VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA STROJNÍHO INŽENÝRSTVÍ
ÚSTAV AUTOMATIZACE A INFORMATIKY

FACULTY OF MECHANICAL ENGINEERING
INSTITUTE OF AUTOMATION AND COMPUTER SCIENCE

VČELÍ ALGORITMUS
BEES ALGORITHM

BAKALÁŘSKÁ PRÁCE
BACHELOR´S THESIS

AUTOR PRÁCE KAMIL MIŠKAŘÍK
AUTHOR

VEDOUCÍ PRÁCE Ing. RADOMIL MATOUŠEK Ph.D
SUPERVISOR

BRNO 2010

Vysoké učení technické v Brně, Fakulta strojního inženýrství

Ústav automatizace a informatiky
Akademický rok: 2009/2010

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

student(ka): Kamil Miškařík

který/která studuje v **bakalářském studijním programu**

obor: **Aplikovaná informatika a řízení (3902R001)**

Ředitel ústavu Vám v souladu se zákonem č.111/1998 o vysokých školách a se Studijním a zkušebním řádem VUT v Brně určuje následující téma bakalářské práce:

Včelí algoritmus

v anglickém jazyce:

Bees Algorithm

Stručná charakteristika problematiky úkolu:

Daná BP se bude zabývat implementaci včelího algoritmu (BA - Bees Algorithm, BCO - Bee Colony Optimization) a jeho využitím v oblasti globální optimalizace. Vývojovou platformou bude Java a Matlab.

Cíle bakalářské práce:

- Návrh vlastní Java třídy implementující včelí algoritmus.
- Návrh Matlab skriptu/funkce zpřístupňující danou implementaci algoritmu.
- Ověření výkonu včelího algoritmu na sadě standardních testovacích problémů.
- Dokumentace projektu (stručný popis BCO, popis vytvořených tříd a funkcí, příklady využití této optimalizační metody.

Seznam odborné literatury:

[1] Pham DT, Ghanbarzadeh A, Koc E, Otri S, Rahim S and Zaidi M. The Bees Algorithm. Technical Note, Manufacturing Engineering Centre, Cardiff University, UK, 2005

[2] Pham D.T., Ghanbarzadeh A., Koç E., Otri S., Rahim S., and M.Zaidi "The Bees Algorithm – A Novel Tool for Complex Optimisation Problems"", Proceedings of IPROMS 2006 Conference, pp.454-461

Vedoucí bakalářské práce: Ing. Radomil Matoušek, Ph.D.

Termín odevzdání bakalářské práce je stanoven časovým plánem akademického roku 2009/2010.

V Brně, dne

L.S.

Ing. Jan Roupec, Ph.D. Ředitel ústavu	prof. RNDr. Miroslav Doupovec, CSc. Děkan fakulty

ABSTRAKT:

Obsahem práce je naprogramování Java třídy a skriptu v Matlabu, které budou implementovat včelí algoritmus (BeesAlgorthm). K otestování výkonu tohoto algoritmu je využito standartních optimalizačních funkcí Rosenbrock a Rastrigin.

ABSTRACT:

In this paper was tested Bees Algorithm on standard optimalizacion problems Rosenbrock and Rastrigin function. For this test was programmed Java class and Matlab script for implementation Bees algorithm.

KLÍČOVÁ SLOVA:

Včelí algoritmus, optimalizace, Rosenbrock function, Rastrigin function

KEY WORDS:

Bees algorithm, optimalization, Rosenbrock function, Rastrigin function

PODĚKOVÁNÍ

Děkuji vedoucímu bakalářské práce Ing. Radomilu Matouškovi Ph.D. za pomoc a odborné vedení při vypracování této práce.

OBSAH

Zadání závěřečné práce 3
Licenční smlouva 5
Abstrakt 7
Poděkování 13
Použité symboly a výrazy 13
1. ÚVOD 15
2. POPIS VČELÍHO ALGORITMU 17
2.1 Používané metody 17
2.2 Chování včelího roje 18
2.3 Včelí algoritmus 19
2.3.1 Optimální dělení práce 19
2.3.2 Základní struktura algoritmu 19
2.3.3 Zápis původního algoritmu v Matlabu 20
2.3.4 Popis algoritmu 21
2.4 Optimalizační problémy řešitelné včelím algoritmem 22
3. IMPLEMENTACE ALGORITMU 23
3.1 Úvod do Matlabu 23
3.2 Implementace v Matlabu 24
3.3 Úvod do Javy 24
3.4 Implementace v Javě 25
3.5 Otestování výsledků implementace 26
3.6 Srovnání 27
4. TESTOVÁNÍ NA FUNKCÍCH 29
4.1 Popis testování 29
4.2 Tabulkové srovnání 30
4.3 Grafické srovnání 32
4.4 Vliv vyhledávajícího rádiusu 34
4.4 Minima a maxima funkce 35
5 ZÁVĚR 37
Seznam použité literatury 39

Použité symboly a výrazy

Bees	včely
Bees algorithm, BA	včelí algoritmus
Elite places	počet nejlepších výsledků s plným počtem včel
Best places	počet nejlepších výsledků se sníženým počtem včel
Elite bees	plný počet včel okolo nejlepšího výsledku
Best bees	snížený počet včel okolo nejlepšího výsledku
Ngh	radius vyhledávání okolo zvoleného nejlepšího výsledku
Number variables, D	počet proměnných ve funkci
Max value all variables	maximum definičního oboru funkce
Min value all variables	minimum definičního oboru funkce
Number of samples	počet měření
Max iterations	počet iterací

1. ÚVOD

Optimalizační metody hejnových algoritmů spočívají v napodobování chování skupin zvířat v přírodě jako jsou včely nebo mravenci. Matematickým popsáním jejich chování jsou tzv. hejnové algoritmy, mezi které patří i včelí algoritmus. Tyto algoritmy využívají takzvané kolektivní inteligence, kterou lze popsat jako schopnost skupiny dosáhnout výhodnějšího řešení problému než její jednotliví členové [1]. Včelí algoritmus napsal Dervis Karaboga v roce 2005. Snahou jeho algoritmu je popsat chování včel při hledání zdrojů potravy v okolí úlu, neboli řešení vícedimenzionálních optimalizačních problémů [2]. Včely mají složitý systém komunikace mezi sebou. Hlavní způsob výměny informací o zdrojích potravy a jejich umístění je známý jako včelí tanec, při kterém se některé včely při návratu z pastvy do úlu, točí a pohupují, čímž předávají ostatním včelám informace o zdroji potravy, od kterého se právě vrátily.

