zadaním parametrov.

Kapitola 5

Experimentovanie a zhrnutie výsledkov

Táto kapitola sa zaoberá experimentami, ktoré boli vykonané na vyššie spomínaných algoritmoch pomocou určených testovacách funkcií a ich výsledkami. Nakoľko každý algoritmus pracuje s rôznymi parametrami, ktoré ovplyvňujú rýchlosť aj presnosť výpočtu, boli porovnávané výsledky v rámci jedného algoritmu pre rôzne hodnoty jeho parametrov a zároveň bola porovnávaná aj úspešnosť jednotlivých algoritmov medzi sebou. Cieľom experimentov bolo určit najúspešnejší algoritmus pri hľadaní minima funkcie pre jednotlivé testovacie funkcie.
Aj napriek tomu, že všetky algoritmy majú rôzne parametre, existujú parametre, ktoré sú pre všetky algoritmy rovnaké a to počet dimenzií funkcie, maximálny počet iterácií a počet jedincov v populácii.
Ako už bolo spomínané každý algoritmus má aj svoje jedinečné parametre pre jednotlivé algoritmy sú to:

- minimálna a maximálna frekvencia (označujeme fmin a fmax)
- konštanta alfa
- náhodný vektor získaný z rovnomerného rozdelenia (označujeme beta)
- kontrolný parameter pre rýchlosť vysielania impulzov (označuejeme gamma)
- hlasitosť (označujeme A)
- rýchlosť vysielania impulzov (označujeme r)

, pre netopierí algoritmus.

- náhodný parameter alfa
- príťažlivosť (označujeme beta)
- koeficient absorbcie svetla (označujeme gamma)

, pre algoritmus svätojánskych mušiek.

- pravdepodobnosť prepnutia
- parameter lambda

, pre algoritmus opeľovania kvetov.

Algoritmus čiernych dier nemá žiadne špecifické parametre.
Pri všetkých experimentoch sú uvedené najlepšie, najhoršie a priemerné hodnoty,[1] pre hodnotu parametru, ktorý bol najúspešnejší, hodnoty pre ostatné, menej úspešné hodnoty, daného parametru sú uvedené v prílohách A.

5.1 Griewankova funkcia

Globálne minimum Griewankovej funkcie má hodnotu nula, v rámci testovania algoritmy pracovali na intervale <-10,10>. Predpis Griewankovej funkcie je uvedený v 3.1.

Prvý experiment

Cieľom prvého experimentu bolo nájsť, taký počet jedincov, pre ktorý sa bude výsledok blížiť hodnote globálneho minima a zároveň nebude jeho hľadanie trvať príliš dlho. Počet jedincov bol testovaný na hodnotách 1, 10, 50, 100, 200, 500, 1000 a 2000 s výnimkou algoritmu opeľovania kvetov a algoritmu čiernych dier kde minimálny počet jedincov bol dvaja.

Netopierí algoritmus

Počet jedincov	Najlepší výsledok	Najhorší výsledok	Priemerný výsledok
200	0,007573102577	0,01723131613	0,03213877634

Algoritmus dosahoval relatívne presné výsledky aj pri nižšom počte iterácií a preto sa v ďalších experimentoch bude pracovať s populáciou 200 jedincov.

Algoritmus svätojánskych mušiek

Počet jedincov	Najlepší výsledok	Najhorší výsledok	Priemerný výsledok
10	0,01433742719	1,338519517	0,6135962388

Experiment priniesol zaujímavý výsledok a to, že algoritmus dosiahol presnejšie výsledky s nižším počtom jedincov, naďalej sa bude pracovať s 10 jedincami.

Algoritmus opelovania kvetov

Algoritmus bol menej presný ako netopierí algoritmus a preto v ďalších experimentoch pracuje s počtom jedincov 500.

[1] Pri každom experimente sa vykonalo 10 testov

Počet jedincov	Najlepší výsledok	Najhorší výsledok	Priemerný výsledok
500	0,00764675031	0,03218509109	0,01653232427

Algoritmus čiernych dier

Počet jedincov	Najlepší výsledok	Najhorší výsledok	Priemerný výsledok
200	0	0,007396053994	0,00147920946

Experiment ukázal, že algoritmus dosahuje veľmi presné výsledky aj pri nižšom počte iterácií, v ďalších experimentoch pracoval s hodnotou 200.

Druhý experiment

V druhom experimente som testovala schopnosť priblížiť sa hodnote minima pri výpočte cez viaceré dimenzie a to cez 2, 3, 4, 5 dimenzií, prvý experiment pracoval s 2 dimenziami.

Netopierí algoritmus

Počet dimenzií	Najlepší výsledok	Najhorší výsledok	Priemerný výsledok
2	0,007692743786	0,0403162668	0,01828576928

Algoritmus svätojánskych mušiek

Počet dimenzií	Najlepší výsledok	Najhorší výsledok	Priemerný výsledok
2	0,02287229099	0,9990734594	0,5197493197

Algoritmus opelovania kvetov

Počet dimenzií	Najlepší výsledok	Najhorší výsledok	Priemerný výsledok
2	0,008365510621	0,02756232835	0,01894701647

Algoritmus čiernych dier

Počet dimenzií	Najlepší výsledok	Najhorší výsledok	Priemerný výsledok
2	0	0,007396043327	0,005177543468

Ako je vidieť z tabuliek A.1.2, algoritmy, s výnimkou algoritmu čiernych dier, boli menej presné s vyšším počtom dimenzií, v ďalších experimentoch algoritmy pracujú s dvomi dimenziami.

Tretí experiment

Účelom tretieho experimentu bolo porovnať presnosť s rastúcim počtom iterácií a to 100, 200, 500 a 1000.

Netopierí algoritmus

Počet iterácií	Najlepší výsledok	Najhorší výsledok	Priemerný výsledok
100	0,00996216925	0,02061282859	0,0226013252

Experiment ukázal, že s rastúcim počtom iterácií nedochádza k výraznému zlepšeniu výpočtu a preto v ďalších experimentoch pracujeme s maximálnym počtom iterácií 100.

Algoritmus svätojánskych mušiek

Počet iterácií	Najlepší výsledok	Najhorší výsledok	Priemerný výsledok
1000	0,0000008975443623	0,7762232803	0,1555533201

Ako najúspešnejší sa ukázal počet iterácií nastavený na 1000 a s touto hodnotou algoritmus pracoval aj v ďalších experimentoch.

Algoritmus opelovania kvetov

Počet iterácií	Najlepší výsledok	Najhorší výsledok	Priemerný výsledok
500	0,003124347856	0,02428250632	0,01175231318

Ako najúspešnejší sa ukázal počet iterácií nastavený na 500 a s touto hodnotou algoritmus pracoval aj v ďalších experimentoch.

Algoritmus čiernych dier

Počet iterácií	Najlepší výsledok	Najhorší výsledok	Priemerný výsledok
1000	0	0	0

Ako najúspešnejší sa ukázal počet iterácií nastavený na 1000 ale algoritmus dosiahol veľmi dobré výsledky aj pri nižšom počte iterácií.

Štvrtý experiment

Cieľom štvrtého experimentu bolo zistiť aký vplyv má na výpočet hodnota parametru beta, testované hodnoty boli: 0.1, 0.3, 0.5, 0.7 a 0.9, experiment prebiehal len pri netopiérom algoritme a algoritme svätojánskych mušiek.

Netopierí algoritmus

Hodnota beta	Najlepší výsledok	Najhorší výsledok	Priemerný výsledok
0,3	0,00991705896	0,02277137025	0,01356417137

Ako najvhodnejšia sa ukázala hodnota parametru beta 0.3 s ktorou sa pracovalo v ďalších experimentoch.

Algoritmus svätojánskych mušiek

Hodnota beta	Najlepší výsledok	Najhorší výsledok	Priemerný výsledok
0,5	0,0002126575541	0,03924737346	0,0168604054

Ako najvhodnejšia sa ukázala hodnota 0.5 s ktorou sa pracovalo v ďalších experimentoch.

Piaty experiment

Piaty experiment mal zistiť aký vplyv má na výpočet hodnota parametrov A a r, pričom ich hodnoty boli rovnaké, testované hodnoty boli: 0.1, 0.3, 0.5, 0.7 a 0.9 experiment prebiehal len na netopiérom algoritme.

Netopierí algoritmus

Hodnota A a r	Najlepší výsledok	Najhorší výsledok	Priemerný výsledok
0,7	0,01218740669	0,06470842586	0,02912040092

Ako najlepšia sa ukázala hodnota 0.7 s ktorou sa pracuje v ďalších experimentoch.

Šiesty experiment

Šiesty experiment mal zistiť aký vplyv má na výpočet hodnota parametrov alfa a gamma, pričom ich hodnoty boli rovnaké, testované hodnoty boli: 0.1, 0.3, 0.5, 0.7 a 0.9. Testovanie prebehlo na netopiérom algoritme a algoritme svätojánskych mušiek.

Netopierí algoritmus

Hodnota alfa a gamma	Najlepší výsledok	Najhorší výsledok	Priemerný výsledok
0,5	0,0005564188455	0,03024721729	0,01389304808

Ako najlepšia sa ukázala hodnota 0.5 s ktorou sa pracuje v ďalších experimentoch.

Algoritmus svätojánskych mušiek

Hodnota alfa a gamma	Najlepší výsledok	Najhorší výsledok	Priemerný výsledok
0,1	0,007401660845	0,2394120335	0,0626120308

Ako najlepšia hodnota sa ukázala hodnota 0.1.