Cílem této práce je otestovat schopnost včelího algoritmu najít minimum na standartních optimalizačních úlohách Rosenbrock a Rastrigin. Obě funkce mají minimální hodnutu 0. Dalším bodem práce je napsání tříd a funkcí pro platformy Matlab a Java, které budou tento algoritmus implementovat.

2. POPIS VČELÍHO ALGORITMU

2.1 Používané metody

Pro řešení optimalizačních problémů lze využít přímé matematické metody jako například simplexová metoda nebo lineární programování. Tyto metody jsou náročné na přesné zadání vstupních parametrů a přesné popsání daného problému. Potřebují velký výpočetní výkon. Druhou možností jsou takzvané soft-computingové metody, někdy překládané jako „měkké‘‘, protože nevyžadují takový výpočetní výkon.

Soft-computingové metody využívají k řešení zadaného problému nepřesné či náhodné vyhledávání. Tyto metody jsou většinou inspirovány přírodními zákony, např. neuronové sítě nebo genetické algoritmy. Tyto metody jsou výhodné pro řešení složitých biologických, optimalizačních či matematických problémů. Výhodou je jejich relativní jednoduchost na matematický popis, který se dá shrnout jako interakce jednotlivých prvků systému, jejich následné vyhodnocení a vybrání vhodných kandidátů k dalšímu vývoji. Jednotlivé metody je nutné kombinovat k dosažení nejlepších možných výsledků. Nelze označit jednu metodu jako výhodnější než všechny ostatní, každá se dá použít k jinému účelu a s jiným množstvím vstupních informací. Pro úspěšné vyřešení daného problému je nutné znát charakteristické vlastnosti jednotlivých metod, nastavení vstupních dat a přesnost jejich výsledku.

Fuzzy systémy (Fuzzy systems)

Vyhodnocují výsledky nikoliv na binární úrovni 0 nebo 1, ale určují jeho částečnou příslušnost ke konkrétnímu extrému. Využívají spíše slovní popis hodnoty než jeho binární popis. Fuzzy neboli „rozmazané‘‘ či „neostré‘‘ řízení představil v roce 1965 Dr. Lotfi Zadeh. V dnešní době se používá k velkému množství aplikací od řízení praček přes videokamery po specializované metody rozpoznání obrazu při identifikaci osob.

Umělé neuronové sítě (Artificial neural network)

Umělé neuronové sítě simulují chování biologického neuronu, jedná se o spoustu jednotlivých prvků, které mají vstupy a výstupy vzájemně propojené a vzájemně si předávající informační impuls. Jejich výhodou je, že se mohou učit a dosahovat lepších výsledků. Využívají se například k rozpoznání ručně psaného písma nebo identifikaci osob podle hlasu. Nejrozšířenějším typem neuronových sítí je perceptonová síť.

Evoluční algoritmy (Evolutionary algorithm)

Jsou založeny na Darwinově evoluční teorii, která předpokládá přežití nejsilnějších a nejlepších jedinců. Během vývoje mnoha generací dochází k mutacím a výběru nejvhodnějších jedinců k vytvoření další generace.

Hejnové algoritmy (Swarm algorithm)

Tyto algoritmy se snaží popsat fungování hejna živočichů, například ptáků, mravenců nebo včel. Fungují na základních pravidlech, kdy je známo globální optimum a optimum každého jedince. Během optimalizace dochází k náhodnému prohledávání prostoru, a uchovávání nejvhodnějších funkčních hodnot.

2.2 Chování včelího roje

Včela medonosná je létavý blanokřídlý hmyz, který žije ve společenstvích čítajících až několik desítek tisíc jedinců. Původně žila jen v Evropě, ale na konci 17. století ji lidé rozšířili po celém světě kvůli jejímu hospodářskému využití při opilování zemědělských rostlin a sběru medu. Jako vedlejší produkty včelstva lidé využívají vosk, propolis, pyl a mateří kašičku. Na konci 19. století Johannes Mehring vyslovil názor, že se roj dá označit za obratlovce, protože jednotlivé povolání dělnic funguje jako jednotlivý orgán. Proto je nutné na včelstvo nahlížet jako na jeden nedělitelný organismus, nikoliv na jednotlivé spolupracující živočichy [3].

Ve včelstu jsou tři základní typy včel, jsou to matka kladoucí vajíčka, trubci s jediným úkolem oplodnit matku a dělnice, které se starají o veškeré další činnosti v úlu, mezi které patří především sběr potravy, čistění vnitřních prostor, obrana včelstva a výchova nových včel a matek. Dělnice projdou během svého života všechna tato povolání, po vylíhnutí 3 dny uklízejí buňky, aby do nich mohla matka naklást vajíčka. Poté asi týden krmí larvy a matku potravou. V následujícím stádiu pracují na stavbě a údržbě úlu až do 18 dne života, kdy vylétávají ven bránit úl a sbírat pyl, nektar a vodu. Neustálou námahou při sběru potravy dochází k opotřebení a vysílení jejich organismu a včela po 30-40 dnech života umírá.

Vyhledávání nových zdrojů potravy probíhá náhodným vyhledáváním několika málo starších včel, které tuto informaci předávají v úlu mladším včelám, které se rozletí daným směrem. Komunikace mezi včelami probíhá na různých úrovních od nejznámějšího včelího tance , přes předávání vzorku nalezeného nektaru po vylučování chemických látek. Úplné rozluštění komunikace je stále snem mnoha biologů. Nejlépe je popsán právě včelí tanec, který poprvé zkoumal Karl von Frish [3]. Včelí tanec má dva hlavní tvary, pokud se zdroj potravy nachází v blízkém okolí úlu maximálně do 100 m, jeho tvar je téměř kruhový a předává jen informaci o tom, že někde v okolí je zdroj potravy. O jaký zdroj se jedná, včely zjistí tím, že přijmou pachovou stopu tančící včely a nalezení konkrétného místa sběru potravy je už jen otázka krátkého náhodného hledání v okolí úlu.