Siedmy experiment

Siedmy experiment mal zistiť aký vplyv má na výpočet hodnota maximálnej frekvencie, testované hodnoty boli: 2, 4, 6, 8 a 10 testovanie prebehlo len na netopiérom algoritme.

Netopierí algoritmus

Maximálna frekvencia	Najlepší výsledok	Najhorší výsledok	Priemerný výsledok
6	0,007435026664	0,02554925514	0,01458289157

Ako najlepšia sa ukázala frekvencia s hodnotou 6.

Ôsmy experiment

Cieľom ôsmeho experimentu bolo zistiť aký vplyv má na výpočet hodnota pravdepodobnosti prepnutia, testované hodnoty boli: 0.1, 0.3, 0.5, 0.7 a 0.9 experiment prebehol len na algoritme opeľovania kvetov.

Algoritmus opelovania kvetov

Parameter prepnutia	Najlepší výsledok	Najhorší výsledok	Priemerný výsledok
0,5	0,007893794827	0,03786082396	0,01649626589

Ako najvhodnejšia sa ukázala hodnota 0.5 s ktorou sa pracovalo v ďalších experimentoch.

Deviaty experiment

Cieľom deviateho experimentu bolo zistiť najlepšiu hodnotu parametru lambda, testované hodnoty boli 0.5, 1.5, 5 a 15 experiment prebehol len na algoritme opeľovania kvetov.

Algoritmus opelovania kvetov

Parameter lambda	Najlepší výsledok	Najhorší výsledok	Priemerný výsledok
5	0,004630787577	0,03023308173	0,01595172248

Ako najlepšia hodnota sa ukázala hodnota 5.

Zhrnutie

Ako najefektívnejší algoritmus sa ukázal algoritmus čiernych dier narozdiel od neho najmenej presný bol algoritmus svätojánskych mušiek.

5.2 Rastringinova funkcia

Globálne minimum Rastringinovej funkcie má hodnotu nula, v rámci testovania algoritmy pracovali na intervale <-10,10>. Predpis rastringinovej funkcie je uvedený v 3.2.

Prvý experiment

Cieľom prvého experimentu bolo nájsť, taký počet jedincov, pre ktorý sa bude výsledok blížiť hodnote globálneho minima a zároveň nebude trvať príliš dlho. Počet jedincov bol testovaný na hodnotách 10, 50, 100, 200, 500, 1000, 2000 a 5000.

Netopierí algoritmus

Počet jedincov	Najlepší výsledok	Najhorší výsledok	Priemerný výsledok
10	0,00004147881034	1,179943883	0,6703491518

Algoritmus sa najviac priblížil hodnote minima pri nižšom počte jedincov a tak naďalej bude pracovať s 10 jedincami.

Algoritmus svätojánskych mušiek

Počet jedincov	Najlepší výsledok	Najhorší výsledok	Priemerný výsledok
10	15,37595263	126,9741266	54,27290411

Vzhľadom na výsledky sa v ďalších experimentoch pracuje s 10 jedincami.

Algoritmus opelovania kvetov

Počet jedincov	Najlepší výsledok	Najhorší výsledok	Priemerný výsledok
5000	0,08243606734	2,561233262	1,632101099

Algoritmus bol menej presný pri nižšom počte jedincov a preto v ďalších experimentoch pracuje s počtom jedincov 5000.

Algoritmus čiernych dier

Narozdiel od ostatných algoritmov, počet dimenzií bol nastavený na tri vzhľadom na vysokú presnosť výpočtu pri dvoch dimenziách.

Počet jedincov	Najlepší výsledok	Najhorší výsledok	Priemerný výsledok
100	0	0,00739612606	0

Experiment ukázal, že algoritmus dosahuje veľmi presné výsledky aj pri nižšom počte iterácií, v ďalších experimentoch pracoval s hodnotou 100.

Druhý experiment

V druhom experimente som testovala schopnosť priblížiť sa hodnote minima pri výpočte cez viaceré dimenzie a to cez 2,3,4,5 dimenzií, prvý experiment pracoval s 2 dimenziami (s výnimkou BH).

Netopierí algoritmus

Počet dimenzií	Najlepší výsledok	Najhorší výsledok	Priemerný výsledok
2	0,0005589468703	2,300159269	1,160290399

Ako je vidieť z tabuľky, algoritmus bol menej presný pri vyššom počte, v ďalších experimentoch algoritmus pracuje s dvomi dimenziami.

Algoritmus svätojánskych mušiek

Počet dimenzií	Najlepší výsledok	Najhorší výsledok	Priemerný výsledok
2	3,92034734	140,0984128	56,79547146

Experiment ukázal, že algoritmus pri hľadaní minima, pri väčšom počte dimenzií je značne problémový, v ďalších experimentoch sa pracuje s dvomi dimenziami.

Algoritmus opelovania kvetov

Počet dimenzií	Najlepší výsledok	Najhorší výsledok	Priemerný výsledok
2	0,1253299059	3,264515564	1,685323553

Ako je vidieť z tabuľky, algoritmus bol menej presný s vyšším počtom dimenzií ale aj napriek tomu sa dokázal priblížiť k minimu, v ďalších experimentoch algoritmus pracuje s dvomi dimenziami.

Algoritmus čiernych dier

Počet dimenzií	Najlepší výsledok	Najhorší výsledok	Priemerný výsledok
3	0	0,0000000001458708709	0

Algoritmus ukázal veľmi presné výsledky aj pri výpočte cez viaceré dimenzie, ale v ďalších experimentoch pracuje s 3 dimenziami.

Tretí experiment

Účelom tretieho experimentu bolo porovnať presnosť s rastúcim počtom iterácií a to 100, 200, 500 a 1000.

Netopierí algoritmus

Počet iterácií	Najlepší výsledok	Najhorší výsledok	Priemerný výsledok
100	0,1156676058	1,99020221	1,120416678

Experiment ukázal, že s rastúcim počtom iterácií nedochádza k výraznému zlepšeniu výpočtu a preto v ďalších experimentoch pracujeme s maximálnym počtom iterácií 100.

Algoritmus svätojánskych mušiek

Počet iterácií	Najlepší výsledok	Najhorší výsledok	Priemerný výsledok
500	1,082763848	110,6537168	36,767079

Presnosť algoritmu sa zvyšovala s rastúcim počtom iterácií, v ďalších experimentoch sa pracuje s 500 iteráciami.

Algoritmus opelovania kvetov

Počet iterácií	Najlepší výsledok	Najhorší výsledok	Priemerný výsledok
1000	0,2243844672	2,385584955	1,432431127

Ako najúspešnejší sa ukázal počet iterácií nastavený na 1000 a s touto hodnotou algoritmus pracoval aj v ďalších experimentoch.

Algoritmus čiernych dier

Počet iterácií	Najlepší výsledok	Najhorší výsledok	Priemerný výsledok
100	0	0,00001133032283	0,000001133032337

Na dosiahnutie veľmi presného výsledku stačil aj veľmi nízky počet iterácií a to 100.

Štvrtý experiment

Cieľom štvrtého experimentu bolo zistiť aký vplyv má na výpočet hodnota parametru beta, testované hodnoty boli: 0.1, 0.3, 0.5, 0.7 a 0.9 pre BA, a 0.5, 1, 1.5 a 5 pre FFA, experiment prebiehal len pri netopiérom algoritme a algoritme svätojánskych mušiek.

Netopierí algoritmus

Hodnota beta	Najlepší výsledok	Najhorší výsledok	Priemerný výsledok
0,1	0,08194764076	1,859249461	1,146775521

Ako najvhodnejšia sa ukázala hodnota parametru beta 0.1 s ktorou sa pracovalo v ďalších experimentoch.

Algoritmus svätojánskych mušiek

Hodnota beta	Najlepší výsledok	Najhorší výsledok	Priemerný výsledok
1	1,993767247	75,54723842	24,03074866

Ako najvhodnejšia sa ukázala hodnota 1 s ktorou sa pracovalo v ďalších experimentoch.

Piaty experiment

Piaty experiment mal zistiť aký vplyv má na výpočet hodnota parametrov A a r, pričom ich hodnoty boli rovnaké, testované hodnoty boli: 0.1, 0.3, 0.5, 0.7 a 0.9 experiment prebiehal len na netopiérom algoritme.

Netopierí algoritmus

Hodnota A a r	Najlepší výsledok	Najhorší výsledok	Priemerný výsledok
0,1	0,003141218675	2,472350843	1,040038565

Ako najlepšia sa ukázala hodnota 0.1 s ktorou sa pracuje v ďalších experimentoch.

Šiesty experiment

Šiesty experiment mal zistiť aký vplyv má na výpočet hodnota parametrov alfa a gamma, pričom ich hodnoty boli rovnaké, testované hodnoty boli: 0.1, 0.3, 0.5, 0.7 a 0.9 (táto hodnota nebola testovaná pri FFA). Testovanie prebehlo na netopiérom algoritme a algoritme svätojánskych mušiek.

Netopierí algoritmus

Hodnota alfa a gamma	Najlepší výsledok	Najhorší výsledok	Priemerný výsledok
0,9	0,0004974598076	1,675150029	0,8620714337

Ako najlepšia sa ukázala hodnota 0.9 s ktorou sa pracuje v ďalších experimentoch.

Algoritmus svätojánskych mušiek

Hodnota alfa a gamma	Najlepší výsledok	Najhorší výsledok	Priemerný výsledok
0,1	0,9982200609	120,372184	35,29124584

Ako najlepšia hodnota sa ukázala hodnota 0.1.

Algoritmus svätojánskych mušiek piaty experiment časť druhá

Algoritmus mal problém, priblížiť sa minimu, avšak pri prehľadávaní na intervale <-1,1> sa už minimu priblížil.

Interval	Najlepší výsledok	Najhorší výsledok	Priemerný výsledok
<-1,1>	0,1780489718	1,341274138	2,831096128

Siedmy experiment

Siedmy experiment mal zistiť aký vplyv má na výpočet hodnota maximálnej frekvencie, testované hodnoty boli: 2, 4, 6, 8 a 10 testovanie prebehlo len na netopiérom algoritme.

Netopierí algoritmus

Ako najlepšia sa ukázala frekvencia s hodnotou 10.

Maximálna frekvencia	Najlepší výsledok	Najhorší výsledok	Priemerný výsledok
10	0,002331769642	1,992390323	0,7109570712

Ôsmy experiment

Cieľom ôsmeho experimentu bolo zistiť aký vplyv má na výpočet hodnota pravdepodobnosti prepnutia, testované hodnoty boli: 0.1, 0.3, 0.5, 0.7, 0.8, 0.9 a 1 experiment prebehol len na algoritme opeľovania kvetov.

Algoritmus opelovania kvetov

Parameter prepnutia	Najlepší výsledok	Najhorší výsledok	Priemerný výsledok
0,8	0,1191618748	1,716773984	1,17212104

Ako najvhodnejšia sa ukázala hodnota 0,8 s ktorou sa pracovalo v ďalších experimentoch.

Deviaty experiment

Cieľom deviateho experimentu bolo zistiť najlepšiu hodnotu parametru lambda, testované hodnoty boli 0.5, 1.5, 5 a 15 experiment prebehol len na algoritme opeľovania kvetov.

Algoritmus opelovania kvetov

Parameter lambda	Najlepší výsledok	Najhorší výsledok	Priemerný výsledok
5	0,4403386556	2,840347348	1,290710047

Ako najlepšia hodnota sa ukázala hodnota 5.

Zhrnutie

Ako najefektívnejší algoritmus sa ukázal algoritmus čiernych dier narozdiel od neho najmenej presný bol algoritmus svätojánskych mušiek.

5.3 Rosenbrockova funkcia

Globálne minimum Rosenbrockovej funkcie má hodnotu nula, v rámci testovania algoritmy pracovali na intervale <-10,10>. Predpis Rosenbrockovej funkcie je uvedený v 3.3.

Prvý experiment

Cieľom prvého experimentu bolo nájsť, taký počet jedincov, pre ktorý sa bude výsledok blížiť hodnote globálneho minima a zároveň nebude trvať príliš dlho. Počet jedincov bol testovaný na hodnotách 500, 1000 a 2000. FPA potreboval väčší počet jedincov a tak pracoval aj s hodnotou 5000 naopak BH pre jeho presnosť pracoval aj s menším počtom jedincov

a to: 10, 50, 100, 200 Na rozdiel od Griewankovej funkcie pri Rosenbrockovej funkcii prebiehali experimenty minimálne pri 3 dimenziách, nakoľko pri dvoch dimenziách algoritmus našiel globálne minimum bez ohľadu na počet jedincov.

Netopierí algoritmus

Počet jedincov	Najlepší výsledok	Najhorší výsledok	Priemerný výsledok
500	0,00544069109	5,456921114	1,90742419

Pri výpočte cez tri dimenzie už algoritmus potreboval väčší počet jedincov aby sa priblížil výsledkom, avšak pre zrýchlenie výpočtu a zistenie ideálnych hodnôt ďalších parametrov sa v ďalších experimentoch pracuje s populáciou o veľkosti 500 jedincov.

Algoritmus svätojánskych mušiek

Počet jedincov	Najlepší výsledok	Najhorší výsledok	Priemerný výsledok
500	59,0602898	781704,8838	277488,3518

Vzhľadom na výsledky sa v ďalších experimentoch pracuje s 500 jedincami.

Algoritmus opelovania kvetov

Počet jedincov	Najlepší výsledok	Najhorší výsledok	Priemerný výsledok
5000	0,01318356611	0,583191562	0,2883805278

Algoritmus značne nepresný s nižším počtom jedincov a tak v ďalších experimentoch pracuje s 5000 jedincami.

Algoritmus čiernych dier

Počet jedincov	Najlepší výsledok	Najhorší výsledok	Priemerný výsledok
500	0	0,0004337343662	0,0000851810399

Experiment ukázal, že algoritmus dosahuje veľmi presné výsledky aj pri nižšom počte iterácií, v ďalších experimentoch pracoval s hodnotou 500.

Druhý experiment

V druhom experimente som testovala schopnosť priblížiť sa hodnote minima pri výpočte cez viaceré dimenzie a to cez 2, 3, 4, 5 dimenzií, FPA algoritmus nebol testovaný na druhej strane BH pre jeho úspešnosť pri hľadaní minima bol testovaný aj pri výpočte cez 6 dimenzií ale nebol testovaný pre 2 dimenzie nakoľko dosiahol globálne minimum bez ohľadu na počet jedincov.

Netopierí algoritmus

Počet dimenzií	Najlepší výsledok	Najhorší výsledok	Priemerný výsledok
3	0,4113228428	11,56095239	4,327925627

Ako je vidieť z tabuľky, algoritmus mal pri vyššom počte dimenzií problém priblížiť sa minimu, vzhľadom na výsledky prvého experimentu môžme predpokladať, že s vyšším počtom jedincov by sa tento problém podarilo minimalizovať.

Algoritmus svätojánskych mušiek

Počet dimenzií	Najlepší výsledok	Najhorší výsledok	Priemerný výsledok
3	17,55244292	841458,8938	163451,0564

Experiment ukázal, že algoritmus pri hľadaní minima, pri väčšom počte dimenzií je značne problémový, v ďalších experimentoch sa pracuje s tromi dimenziami.

Algoritmus čiernych dier

Počet dimenzií	Najlepší výsledok	Najhorší výsledok	Priemerný výsledok
3	0	0,005396281866	0,0009892934476

Algoritmus ukázal veľmi presné výsledky aj pri výpočte cez viaceré dimenzie, ale v ďalších experimentoch pracuje s 3 dimenziami aby bolo jednoduchšie porovnať výsledky s ostatnými algoritmami.

Tretí experiment

Účelom tretieho experimentu bolo porovnať presnosť s rastúcim počtom iterácií a to 100, 200, 500 a 1000.

Netopierí algoritmus

Počet iterácií	Najlepší výsledok	Najhorší výsledok	Priemerný výsledok
200	0,006019378884	2,567201751	1,326577626

Z výsledkov je vidieť, že počet iterácií nemá až taký veľký vplyv na výsledok ako počet jedincov, v ďalších experimentoch sa pracuje s počtom iterácií 200, nakoľko bol algoritmus pre túto hodnotu najbližšie k výsledku.

Algoritmus svätojánskych mušiek

Počet iterácií	Najlepší výsledok	Najhorší výsledok	Priemerný výsledok
1000	0,7710232172	144707,0489	14513,93728

Presnosť algoritmu sa zvyšovala s rastúcim počtom iterácií, v ďalších experimentoch sa pracuje s 1000 iteráciami.

Algoritmus opelovania kvetov

Počet iterácií	Najlepší výsledok	Najhorší výsledok	Priemerný výsledok
500	0,05398263936	0,4854301325	0,1722102941

Ako najúspešnejší sa ukázal počet iterácií nastavený na 500 a s touto hodnotou algoritmus pracoval aj v ďalších experimentoch.

Algoritmus čiernych dier

Počet iterácií	Najlepší výsledok	Najhorší výsledok	Priemerný výsledok
1000	0	0,000000006286430644	0,0000000006313666891

Ako najúspešnejší sa ukázal počet iterácií nastavený na 1000 ale algoritmus dosiahol veľmi dobré výsledky aj pri nižšom počte iterácií.

Štvrtý experiment

Cieľom štvrtého experimentu bolo zistiť aký vplyv má na výpočet hodnota parametru beta, testované hodnoty boli: 0.1, 0.3, 0.5, 0.7 a 0.9 pre BA a 0.5, 1, 1.5 a 5 pre FFA, experiment prebiehal len pri netopiérom algoritme a algoritme svätojánskych mušiek.

Netopierí algoritmus

Hodnota beta	Najlepší výsledok	Najhorší výsledok	Priemerný výsledok
0,1	0,0004046719806	1,856229405	0,8065301178

Ako najvhodnejšia sa ukázala hodnota parametru beta 0.1 s ktorou sa pracovalo v ďalších experimentoch.

Algoritmus svätojánskych mušiek

Ako najvhodnejšia sa ukázala hodnota 1 s ktorou sa pracovalo v ďalších experimentoch.

Hodnota beta	Najlepší výsledok	Najhorší výsledok	Priemerný výsledok
1	0,1731417759	209,6845209	65,85160709

Piaty experiment

Piaty experiment mal zistiť aký vplyv má na výpočet hodnota parametrov A a r, pričom ich hodnoty boli rovnaké, testované hodnoty boli: 0.1, 0.3, 0.5, 0.7 a 0.9 a testovaný bol len BA.