Pokud je zdroj potravy ve větší vzdálenosti, včelí tanec má tvar osmičky. Včely vždy tancují na jednom místě poblíž vstupu do úlu. Po návratu do úlu včela začíná tancovat a některé včely začnou její pohyby přesně kopírovat. Dochází k předávání mnoha informací o poloze místa sběru, jako například jeho vzdálenost od úlu, směr letu nebo vydatnost zdroje potravy. Vydatnost je reprezentována velikostí taneční plochy a rychlostí tance. Čím vydatnější zdroj potravy, tím je tanec rychlejší a na větší ploše. Nezaměstnané včely čekají na tanečním místě a sledují vibrace plástve. Když je zaregistrují, vydají se směrem ke zdroji vibrací a zapojí se do tance.

Mladým včelám však trvá nalezení určeného místa až třicetkrát déle než starším, tudíž je nutná i komunikace venku. Karl von Frish zjistil, že některé včely po příletu k místu sběru hlasitě bzučí a vypouštějí vonnou chemickou látku a tím pomáhají najít místo sběru i mladým včelám. Občas dochází k tomu, že starší včela vede 5 – 10 mladých včel na místo sběru a až začně sbírat, začnou se sběrem ostatní členové tohoto malého roje.

2.3 Včelí algoritmus

2.3.1 Optimální dělení práce

Optimalizace cestou včelího algoritmu je dynamický process, který obsahuje vlastní organizaci a dělbu práce, Na globální úrovni je výsledkem interakce mezi základnímy prvky systému (včelami). Ty nemají žádnou představu o globálním stavu systému, jen předávají svůj vlastní aktuální stav. Ten se mění v zavislosti na několika základních pravidlech a vazbách mezi základními prvky. Pravidla o komunikaci mezi prvky jsou definovány [2] jako:

a) **Pozitivní zpětná vazba**

Slouží ke hledání výhodnějších míst sběru potravy nebo výhodnějších tras

b) **Negativní zpětná vazba**

Zabraňuje přesycení, dá se popsat jako vyčerpatelnost každého zdroje potravy

c) **Fluktuace**

Náhodné hledání v prostoru kde se může nacházet řešení

d) **Vlastní organizace systému**

Vyhodnocení výsledků, aktuální stav zdrojů potravy, nové rozložení kolonie

Dělba práce probíhá průběžně v závislosti na zjištěném stavu systému. V roji je několik možných prací, které je nutno vykonávat současně (hledání nových zdrojů potravy, předávání informací o nich, sběr potravy). Každou práci vykonává specializovaný jedinec. Tento způsob dělby práce je považován za výhodnější, než když se sekvenčně zpracovávají úkoly mnoha univerzálními jedinci.

2.3.2 Základní struktura algoritmu

K základnímu popsání algoritmů se používá pseudokód. Jde o zjednodušený zápis programu, který zahrnuje pouze jeho vazby a strukturu. Nedefinuje proměnné, procedury ani cykly. Mnohdy je jen vyjádřen slovně a popisuje co vlastně daný blok má udělat.

Pseudokód:

```
1) Vyslání náhodných včel
 Pokud nebude splněna ukončující podmínka opakuj:
 {
 2)  Seřazení včel dle jejich výsledků (vzestupně/sestupně)
 3)  Vyslání skupiny včel do okolí určeného počtu nejlepších
 výsledků ( včely s nejlepší hodnotou )
 4)  Pokud najde některá ze včel vyslaných na určené místo lepší
 hodnotu než původní včela, tak ji nahradí
 5)  Včely, které nebyly označeny za vhodné k dalšímu rozvoji se
 přepíšou novými náhodnými a vypočte se jejich hodnota
 6)  Vyhodnocení ukončující podmínky (počet iterací, dosažení
     určené hodnoty atd. )
 }
```

2.3.3 Zápis původního algoritmu v Matlabu

```
n= 30;      %  počet včel v celém roji
itr=15;     %  počet iterací
e=20;       %  počet nejlepších míst
m=10;       %  počet nejlepších míst s nižší hustotou včel
n2=30;      %  počet včel okolo nejleších míst
n1=15;      %  snížený počet včel okolo nejlepších míst
ngh=0.0234; %  rádius vyhledávání okolo nejlepších výsledků
x_max=3;y_max=3;x_min=-3;y_min=-3;     % definiční obor
```

```
1.   U=X_random(n,x_max, y_max, x_min, y_min);
2.   Par_Q=sortrows([U(:,1), U(:,2), peaks(U(:,1),U(:,2))],-3);
3.   clear U;

4.   for k=1:itr            % počet iterací
5.     for j=1:e            % počet nejlepších míst
6.       for i=1:n2         % počet včel kolem nejlepšího místa
7.         U=bee_dance(ngh, Par_Q(j,1), Par_Q(j,2));
8.         if peaks(U(1),U(2))> Par_Q(j,3)
9.             Par_Q(j,:)=[U(1), U(2), peaks(U(1),U(2))];
10.        end
11.      end
12.     end

13.    for j=e+1:(m + e)     % počet nejlepších míst se sníženým
                               počtem včel
14.      for i=1 : n1        % snížený počet včel
15.         U=bee_dance(ngh,Par_Q(j,1),Par_Q(j,2));
16.      if peaks(U(1),U(2))> Par_Q(j,3)
17.         Par_Q(j,:)=[U(1), U(2), peaks(U(1), U(2))];
18.
19.      end
20.    end

21.    for i=j+1:n           % počet všech včel
22.      U=X_random(2,x_max, y_max, x_min, y_min);
23.       Par_Q(i,:)=[U(1), U(2), peaks(U(1),U(2))];
24.    end

25.    Par_Q=sortrows(Par_Q,-3);
26.    Best(k,:)=Par_Q(1:15,3)';
27.  end
```

Doplňující funkce:

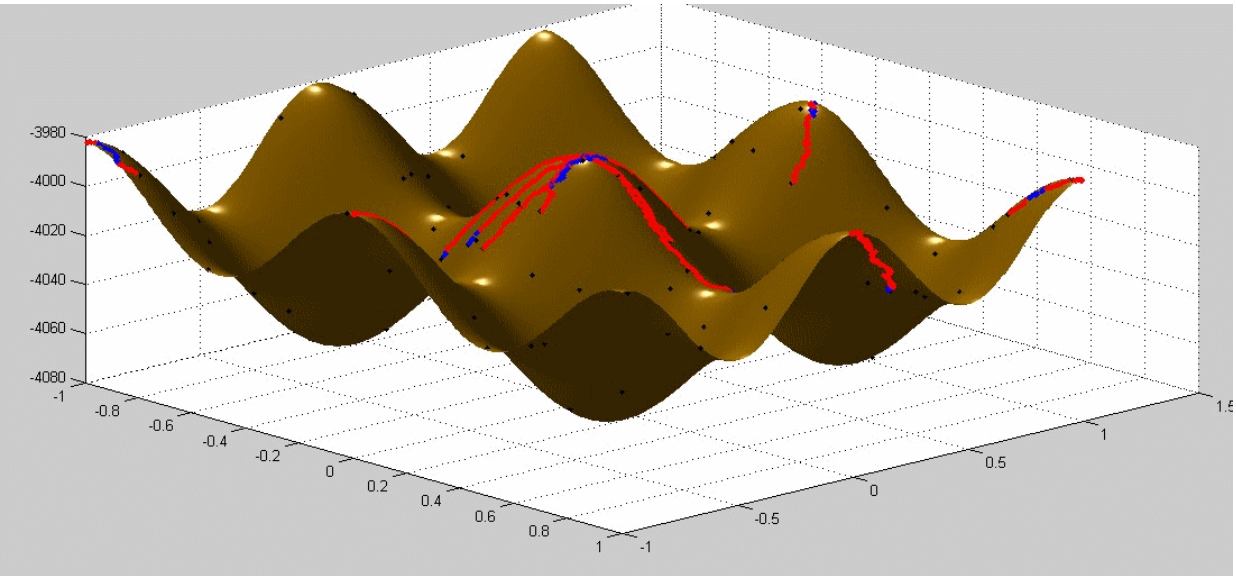
```
function U=bee_dance(ngh,x1, x2)
    U(:,1)=(x1-ngh)+(2*ngh.*rand(size(x1,1),1));
    U(:,2)=(x2-ngh)+(2*ngh.*rand(size(x2,1),1));
end

function X=X_random(n, x_max, y_max, x_min, y_min)
    X=[x_min+((rand(n,1)).*(x_max-x_min)), y_min+((rand(n,1)).*(y_max-
    y_min))];
end
```

2.3.4 Popis algoritmu

První tři řádky vygenerují náhodné rozložení včel v zadaném prostoru ohraničeném hodnotami x_max, y_max, y_min, x_min. Metoda sortrows je seřadí dle sloupce 3, znaménko mínus označuje sestupné řazení. Tento algoritmus je jen demonstrační pro funkci dvou proměnných vestavěné funkce peaks v Matlabu. Následující řádky 4 – 12 zajišťují vlastní funkci algoritmu. Začínají zde iterace, v každé z nich se vyberou nejlepší výsledky, a kolem každého z nich se postupně vygeneruje určený počet včel, v počtu n1, pokud jde o výsledky se sníženým počtem včel bude jejich počet n2. Postupně se po jedné generují a pokud má konkrétní včela lepší hodnotu než ta původní, přepíše ji. Zde dochází ke zbytečnému porovnávání a přepisování jednoho výsledku. Tento problém byl vyřešen tím, že cyklus na šestém řádku byl odstraněn a nahrazen funkcí RandBeesPlace, která kolem určeného nejlepšího výsledku vygeneruje určený počet včel. Tyto včely následně seřadí a předá jejich seřazenou matici zpět do vlastního algoritmu, který porovnává jen původní včelu s nejlepší novou hodnotou, a pokud je lepší, nahradí ji. Řádky 13 až 20 mají stejný význam jako předchozí blok, jediné v čem se liší je snížený počet včel okolo zvoleného nejlepšího výsledku. Tímto způsobem se šetří výpočetní čas potřebný k vyřešení optimalizačního problému. Následující blok čtyř řádků slouží k novému vygenerování náhodných včel, které neměli dostatečnou hodnotu k zařazení mezi nejlepší vysledky. Poslední 3 řádky seřadí všechny včely podle jejich hodnoty a tím se ukončí jedna iterace.

Funkce bee_dance má vstupní parametry ngh, x1 a x2. Ngh určuje rozsah okolí, ve kterém se bude generovat jedna včela, kterou funkce vrátí do algoritmu k porovnání. X_random vytváří matici včel, parametr n určuje počet včel, x a y jsou parametry ohraničení prohledávaného prostoru.

Při prvních testech na 500 vzorcích se ukázalo, že původní algoritmus má 3x delší dobu výpočtu než urychlený, z tohoto důvodu jsem využíval upravenou verzi pro další programování v Javě.



*Obr. 1 Vykreslení postupného hledání maxima funkce Rastrigin * -1*

2.4 Optimalizační problémy řešitelné včelím algoritmem

Včelí algoritmus lze využít k optimalizaci množství rozmanitých problémů. Na stránkách projektu [4] jsou ukázky prací, kde řeší například celulární automaty, rozdělování práce strojů, optimalizování umístění a velikosti svaru při svařování, výběr tvaru láhve na nápoje, rozmístění součástek na plošných spojích a spousta dalších. Většina se jich v závěru shoduje v tom, že včelí algoritmus je výhodnější k řešení daného problému než jiné běžně používané metody a má ještě jiné výhody, například směrodatná odchylka je menší, je rychlejší a lépe nachází konečnou hodnotu.