Netopierí algoritmus

Hodnota A a r	Najlepší výsledok	Najhorší výsledok	Priemerný výsledok
0,3	0,0000921240995	1,511773921	0,361683764

Ako najlepšia sa ukázala hodnota 0.3 s ktorou sa pracuje v ďalších experimentoch.

Šiesty experiment

Šiesty experiment mal zistiť aký vplyv má na výpočet hodnota parametrov alfa a gamma, pričom ich hodnoty boli rovnaké, testované hodnoty boli: 0.1, 0.3, 0.5, 0.7 a 0.9. Testovanie prebehlo na netopiérom algoritme a algoritme svätojánskych mušiek.

Netopierí algoritmus

Hodnota alfa a gamma	Najlepší výsledok	Najhorší výsledok	Priemerný výsledok
0,9	0,0008430305235	4,29390056	1,163204017

Ako najlepšia sa ukázala hodnota 0.9 s ktorou sa pracuje v ďalších experimentoch.

Algoritmus svätojánskych mušiek

Hodnota alfa a gamma	Najlepší výsledok	Najhorší výsledok	Priemerný výsledok
0,9	2,284901764	495476,331	99629,07211

Ako najlepšia hodnota sa ukázala hodnota 0.9.

Siedmy experiment

Siedmy experiment mal zistiť aký vplyv má na výpočet hodnota maximálnej frekvencie, pričom ich hodnoty boli rovnaké, testované hodnoty boli: 2, 4, 6, 8 a 10, pričom testovanie prebehlo len na BA.

Netopierí algoritmus

Ako najlepšia sa ukázala frekvencia s hodnotou 6.

Maximálna frekvencia	Najlepší výsledok	Najhorší výsledok	Priemerný výsledok
6	0,06311679124	2,90135066	1,121402346

Ôsmy experiment

Cieľom ôsmeho experimentu bolo zistiť aký vplyv má na výpočet hodnota pravdepodobnosti prepnutia, testované hodnoty boli: 0.1, 0.3, 0.5, 0.7 a 0.9 experiment prebehol len na algoritme opeľovania kvetov.

Algoritmus opelovania kvetov

Parameter prepnutia	Najlepší výsledok	Najhorší výsledok	Priemerný výsledok
0,3	0,02295269608	0,2339860031	0,1752899388

Ako najvhodnejšia sa ukázala hodnota 0.3 s ktorou sa pracovalo v ďalších experimentoch.

Deviaty experiment

Cieľom deviateho experimentu bolo zistiť najlepšiu hodnotu parametru lambda, testované hodnoty boli 0.5, 1, 1.5 a 2 experiment prebehol len na algoritme opeľovania kvetov.

Algoritmus opelovania kvetov

Parameter lambda	Najlepší výsledok	Najhorší výsledok	Priemerný výsledok
1	0,00087581596	0,5719110734	0,1514592248

Ako najlepšia hodnota sa ukázala hodnota 1.

Zhrnutie

Ako najefektívnejší algoritmus sa ukázal algoritmus čiernych dier narozdiel od neho algoritmus svätojánskych mušiek bol takmer nepoužiteľný pri výpočte cez viac ako dve dimenzie.

Kapitola 6

Záver

V tejto práci bolo cieľom zvoliť si niekoľko optimalizačných algoritmov inšpirovaných prírodou, preštudovať ich, navrhnúť program na demonštrovanie ich činnosti a tento program následne naimplementovať. Ďalej si vhodne zvoliť spôsob akým bude otestovaná úspešnosť daných algoritmov, vykonať na nich experimenty a určiť úspešnosť algoritmov.
Ako algoritmy boli zvolené netopierí algoritmus a algoritmus svätojánskych mušiek, ktoré zaraďujeme do kategórie algoritmov založených na inteligencii skupiny, algoritmus opelovania kvetín z kategórie algoritmov inšpirovaných biológiou a algoritmus čiernych dier z algoritmov inšpirovaných chemickými a fyzikálnymi javmi. Ako spôsob testovania úspešnosti daných algoritmov bol zvolený problém hľadania extrému funkcie, konrétne snaha nájsť minimum na Griewankovej, Rastringinovej a Rosenbrockovej funkcii.
Pri hľadaní minima Griewankovej funkcie sa všetky algoritmy dokázali značne priblížiť hodnote minima. Avšak ako najúspešnejší sa ukázal BH algoritmus, ktorý aj pri relatívne malom počte jedincov a iterácií sa nielenže dokázal priblížiť hodnote globálneho minima, v niektorých prípadoch ho aj našiel. Za dosť úspešný môžeme považovať aj BA, ktorý sa rovnako ako BH dokázal priblížiť minimu aj pri nižšom počte iterácií a jedincov. FFA a FPA algoritmus sa minimu tiež dokázali priblížiť veľmi úspešne avšak potrebovali vyšší počet iterácií, prípadne jedincov ako BH alebo BA.
Pri testovaní pomocou Rastringinovej funkcie boli rozdiely medzi úspešnosťou algoritmov vidieť zreteľnejšie. Ako najúspešnejší sa ukázal BH algoritmus avšak pri tejto funkcii narozdiel od ostatných algoritmov musel pracovať pri výpočte cez tri dimenzie, pretože pri výpočte cez dve dimenzie našiel globálne minimum bez ohľadu na hodnoty jeho parametrov, jeho úspešnosť bola veľmi vysoká aj pri výpočte cez viaceré dimenzie, BH sa bol schopný priblížiť minimu pri výpočte cez tri dimenzie viac ako zvyšné tri algoritmy ktoré pracovali len s dvomi dimenziami. BA sa minimu dokázal priblížiť najviac zo zvyšných troch algoritmov ale nedosahoval takú úspešnosť ako pri Griewankovej funkcii. FPA algoritmus sa tiež ešte dokázal priblížiť minimu ale bol potrebný vysoký počet iterácií a jedincov, čo sa odzrkadlilo na rýchlosti výpočtu. FFA sa ukázal ako veľmi neúspešný pri hľadaní minima.
Pri testovaní na Rosenbrockovej funkcii všetky algoritmy našli hodnotu globálneho minima pri testovaní cez dve dimenzie preto experimenty prebiehali pri testovaní na troch dimenziách. Ako najúspešnejší sa aj v tomto prípade ukázal BH, ktorý bol schopný nájsť aj hodnotu globálneho minima a nielen sa jej priblížiť. BA sa minimu dokázal úspešne priblížiť avšak bol potrebný vyšší počet iterácií aj jedincov. FPA dosiahol úspešnejšie výsledky pri hľadaní avšak počet jedincov, ktorý bol použitý bol značne vyšší ako pri BA, na základe priebehu experimentov môžme predpokladať, že pri rovnakom počte jedincov by bol BA oveľa úspešnejší. FFA robilo aj pri tejto funkcii značný problém priblížiť sa minimu funkcie.

Zaujímavým zistením bolo, že presnosť výpočtu stúpala s vyšším počtom jedincov takmer pri všetkých funkciách a algoritmoch okrem FFA pri ktorom najväčší vplyv mal počet iterácií a presnejšie výsledky dosahoval narozdiel od ostatných pri nižšom počte jedincov.

Literatúra

[1] *Separable functions* [online]. Dostupné z: http://www.optique-ingenieur.org/en/courses/OPI_ang_M02_C01/co/Contenu_03.html,cited=.

[2] CONTRIBUTORS, W. *Animal echolocation* [online]. [cit. 2019-10-28]. Dostupné z: https://en.wikipedia.org/wiki/Animal_echolocation.

[3] CONTRIBUTORS, W. *Bat* [online]. [cit. 2019-10-28]. Dostupné z: https://en.wikipedia.org/wiki/Bat.

[4] CONTRIBUTORS, W. *Black hole* [online]. [cit. 2019-10-28]. Dostupné z: https://en.wikipedia.org/wiki/Black_hole.

[5] CONTRIBUTORS, W. *Dopplerov jav* [online]. [cit. 2019-10-28]. Dostupné z: https://sk.wikipedia.org/wiki/Dopplerov_jav.

[6] CONTRIBUTORS, W. *Echolocation* [online]. [cit. 2019-10-28]. Dostupné z: https://www.sciencedirect.com/science/article/pii/S096098220500686X?

[7] CONTRIBUTORS, W. *Firefly* [online]. [cit. 2020-03-28]. Dostupné z: https://en.wikipedia.org/wiki/Firefly.

[8] CONTRIBUTORS, W. *Friedman test* [online]. [cit. 2019-10-28]. Dostupné z: https://en.wikipedia.org/wiki/Friedman_test.

[9] CONTRIBUTORS, W. *Holm–Bonferroni method* [online]. [cit. 2019-10-28]. Dostupné z: https://en.wikipedia.org/wiki/Holm\T1\textendashBonferroni_method.

[10] CONTRIBUTORS, W. *Lévy flight* [online]. [cit. 2019-10-28]. Dostupné z: https://en.wikipedia.org/wiki/L%C3%A9vy_flight.

[11] CONTRIBUTORS, W. *Megabat* [online]. [cit. 2019-10-20]. Dostupné z: https://en.wikipedia.org/wiki/Megabat.

[12] CONTRIBUTORS, W. *Microbat* [online]. [cit. 2019-10-28]. Dostupné z: https://en.wikipedia.org/wiki/Microbat.

[13] CONTRIBUTORS, W. *Pollination* [online]. [cit. 2019-10-28]. Dostupné z: https://en.wikipedia.org/wiki/Pollination.

[14] CONTRIBUTORS, W. *Random walk* [online]. [cit. 2019-10-28]. Dostupné z: https://en.wikipedia.org/wiki/Random_walk.

[15] CONTRIBUTORS, W. *Čierna diera* [online]. [cit. 2019-10-28]. Dostupné z: https://sk.wikipedia.org/wiki/%C4%8Cierna_diera.

[16] Hassanien, A. E. a Emary, E. *Swarm intelligence: principles, advances, andapplications*. CRC Press, 2016. ISBN 978-1-4987-4106-4.

[17] Hatamlou, A. *Black hole: A new heuristic optimization approach for data clustering* [online]. Information Sciences, 2013 [cit. 2019-10-28]. Dostupné z: https://www.sciencedirect.com/science/article/pii/S0020025512005762.

[18] Mazhar, A. *Griewank Function* [online]. [cit. 2020-04-28]. Dostupné z: http://benchmarkfcns.xyz/benchmarkfcns/griewankfcn.html.

[19] Mazhar, A. *Rastrigin Function* [online]. [cit. 2020-04-28]. Dostupné z: http://benchmarkfcns.xyz/benchmarkfcns/rastriginfcn.html.

[20] Mazhar, A. *Rosenbrock Function* [online]. [cit. 2020-04-28]. Dostupné z: http://benchmarkfcns.xyz/benchmarkfcns/rosenbrockfcn.html.

[21] Surjanovic, D. *Griewank Function* [online]. [cit. 2020-04-28]. Dostupné z: https://www.sfu.ca/~ssurjano/griewank.html.

[22] Surjanovic, D. *Rastrigin Function* [online]. [cit. 2020-04-28]. Dostupné z: https://www.sfu.ca/~ssurjano/rastr.html.

[23] Surjanovic, D. *Rosenbrock Function* [online]. [cit. 2020-04-28]. Dostupné z: https://www.sfu.ca/~ssurjano/rosen.html.

[24] Wikipedie, P. *Diferencovatelnost* [online]. Dostupné z: https://cs.wikipedia.org/wiki/Diferencovatelnost,cited=.

[25] Yang, X.-S. *Firefly Algorithms for Multimodal Optimization* [online]. [cit. 2020-03-28]. Dostupné z: https://arxiv.org/pdf/1003.1466.pdf?fbclid=IwAR06NX93hbWPT67V_AxGwRmEnR9bXmWIZ5GbGsoVDerx38bowDDwuW6PfTw.

[26] Yang, X.-S. *A New Metaheuristic Bat-Inspired Algorithm* [online]. Nature Inspired Cooperative Strategies for Optimization, 2010 [cit. 2019-10-28]. Dostupné z: https://arxiv.org/pdf/1004.4170.pdf.

[27] Yang, X.-S. *Flower pollination algorithm for global optimization* [online]. Unconventional Computation and Natural Computation 2012, 2012 [cit. 2019-10-28]. Dostupné z: https://arxiv.org/pdf/1312.5673.pdf.

[28] Zuniga, R. *Bats Usage of Echolocation* [online]. [cit. 2019-10-28]. Dostupné z: https://www.sticksandstonesrescue.org/news/2018/4/15/weekly-article-bats-usage-of-echolocation.

Príloha A

Výsledky experimentov

A.1 Griewankova funkcia

A.1.1 Prvý experiment

Netopierí algoritmus

Počet jedincov	Najlepší výsledok	Najhorší výsledok	Priemerný výsledok
1	0,03036685105	0,9577428442	0,4104210829
10	0,03521979624	0,4512588682	0,1610618123
50	0,0009336806186	0,1094392288	0,04332232299
100	0,01337955943	0,04788540741	0,0318446504
200	0,007573102577	0,01723131613	0,03213877634
500	0,008128512621	0,03303233419	0,01484846767
1000	0,006574213015	0,02245122647	0,0094342877
2000	0,0003838300946	0,008284422751	0,006476435833

Algoritmus svätojánskych mušiek

Počet jedincov	Najlepší výsledok	Najhorší výsledok	Priemerný výsledok
1	0,08012630767	1,967677138	1,115036593
10	0,01433742719	1,338519517	0,6135962388
50	0,04158226446	1,614561402	0,726505048
100	0,2136809793	1,60701397	0,7633799041
200	0,05883939907	1,307510026	0,7103925461
500	0,0908446413	1,956546983	0,9132847336
1000	0,3877317692	1,484553132	1,077680552
2000	0,8596368163	1,793138828	1,271952318

Algoritmus opelovania kvetov

Počet jedincov	Najlepší výsledok	Najhorší výsledok	Priemerný výsledok
2	0,06689944238	1,370973677	0,7221008571
10	0,08243606734	0,7819488955	0,2891709477
50	0,01761403036	0,2527536204	0,07590693134
100	0,003745511614	0,06878528537	0,03577513219
200	0,00572003753	0,03610711588	0,02167667846
500	0,00764675031	0,03218509109	0,01653232427
1000	0,003057333051	0,02113675788	0,01107818663
2000	0,08243606734	0,01765884434	0,008574568759

Algoritmus čiernych dier

Počet jedincov	Najlepší výsledok	Najhorší výsledok	Priemerný výsledok
2	0,07164816222	1,221438023	0,6312538717
10	0,000007341842613	0,01832790159	0,006911927502
50	0	0,00986536995	0,003945047959
100	0	0,00739612606	0,001479220966
200	0	0,007396053994	0,00147920946
500	0	0,007396040673	0,0007396046965
1000	0	0	0
2000	0	0	0

A.1.2 Druhý experiment

Netopierí algoritmus

Počet dimenzií	Najlepší výsledok	Najhorší výsledok	Priemerný výsledok
2	0,007692743786	0,0403162668	0,01828576928
3	0,005669060574	0,1260445736	0,05960570579
4	0,09668130793	0,3627509576	0,2113962306
5	0,1425619407	0,6128282781	0,3225496405

Algoritmus svätojánskych mušiek

Počet dimenzií	Najlepší výsledok	Najhorší výsledok	Priemerný výsledok
2	0,02287229099	0,9990734594	0,5197493197
3	0,4498292895	1,207595534	0,9003484523
4	0,5441658581	1,609420742	0,9486947545
5	0,9390924025	1,605830821	1,120737668

Algoritmus opelovania kvetov

Počet dimenzií	Najlepší výsledok	Najhorší výsledok	Priemerný výsledok
2	0,008365510621	0,02756232835	0,01894701647
3	0,05319492262	0,1248100928	0,08301344733
4	0,06497520043	0,268710733	0,1291176186
5	0,08466455794	0,3815987372	0,271485496

Algoritmus čiernych dier

Počet dimenzií	Najlepší výsledok	Najhorší výsledok	Priemerný výsledok
2	0	0,007396043327	0,005177543468
3	0	0,01232100003	0,006574875259
4	0	0,03203677552	0,01527682922
5	0,009857287332	0,04187995035	0,01749103516

A.1.3 Tretí experiment

Netopierí algoritmus

Počet iterácií	Najlepší výsledok	Najhorší výsledok	Priemerný výsledok
100	0,00996216925	0,02061282859	0,0226013252
200	0,000179653844	0,02369155965	0,02369155965
500	0,0001469100805	0,02033854005	0,02155212718
1000	0,00841365552	0,02521497027	0,02573985929

Algoritmus svätojánskych mušiek

Počet iterácií	Najlepší výsledok	Najhorší výsledok	Priemerný výsledok
100	0,0003793750137	1,788675456	0,6236762267
200	0,008165792686	1,919954995	0,4121564506
500	0,0005653840749	1,773418448	0,2010488737
1000	0,0000008975443623	0,7762232803	0,1555533201

Algoritmus opelovania kvetov

Počet iterácií	Najlepší výsledok	Najhorší výsledok	Priemerný výsledok
100	0,008480625488	0,02603366671	0,01388620394
200	0,008867932903	0,03674071533	0,01831245856
500	0,003124347856	0,02428250632	0,01175231318
1000	0,009053095333	0,03030077079	0,01744011436

Algoritmus čiernych dier

Počet iterácií	Najlepší výsledok	Najhorší výsledok	Priemerný výsledok
100	0	0,007396040336	0,0007396040336
200	0	0,007396043887	0,0007396043887
500	0	0,00739604196	0,000739604196
1000	0	0	0

A.1.4 Štvrtý experiment

Netopierí algoritmus

Hodnota beta	Najlepší výsledok	Najhorší výsledok	Priemerný výsledok
0,1	0,01574158849	0,04190893279	0,02768874659
0,3	0,00991705896	0,02277137025	0,01356417137
0,5	0,002997797375	0,03820559497	0,01916418376
0,7	0,007051517104	0,03574562505	0,0192613311
0,9	0,00004100751022	0,03082173493	0,01543653649

Algoritmus svätojánskych mušiek

Hodnota beta	Najlepší výsledok	Najhorší výsledok	Priemerný výsledok
0,1	0,007531738146	1,450655284	0,2205282525
0,3	0,001328144919	1,283068829	0,1537995309
0,5	0,0002126575541	0,03924737346	0,0168604054
0,7	0,007398575001	0,04577264997	0,02314113291
0,9	0,007478234568	0,2744357831	0,04652499082

A.1.5 Piaty experiment

Netopierí algoritmus

Hodnota A a r	Najlepší výsledok	Najhorší výsledok	Priemerný výsledok
0,1	0,003966785613	0,05128240854	0,02188882998
0,3	0,003528222372	0,03085564339	0,02356482891
0,5	0,002973990697	0,03346287829	0,01966319084
0,7	0,01218740669	0,06470842586	0,02912040092
0,9	0,01303006132	0,04649762442	0,02615937243