3. IMPLEMENTACE ALGORITMU

3.1 Úvod do Matlabu

Matlab je produkt softwarové společnosti Mathworks. Od roku 1984 jde o nejrozšířenější sotware pro matematické výpočty a modelování. Jejich hlavním produktem je Matlab, jehož název je zkrácením anglických slov MATrix a LABoratory, která se dají přeložit jako laboratoř s maticemi, protože základním formátem čísla zde je matice. Nádstavbou Matlabu je Simulink, umožňující modelování diferenciálních nespojitých rovnic.

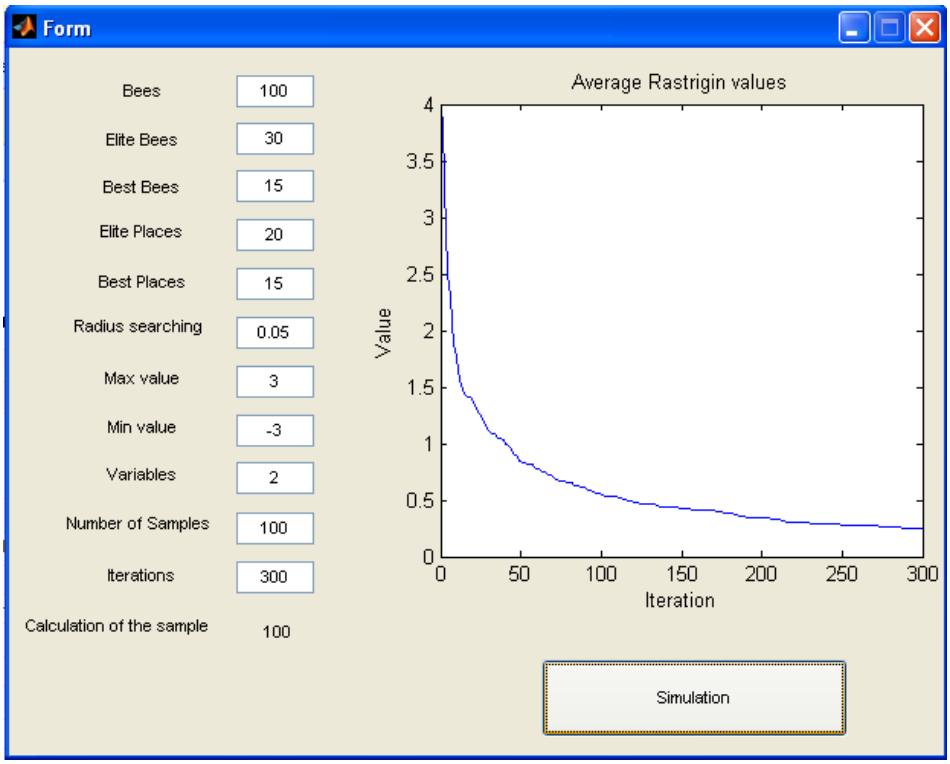
Matlab byl původně nádstavbou knihoven Linpack a Eispack [5], a zaujal především tím, že nebylo nutné zadávat velikost matice, se kterou se právě pracovalo, každá proměnná byla automaticky matice a její rozměr se nastavoval automaticky. V dnešní době je Matlab podstatně více než jen nádstavbou dvou knihoven, dokáže načíst uživatelské funkce ze souborů nazvaných m-file, lze do něj přidat moduly mex-file zkompilované do strojového kódu pro processor. Tyto mex-file lze psát v libovolném programovacím jazyku například C++ nebo Fortran. Lze tímto způsobem zkompilovat i m-file za pomoci kompilátoru, který je přímo obsažen v Matlabu.

Uživatelské prostředí je velmi pohodlné, lze jednoduše pomocí průvodce vytvořit uživatelské rozhraní se spoustou nastavitelných parametrů, tlačítka volající konkrétní funkce a pousta dalších uživatelským možností. Výsledky lze prezentovat za pomocí spousty typů grafů, dle požadavků ve 2D nebo 3D. Základním prvkem v Matlabu jsou toolboxy, které obsahují již zpracovanou část matematiky, jako například statistiku, numeriku, atd. Jsou to vlastně adresáře obsahující již napsané m a mex soubory s patřičnými komentáři a návody k použití.

Prostředí Matlab jsem zvolil, protože na stránkách projektu [4] je ukázka včelího algoritmu při hledání optima prezentační funkce peaks i s kódem a komentáři.

3.2 Implementace v Matlabu

V originální verzi programu jsou skripty main, obsahující vlastní algoritmus, a dvě funkce random_x a bee_dance. Random funkce generovala náhodné rozmístění včel a bee_dance vytvářela náhodnou včelu v okolí konkrétního zvoleného vhodného výsledku. Po úpravách byl přidán skript form, který obsahuje uživatelské rozhraní umožňující pohodlně měnit vstupní parametry, a obsahující vlastní algoritmus. Dále jsou potřebné tři funkce. RandBees, která generuje matici náhodných včel, RandBeesPlace tvořící matici včel v okolí konkrétného zvoleného místa a sortmatice, řadící včely dle kritéria. Následně se porovnává jen první nejlepší včela s původní, a pokud bude mít lepší hodnotu, tak ji nahradí. Po nastavení hodnot a stisknutí tlačítka se spustí skript. Po jeho ukončení se v grafu se vykreslí průměrná hodnota v konkrétním cyklu.



Obr. 2 Uživatelské rozhraní v Matlabu, funkce Rastrigin

3.3 Úvod do Javy

Jazyk Java je produkt společnosti Sun Microsystems, která jej vyvinula v roce 1995. Jedná se o plně objektový programovací jazyk. V roce 2007 byly uvolněny zdrojové kódy a licence změněna na open source. Jazyk Java je interpretovaný jazyk, tedy nevytváří spustitelný soubor, ale jen mezikód, který se při spuštění znovu přeloží pro konkrétní platformu, kde je spouštěn. Program napsaný v Javě lze tedy spustit na jakémkoliv operačním systému, který má k dispozici interpret JVM – Java Virtual Machine. Při přenosu aplikace z vyšší platformy (osobní počítač) na platform nižší (mobilní telefon) může dojít ke ztrátě funkčnosti, z důvodu jednoduchosti interpretoru, který neobsahuje všechny potřebné funkce. Java byla v dřívějších verzích oproti C++

výrazně pomalejší, protože nejprve docházelo k překladu a teprve po přeložení se spouštěl vlastní program. Současné verze jsou již srovnatelně rychlé a tento problém obcházejí procesem nazvaným garbage collector. Jde o změnu správy paměti, kdy dochází k mazání již nepotřebných dat. Další urychlení běhu programu umožnilo efektivnější překládání formou „právě včas'' kdy se překládá jen ta část kódu, která je právě potřeba.

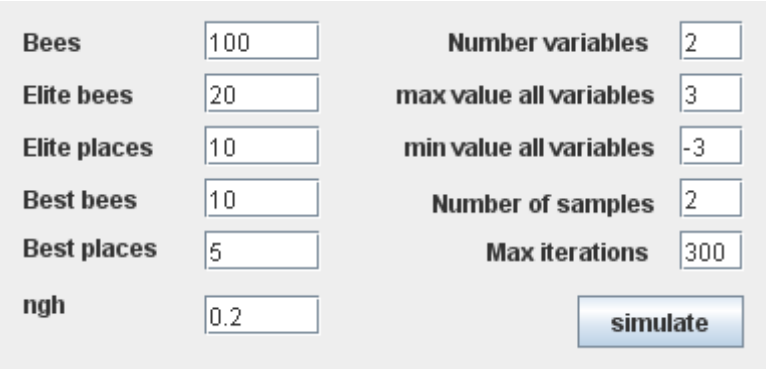
Psaní programu v Javě je intuitivní, jde totiž o zjednodušený zápis podobný C++, doplněný o vyžadování ošetření možných chyb (například zápis do souboru musí jít přes try-catch), vylepšenou správou paměti a snahou znemožnit programátorům psát složité konstrukce z jazyka C, které způsobovaly problémy v běhu programu, například skoky goto při běhu programu.

Jazyk Java jsem zvolil z důvodu snadné přenositelnosti mezi platformami na osobních počítačích a schopnosti Matlabu spouštět Java třídy a pracovat s nimi. Tohoto se mi nepovedlo dosáhnout, přestože jednodušší třídy Matlab spustil, u tohoto projektu nastal problém s datovými typy.

3.4 Implementace v Javě

Rozdělení funkcí programu v Javě je podobné jako v Matlabu, rozdíl je v počtu tříd, protože Java neumožňuje přímou práci s maticemi, jejich řazení a formátování výstupu do grafu. Importovány jsou open source knihovny JFreeChart a jExcelAPI. Pro generování grafů a xls souborů.

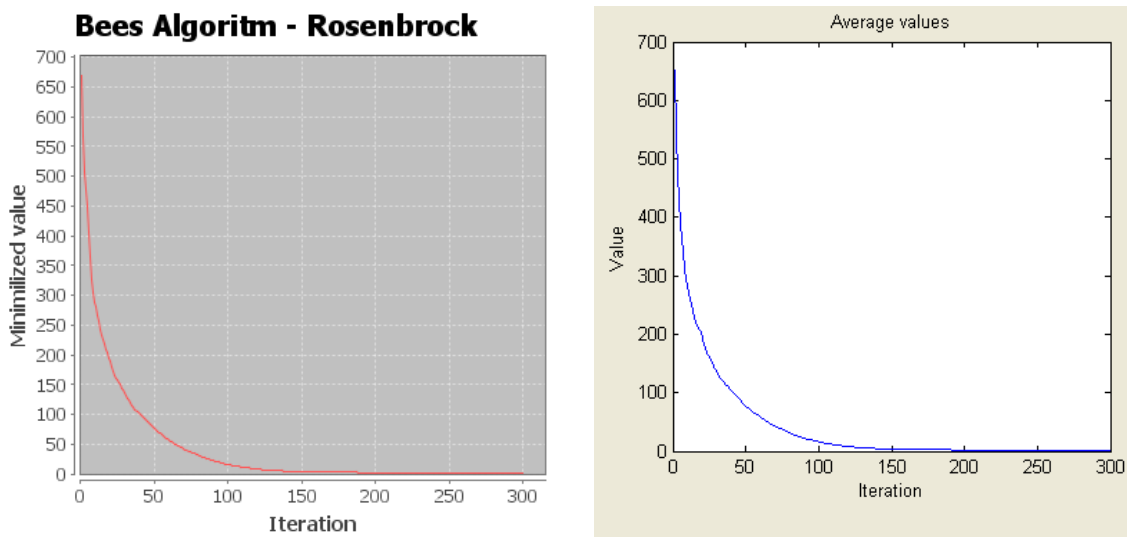
BeesAlgorithm – Obsahuje vlastní algoritmus
Frame – Uživatelské rozhraní
Function – optimalizovaná funkce
NahodneRozmisteni – Náhodně vygeneruje počet včel v daném rozsahu
Seradit – seřadí matici náhodných včel od nejmenšího k největšímu
ToExcel – předá výsledky algoritmu do xls souboru pro následné statistické zpracování
Graph – vykreslí graf průběhu algoritmu