A.1.6 Šiesty experiment

Netopierí algoritmus

Hodnota alfa a gamma	Najlepší výsledok	Najhorší výsledok	Priemerný výsledok
0,1	0,01035778869	0,08687053235	0,02952791213
0,3	0,00312502995	0,0582215137	0,02657765669
0,5	0,0005564188455	0,03024721729	0,01389304808
0,7	0,004148233705	0,0335003373	0,01653043516
0,9	0,009940789944	0,03756533786	0,02157474896

Algoritmus svätojánskych mušiek

Hodnota alfa a gamma	Najlepší výsledok	Najhorší výsledok	Priemerný výsledok
0,1	0,007401660845	0,2394120335	0,0626120308
0,3	0,00004866431964	1,625117192	0,3067666408
0,5	0,0004157221949	2,005153209	0,5264491126
0,7	0,007634974084	1,587787092	0,4460746641
0,9	0,008899427787	1,017699158	0,5425111724

A.1.7 Siedmy experiment

Netopierí algoritmus

Maximálna frekvencia	Najlepší výsledok	Najhorší výsledok	Priemerný výsledok
2	0,002169135046	0,05064638184	0,02291506882
4	0,007913388527	0,04844867307	0,02135231928
6	0,007435026664	0,02554925514	0,01458289157
8	0,00773317434	0,06910410057	0,02213897488
10	0,007675797173	0,04611359435	0,02377384339

A.1.8 Ôsmy experiment

Algoritmus opelovania kvetov

Parameter prepnutia	Najlepší výsledok	Najhorší výsledok	Priemerný výsledok
0,1	0,006423951033	0,03342094775	0,02103076855
0,3	0,006479058727	0,03983070412	0,01843095192
0,5	0,007893794827	0,03786082396	0,01649626589
0,7	0,0154652246	0,03861669039	0,02294350783
0,9	0,000002746526098	0,04411784764	0,02158785012

A.1.9 Deviaty experiment

Algoritmus opelovania kvetov

Parameter lambda	Najlepší výsledok	Najhorší výsledok	Priemerný výsledok
0,5	0,008459490494	0,03056014138	0,01856707663
1,5	0,008790986406	0,03229093238	0,02109039446
5	0,004630787577	0,03023308173	0,01595172248
15	0,004499863727	00,02988289049	0,01969815423

A.2 Rastringinova funkcia

A.2.1 Prvý experiment

Netopierí algoritmus

Počet jedincov	Najlepší výsledok	Najhorší výsledok	Priemerný výsledok
10	0,00004147881034	1,179943883	0,6703491518
50	0,9992592447	133,1126657	42,70557676
100	4,999685712	202,9858914	76,22219446
200	1,00647315	97,70499229	36,77789931
500	0,1061528901	2,590558687	1,473384431
1000	8,892012007	147,2854811	64,1397204
2000	10,35295166	108,3691248	38,37829656
5000	13,53655727	92,70705394	51,65562884

Algoritmus svätojánskych mušiek

Počet jedincov	Najlepší výsledok	Najhorší výsledok	Priemerný výsledok
10	15,37595263	126,9741266	54,27290411
50	19,8580735	126,2592943	56,54958336
100	33,18237642	193,6100669	93,23078025
200	21,22452581	115,7848219	68,72166532
500	22,08203933	163,6598862	86,68793591
1000	17,54812183	168,452913	90,78476328
2000	20,56704074	117,3066877	71,24309303
5000	24,45164127	157,4901202	90,9238554

Algoritmus opelovania kvetov

Počet jedincov	Najlepší výsledok	Najhorší výsledok	Priemerný výsledok
10	8,571055234	36,22860917	23,07246115
50	2,800274593	25,04452425	14,94189017
100	1,889265973	15,84717015	9,216454404
200	3,131365647	13,89233669	7,90298125
500	2,036492062	8,111763168	4,509060316
1000	0,61748146	4,410585401	2,68868787
2000	0,785871526	6,097635846	2,756550787
5000	0,08243606734	2,561233262	1,632101099

Algoritmus čiernych dier

Počet jedincov	Najlepší výsledok	Najhorší výsledok	Priemerný výsledok
10	0,07820384242	19,8068377	4,549895595
50	0	1,446326654	0,2441314474
100	0	0,00739612606	0
200	0	0,0000000002349018757	0
500	0	0,007396040673	0
1000	0	0	0
2000	0	0,000000000873868089	0
5000	0	0	0

A.2.2 Druhý experiment

Netopierí algoritmus

Počet dimenzií	Najlepší výsledok	Najhorší výsledok	Priemerný výsledok
2	0,0005589468703	2,300159269	1,160290399
3	1,096766891	105,6964176	40,33616455
4	15,02707732	162,1141831	67,39414792
5	5,928411874	116,9620455	64,47626813

Algoritmus svätojánskych mušiek

Počet dimenzií	Najlepší výsledok	Najhorší výsledok	Priemerný výsledok
2	3,92034734	140,0984128	56,79547146
3	83,06162366	289,6098801	157,6515093
4	122,2162528	234,3604835	177,0908938
5	118,3611233	334,7996229	217,9940128

Algoritmus opelovania kvetov

Počet dimenzií	Najlepší výsledok	Najhorší výsledok	Priemerný výsledok
2	0,1253299059	3,264515564	1,685323553
3	4,640373415	10,95360127	8,122220587
4	12,32882466	26,98854593	20,07903693
5	29,59388723	40,39177732	34,32239582

Algoritmus čiernych dier

Počet dimenzií	Najlepší výsledok	Najhorší výsledok	Priemerný výsledok
2	0	0	0
3	0	0,0000000001458708709	0
4	0,0000000009654215205	1,989982655	0,8808015696
5	0,9949723869	1,990070896	1,500142605

A.2.3 Tretí experiment

Netopierí algoritmus

Počet iterácií	Najlepší výsledok	Najhorší výsledok	Priemerný výsledok
100	0,1156676058	1,99020221	1,120416678
200	17,01597231	155,2751028	81,4026904
500	33,14855856	150,6990394	73,11351161
1000	19,93694595	215,1202157	87,16883665

Algoritmus svätojánskych mušiek

Počet iterácií	Najlepší výsledok	Najhorší výsledok	Priemerný výsledok
100	2,059535012	117,8225077	57,01085469
200	5,470727092	132,9694727	63,63255508
500	1,082763848	110,6537168	36,767079
1000	1,018846196	116,5325029	50,16100481

Algoritmus opelovania kvetov

Počet iterácií	Najlepší výsledok	Najhorší výsledok	Priemerný výsledok
100	3,007877017	9,460547094	5,290007647
200	0,1395859498	6,863381138	3,458439312
500	1,044451483	5,861046889	2,617357056
1000	0,2243844672	2,385584955	1,432431127

Algoritmus čiernych dier

Počet iterácií	Najlepší výsledok	Najhorší výsledok	Priemerný výsledok
100	0	0,00001133032283	0,000001133032337
200	0	0	0
500	0	0	0
1000	0	0	0

A.2.4 Štvrtý experiment

Netopierí algoritmus

Hodnota beta	Najlepší výsledok	Najhorší výsledok	Priemerný výsledok
0,1	0,08194764076	1,859249461	1,146775521
0,3	6,60821496	143,5133427	60,2266255
0,5	17,34402815	102,8824808	57,44805934
0,7	6,82917707	101,7343157	60,21280196
0,9	0,524670109	2,141068477	1,241731501

Algoritmus svätojánskych mušiek

Hodnota beta	Najlepší výsledok	Najhorší výsledok	Priemerný výsledok
0,5	1,761882769	124,9220377	39,38313249
1	1,993767247	75,54723842	24,03074866
1,5	8,103109441	190,6507836	64,982576
5	2,941701208	100,6324296	36,13476274

A.2.5 Piaty experiment

Netopierí algoritmus

Hodnota A a r	Najlepší výsledok	Najhorší výsledok	Priemerný výsledok
0,1	0,003141218675	2,472350843	1,040038565
0,3	4,918551607	91,80632087	47,33774501
0,5	10,79095585	168,3787207	84,7487557
0,7	9,721757894	97,93133522	47,56448906
0,9	0,4786766659	2,233668314	1,244810211

A.2.6 Šiesty experiment

Netopierí algoritmus

Hodnota alfa a gamma	Najlepší výsledok	Najhorší výsledok	Priemerný výsledok
0,1	0,02914507029	1,991723036	0,9337321804
0,3	21,52455389	148,2950072	66,03424976
0,5	13,08632147	149,5849266	71,08549046
0,7	9,172364504	179,5011931	59,87960827
0,9	0,0004974598076	1,675150029	0,8620714337

Algoritmus svätojánskych mušiek

Hodnota alfa a gamma	Najlepší výsledok	Najhorší výsledok	Priemerný výsledok
0,1	0,9982200609	120,372184	35,29124584
0,3	17,91982298	159,5941224	67,12022524
0,5	2,427357408	122,5889351	53,03449043
0,7	10,26746023	155,2607023	70,83476803
0,9	5,114171205	148,933194	58,46473418

A.2.7 Siedmy experiment

Netopierí algoritmus

Maximálna frekvencia	Najlepší výsledok	Najhorší výsledok	Priemerný výsledok
2	0,1149951226	2,15717082	1,065635286
4	5,234798191	99,81708505	55,05760798
6	20,28290421	129,8285947	74,48743204
8	9,938684532	145,8678327	66,28053291
10	0,002331769642	1,992390323	0,7109570712

A.2.8 Ôsmy experiment

Algoritmus opelovania kvetov

Parameter prepnutia	Najlepší výsledok	Najhorší výsledok	Priemerný výsledok
0,1	0,6443330121	2,764874286	1,557006579
0,3	0,232491498	4,893737022	1,742561449
0,5	0,4515978579	3,390016633	1,867910497
0,7	0,4324358895	2,89935805	1,438103983
0,8	0,1191618748	1,716773984	1,17212104
0,9	0,6127750675	2,755680584	1,539330766
1	0,1432916805	2,732042975	1,351174179

A.2.9 Deviaty experiment

Algoritmus opelovania kvetov

Parameter lambda	Najlepší výsledok	Najhorší výsledok	Priemerný výsledok
0,5	0,3400770112	2,441433255	1,397982571
1,5	0,5816981696	1,919589665	1,403719852
5	0,4403386556	2,840347348	1,290710047
15	1,116075259	2,232103414	1,760571421

A.3 Rosenborckova funkcia

A.3.1 Prvý experiment

Netopierí algoritmus

Počet jedincov	Najlepší výsledok	Najhorší výsledok	Priemerný výsledok
500	0,00544069109	5,456921114	1,90742419
1000	0,003088723396	2,931050669	0,9578329733
2000	0,02743958827	0,971715888	0,297981346

Algoritmus svätojánskych mušiek

Počet jedincov	Najlepší výsledok	Najhorší výsledok	Priemerný výsledok
500	59,0602898	781704,8838	277488,3518
1000	57,80549067	960782,1505	313510,3443
2000	124,6341511	753464,7283	149019,2442

Algoritmus opelovania kvetov

Počet jedincov	Najlepší výsledok	Najhorší výsledok	Priemerný výsledok
500	0,07258646301	6,996952782	1,552907914
1000	0,08697596695	3,988699126	1,415249003
2000	0,0339733127	1,835224391	0,7351653693
5000	0,01318356611	0,583191562	0,2883805278

Algoritmus čiernych dier

Počet jedincov	Najlepší výsledok	Najhorší výsledok	Priemerný výsledok
10	0,001371526534	1,377218596	0,2756344539
50	0,00000007000614582	0,2928746247	0,05066751952
100	0,00001516341239	0,07016465021	0,03091905612
200	0	0,1518359463	0,02846435389
500	0	0,0004337343662	0,0000851810399
1000	0	0,002949083887	0,0003982997013
2000	0	0,0000005125256829	0,00000005923410414

A.3.2 Druhý experiment

Netopierí algoritmus

Počet dimenzií	Najlepší výsledok	Najhorší výsledok	Priemerný výsledok
2	0	0	0
3	0,4113228428	11,56095239	4,327925627
4	3,369829862	363,8562185	90,61665465
5	102,5773137	1760,95426	707,8168435

Algoritmus svätojánskych mušiek

Počet dimenzií	Najlepší výsledok	Najhorší výsledok	Priemerný výsledok
2	0	0	0
3	17,55244292	841458,8938	163451,0564
4	1982,214817	996207,8037	335571,6227
5	14091,16385	2238587,718	648666,6031

Algoritmus čiernych dier

Počet dimenzií	Najlepší výsledok	Najhorší výsledok	Priemerný výsledok
3	0	0,005396281866	0,0009892934476
4	0,00003559825247	0,4957157398	0,2048704723
5	0,08598968333	6,655306474	1,931854564
6	0,00543078658	6,387432534	2,192947207

A.3.3 Tretí experiment

Netopierí algoritmus

Počet iterácií	Najlepší výsledok	Najhorší výsledok	Priemerný výsledok
100	0,02378672488	7,500894952	1,523040985
200	0,006019378884	2,567201751	1,326577626
500	0,009838856701	5,480074388	1,395919847
1000	0,008862354551	6,840845569	1,82080004

Algoritmus svätojánskych mušiek

Počet iterácií	Najlepší výsledok	Najhorší výsledok	Priemerný výsledok
100	7911,49378	802154,5369	286349,0776
200	1107,649363	638369,5556	134722,8874
500	18,68581392	656854,1491	91501,76011
1000	0,7710232172	144707,0489	14513,93728

Algoritmus opelovania kvetov

Počet iterácií	Najlepší výsledok	Najhorší výsledok	Priemerný výsledok
100	0,04806151586	1,089880039	0,3608981943
200	0,0008909325347	1,572843	0,3170102107
500	0,05398263936	0,4854301325	0,1722102941
1000	0,05261942746	1,041131708	0,2647132906

Algoritmus čiernych dier

Počet iterácií	Najlepší výsledok	Najhorší výsledok	Priemerný výsledok
100	0	0,003167304391	0,0003594252052
200	0	0,0009887228199	0,0001236554515
500	0	0,00004431626016	0,000004958530976
1000	0	0,000000006286430644	0,0000000006313666891

A.3.4 Štvrtý experiment

Netopierí algoritmus

Hodnota beta	Najlepší výsledok	Najhorší výsledok	Priemerný výsledok
0,1	0,0004046719806	1,856229405	0,8065301178
0,3	0,0002031409339	13,4478893	2,266353626
0,5	0,01365392866	10,27550974	2,687620538
0,7	0,03537941375	5,867692448	1,745284567
0,9	0,004989692581	3,812328071	1,077987572

Algoritmus svätojánskych mušiek

Hodnota beta	Najlepší výsledok	Najhorší výsledok	Priemerný výsledok
0,5	0,2758843609	274785,9765	29632,0686
1	0,1731417759	209,6845209	65,85160709
1,5	0,3325919007	3241,556653	490,0589109
5	2,349772437	22644,66962	6054,271618

A.3.5 Piaty experiment

Netopierí algoritmus

Hodnota A a r	Najlepší výsledok	Najhorší výsledok	Priemerný výsledok
0,1	0,00646970308	10,48414964	1,983652083
0,3	0,0000921240995	1,511773921	0,361683764
0,5	0,00191875286	7,016943111	1,849522576
0,7	0,006466152069	3,758661506	1,403004597
0,9	0,1056793866	3,771042447	1,355388854

A.3.6 Šiesty experiment

Netopierí algoritmus

Hodnota alfa a gamma	Najlepší výsledok	Najhorší výsledok	Priemerný výsledok
0,1	0,01703699684	4,205656317	1,222418758
0,3	0,04092225338	7,130850039	1,889442958
0,5	0,0004116658041	4,094609222	1,5061976
0,7	0,01535037232	14,07681383	2,041310752
0,9	0,0008430305235	4,29390056	1,163204017

Algoritmus svätojánskych mušiek

Hodnota alfa a gamma	Najlepší výsledok	Najhorší výsledok	Priemerný výsledok
0,1	0,5004435565	6612,041349	984,6362662
0,3	6,599833556	549456,2343	61599,6176
0,5	1025,249628	863644,3696	174811,1375
0,7	63,78539836	1151360,706	136482,9032
0,9	2,284901764	495476,331	99629,07211

A.3.7 Siedmy experiment

Netopierí algoritmus

Maximálna frekvencia	Najlepší výsledok	Najhorší výsledok	Priemerný výsledok
2	0,003560985876	3,820628962	1,04071027
4	0,2583229472	2,473904946	1,611493896
6	0,06311679124	2,90135066	1,121402346
8	0,1080506208	4,083228416	1,508205611
10	0,01251479782	7,83874675	2,361056155

A.3.8 Ôsmy experiment

Algoritmus opelovania kvetov

Parameter prepnutia	Najlepší výsledok	Najhorší výsledok	Priemerný výsledok
0,1	0,0005262241095	0,8397998556	0,3437174106
0,3	0,02295269608	0,2339860031	0,1752899388
0,5	0,01147148986	0,9081898322	0,3184024476
0,7	0,001933466388	0,5507243164	0,1775961234
0,9	0,02231260743	1,103362921	0,3216220491