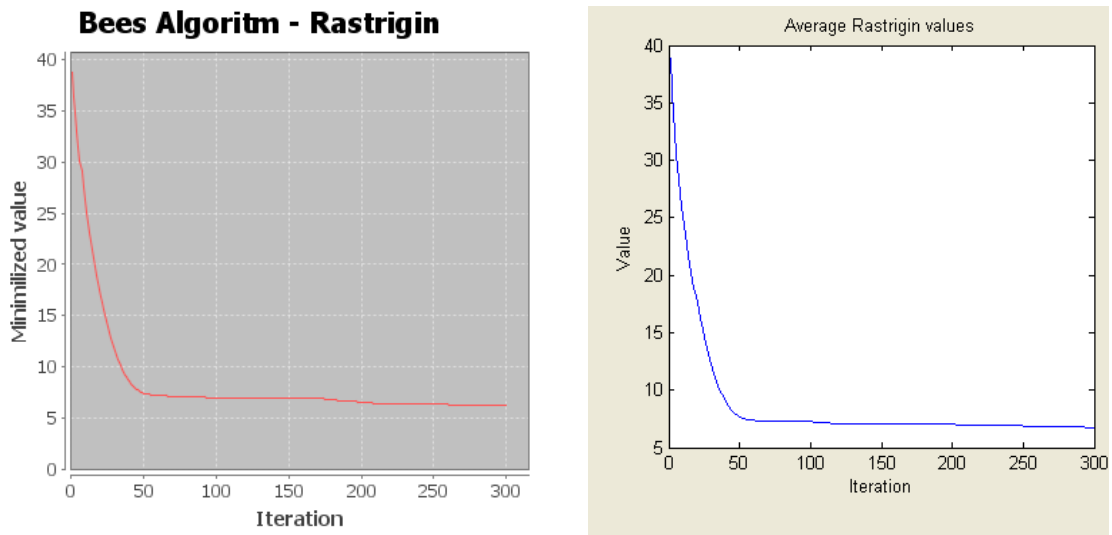
Obr. 3 Uživatelské rozhraní v Javě

3.5 Otestování výsledků implementace

Bees:	100
Elite bees:	20
Elite places:	5
Best bees:	10
Best Places:	2
Počet proměnných	7
Rádius vyhledávání	0.01
Ohraničení prostoru	+3/-3



Obr. 4 Výsledek porovnání u funkce Rosenbrock – Java / Matlab



Obr. 5 Výsledek porovnání u funkce Rastrigin – Java / Matlab

3.6 Srovnání

První testování algoritmu probíhalo v Matlabu, časová náročnost výpočtu se ukázala jako velký problém, kvůli kterému veškeré další testy probíhali v Javě, ve které je tento výpočet asi 2x rychlejší. Z pohledu náročnosti na sepsání kódu je výhodnější Matlab, který má výrazně úspornější zápis, například funkce RandBees v Matlabu má 10 řádků, stejná třída v Javě je asi trojnásobně delší.

	Java	Matlab
Průměr	1.52	1.38
Úspěšnost	8%	7%

Porovnání u funkce Rosenbrock

	Java	Matlab
Průměr	6.36	6.62
Úspěšnost	0%	0%

Porovnání u funkce Rastrigin

Z podobných naměřených hodnot lze předpokládat, že implementace algoritmu proběhla u funkce Rosenbrock i Rastrigin úspěšně.

4. TESTOVÁNÍ NA FUNKCÍCH

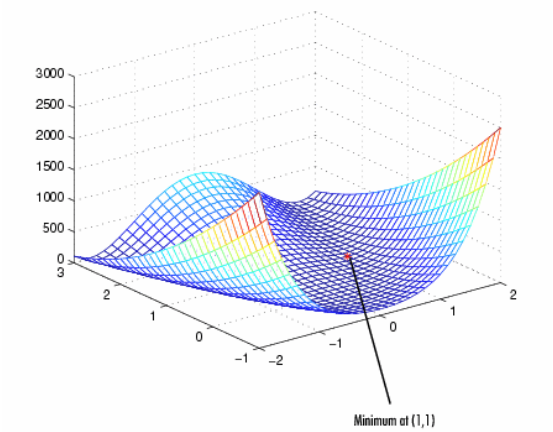
4.1 Popis testování

Včelí algoritmus byl testován na standardních úlohách Rastrigin a Rosenbrock. Zjišťovala se jeho schopnost najít optimální hodnotu funkce pro daný počet proměnných ve funkci, počtu včel a jejich časové náročnosti. Počet iterací byl zvolen s ohledem na potřebný výpočetní čas. Počet elite places a best places byl nastaven procentuálně se zastropováním na maximu 30 a 15. Testování probíhalo na osobním notebooku, kde běžely i jiné programy, čas byl měřen na stopkách mobilního telefonu. Proto je nutné brát změřený čas jako orientační. I kdyby šlo o čistý procesorový čas, vlivem taktu procesoru na jiném zařízená se výpočtový čas může měnit.

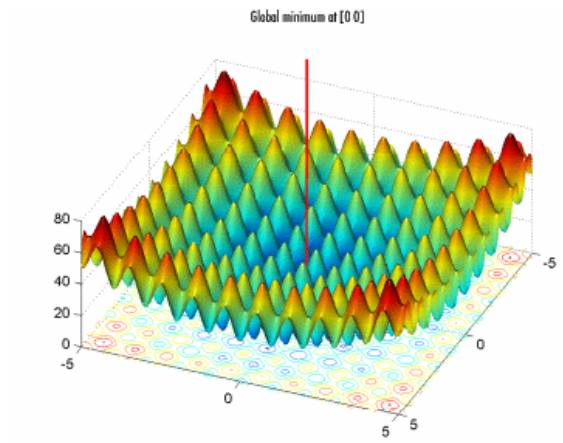
Nastavené parametry:

Počet cyklů:	300
Radius hledání	0.01
Elite place	20% ze včel, max 30
Elite včel	20% ze včel, max 30
Best míst	10% ze včel max 15
Best včel	10% ze včel max 15
Definiční obor	+3/-3
Počet měřených vzorků	100

Výsledek je úspěšný pokud dosáhne hodnoty 0.05 a nebo menší.