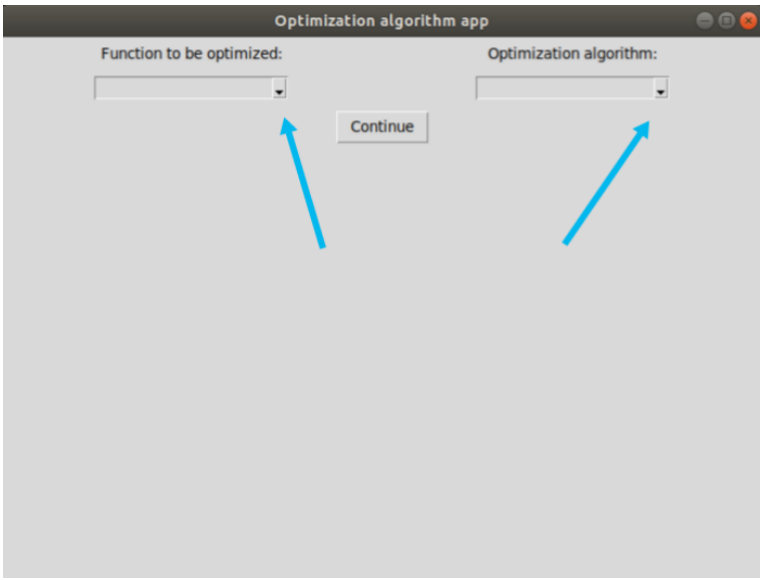
A.3.9 Deviaty experiment

Algoritmus opelovania kvetov

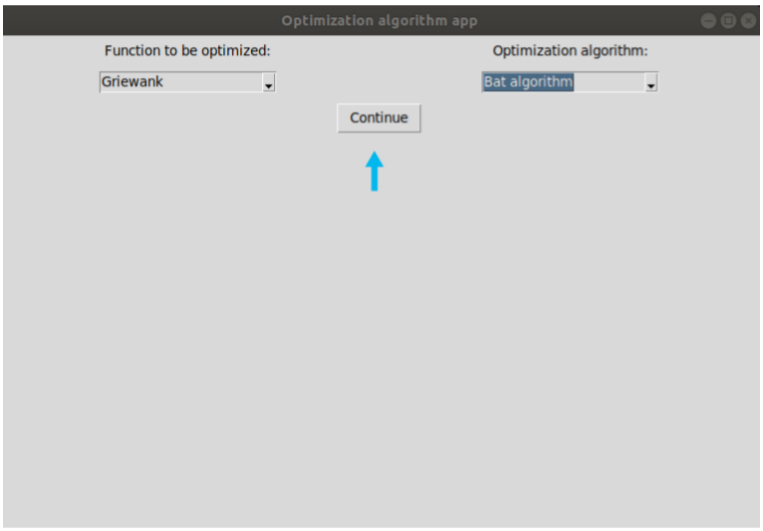
Parameter lambda	Najlepší výsledok	Najhorší výsledok	Priemerný výsledok
0,5	0,0105065552	0,9684225747	0,2320658687
1	0,00087581596	0,5719110734	0,1514592248
1,5	0,01117633878	0,5364332723	0,1740753892
2	0,0158615556	0,3291527846	0,1410825147

Príloha B

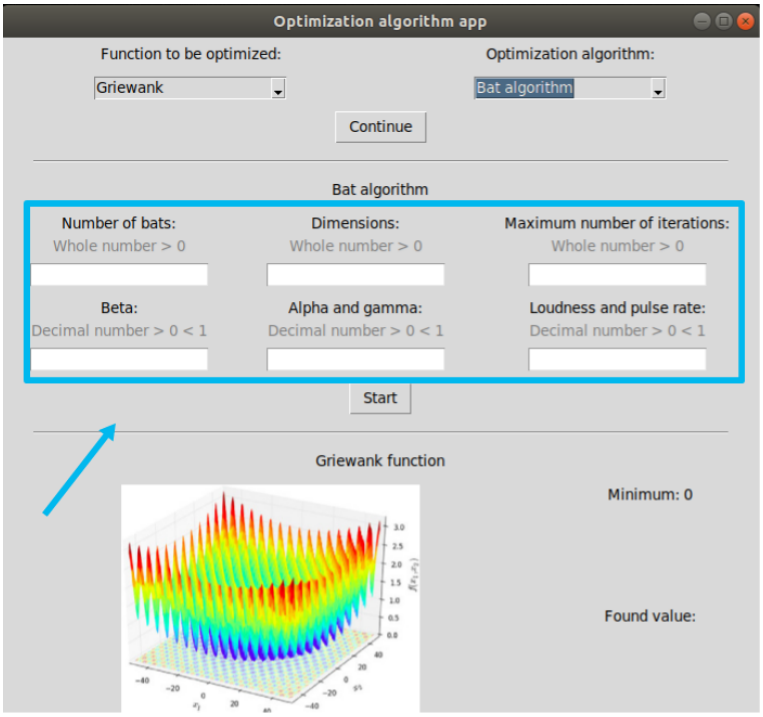
Obrázky uživateľského rozhrania



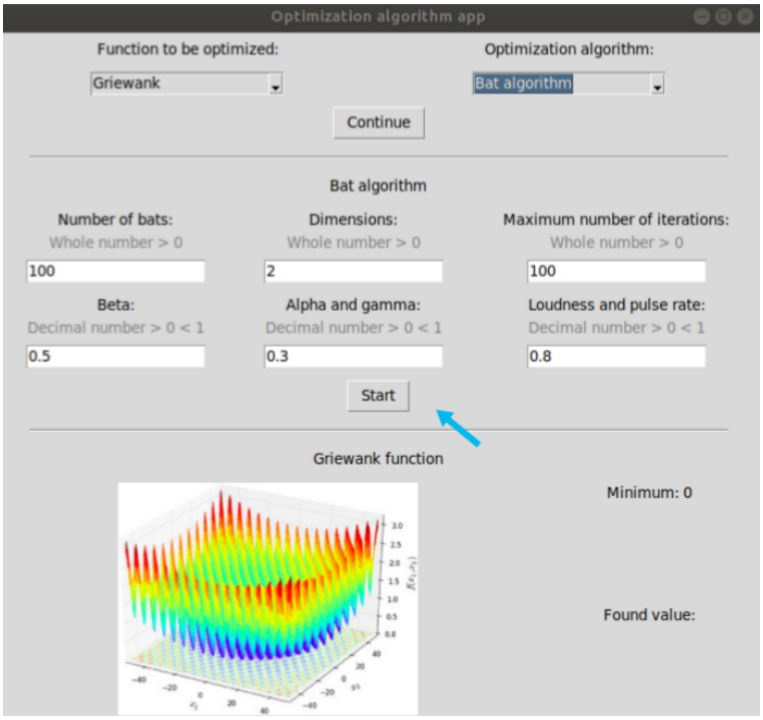
Obr. B.1: Úvodná obrazovka pri spustení aplikácie



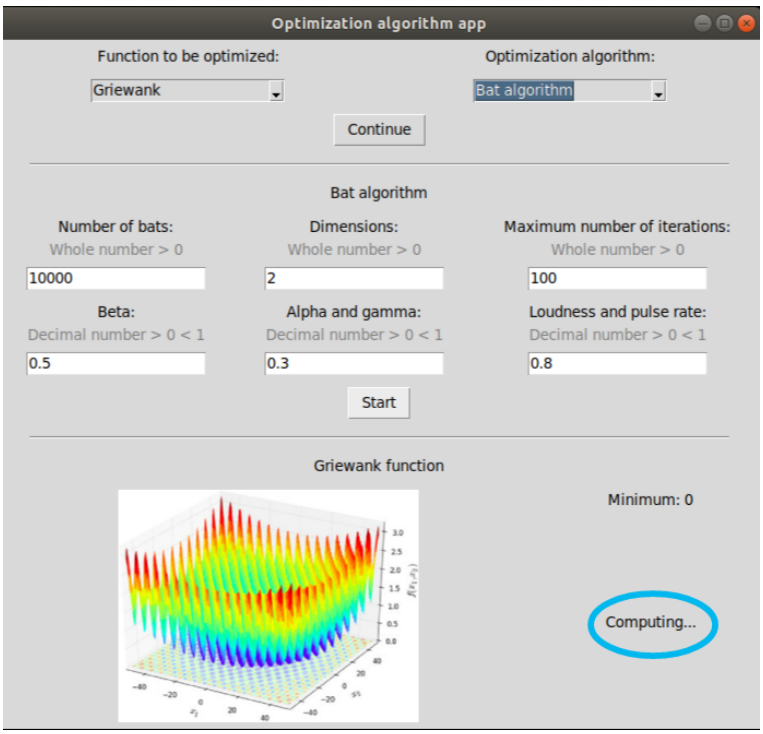
Obr. B.2: Obrazovka po zvolení optimalizačného algoritmu a testovacej funkcie



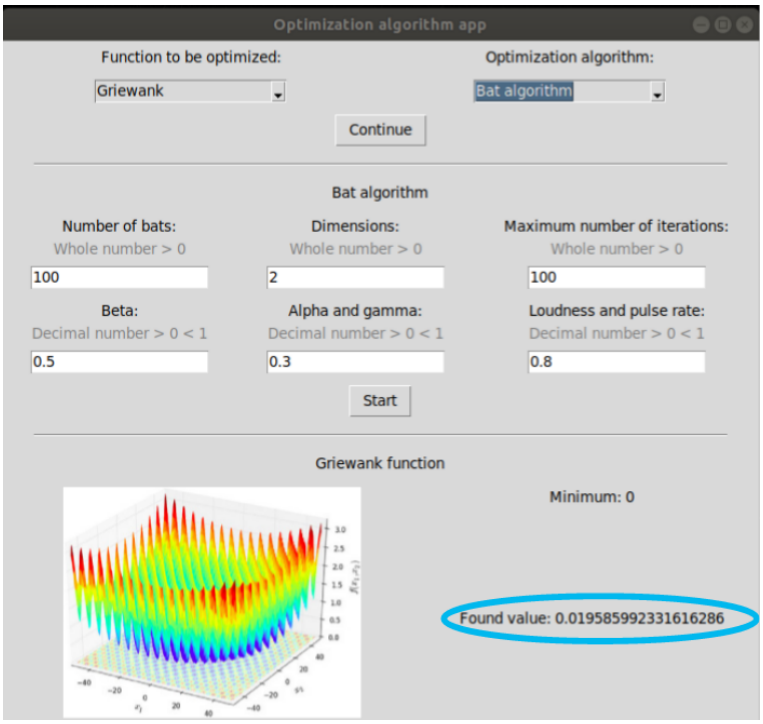
Obr. B.3: Obrazovka po stlačení tlačidla *Continue*



Obr. B.4: Obrazovka po vyplnení hodnôt



Obr. B.5: Obrazovka počas priebehu výpočtu



Obr. B.6: Výsledok výpočtu