Obr. 6 Rosenbrock function
Zdroj: www.mathworks.com



Obr.7 Rastrigin functio
Zdroj: www.mathworks.com

$F(x,y) = (1 - x)^2 - 100*(y - x^2)^2$
Fmin (1,1) = 0

$F(x,y) = 20 + x^2 + y^2 - 10 * (\cos 2\pi x + \cos 2\pi y)$
Fmin (0,0) = 0

4.2 Tabulkové srovnání

Rosenbrock

10 včel – elite places = 2, best places = 1, elite bees = 2, best bees = 1

	2D	5D	10D	25D	50D	100D
Čas	1,7 s	1,9 s	2,5 s	7,1 s	10,2 s	20,1 s
Směr. odchylka	0,011868	0.83	114.25	1420,971	3057,223	6379,509
Medián	0,000348	3.05	254.01	6239,356	26822,4	84843,63
Průměr	0,004337	3.60	248.55	5871.05	26517,18	83132,35
Úspěšnost	98%	1%	0%	0%	0%	0%
Minimální hodnota	2,47E-07	0.025	15.83	1135,472	16699,12	62573,93

20 včel – elite places = 4, best places = 3, elite bees = 4, best bees = 2

	2D	5D	10D	25D	50D	100D
Čas	1,9 s	2,8s	5,3 s	14,9 s	28.2 s	58,2 s
Směr. odchylka	0,000432154	1,113084	36,56601	778,5466	2393,329	5225,651
Medián	2,10551E-06	0,716435	70,9720	3418,939	19198,42	68505,98
Průměr	0,000114565	1,189771	72,64081	3325,613	19144,04	67419,8
Úspěšnost	100%	12%	0%	0%	0%	0%
Minimální hodnota	4,11154E-08	0,001973	4,344661	788,3968	11262,76	49249,76

50 včel – elite places = 10, best places = 5, elite bees = 10, best bees = 5

	2D	5D	10D	25D	50D	100D
Čas	9,1 s	17,3	31,4 s	1 min 12 s	2 min 23 s	4 min 44 s
Směr. odchylka	1,19664E-07	0,230353	8,467393	314,3549	1522,132	4198,577
Medián	5,35969E-08	0,103646	8,45275	1307,881	11957,2	50645,93
Průměr	1,01408E-07	0,188484	10,69524	1334,276	11545,8	49671,81
Úspěšnost	100%	33%	1%	0%	0%	0%
Minimální hodnota	1,28735E-09	0,000959	0,038489	610,4207	6055,992	33116,46

100 včel – elite places = 20, best places = 10, elite bees = 20, best bees = 10

	2D	5D	10D	25D	50D	100D
Čas	37 s	1 min 7 s	1 min 58	4 min 32 s	8 min 55 s	18 min 8s
Směr. odchylka	2,07273E-08	0,060583	2,538421	184,2184	1190,343	3051,279
Medián	1,31998E-08	0,034055	4,084356	694,0429	7966,804	40342,36
Průměr	1,84916E-08	0,055167	3,789978	691,378	7759,732	39873,67
Úspěšnost	100%	61%	6%	0%	0%	0%
Minimální hodnota	1,15367E-10	0,000215	0,024792	277,4166	3803,636	30683,04

250 včel – elite places = 30, best places = 15, elite bees = 30, best bees = 15

	2D	5D	10D	25D	50D	100D
Čas	1 min 20s	2 min 44s	4 min 25s	9 min	18 min	33 min
Směr. odchylka	7,27273E-09	0,029707	1,825159	105,6236	849,3641	2495,202
Medián	4,22522E-09	0,01406	1,94099	394,9684	5855,694	34151,5
Průměr	6,57527E-09	0,026311	2,166116	401,167	5758,268	33724,87
Úspěšnost	100%	79%	4%	0%	0%	0%
Minimální hodnota	4,69842E-11	0,000184	0,005516	182,3106	3383,479	28421,46

500 včel – elite places = 30, best places = 15, elite bees =30, best bees = 15

	2D	5D	10D	25D	50D	100D
Čas	3min 4 s	4 min 32 s	7 min 32 s	15 min	22 min	40 min
Směr. odchylka	4,92283E-09	0,024701	1,636387	87,00608	722,0881	2502,967
Medián	3,11665E-09	0,010697	0,952624	334,1498	5302,454	31964,43
Průměr	5,1528E-09	0,02209	1,713313	336,4272	5233,561	31561,05
Úspěšnost	100%	82%	11%	0%	0%	0%
Minimální hodnota	5,09004E-11	0,000438	0,023626	155,8864	3263,327	23044,73

Rastrigin

10 včel – elite places = 2, best places = 1, elite bees = 2, best bees = 1

	2D	5D	10D	25D	50D	100D
Čas	1,2 s	1,6 s	2,2 s	4.4 s	7,9s	14,8
Směr. odchylka	0,405244745	2,847564	5,256085	9,598149	19,17157	27,31587
Medián	4,46534E-05	4,979447	16,92901	69,73233	241,5833	677,7075
Průměr	0,208993765	5,350752	17,47894	70,10849	239,6669	676,9952
Úspěšnost	79%	1%	0%	0%	0%	0%
Minimální hodnota	2,90654E-08	0,00734	6,98538	46,34282	166,9236	590,4911

20 včel – elite places = 4, best places = 3, elite bees = 4, best bees = 2

	2D	5D	10D	25D	50D	100D
Čas	2,6 s	3,5 s	5,4 s	11,2 s	22,8 s	44 s
Směr. odchylka	0,216843724	1,91409	4,483139	9,01307	14,18562	23,17668
Medián	9,48345E-06	3,981808	14,93815	63,66952	151,1625	498,5925
Průměr	0,049764148	3,966076	14,9322	51,98665	149,6051	492,1998
Úspěšnost	95%	1%	0%	0%	0%	0%
Minimální hodnota	7,12283E-08	0,004286	3,988399	31,08338	103,5898	447,924

50 včel – elite places = 10, best places = 5, elite bees = 10, best bees = 5

	2D	5D	10D	25D	50D	100D
Čas	8.8 s	14, 5 s	25 s	57 s	1 min 54s	3 min 37 s
Směr. odchylka	4,5707E-06	1,242335	3,272887	7,409169	10,76734	15,40064
Medián	2,58014E-06	2,985629	12,94709	47,86402	110,7895	319,0983
Průměr	1,43011E-06	2,787493	12,29118	47,07644	111,6407	317,547
Úspěšnost	100%	2%	0%	0%	0%	0%
Minimální hodnota	1,90926E-08	0,006203	3,997803	20,9869	81,39604	274,1042

100 včel – elite places = 20, best places = 10, elite bees = 20, best bees = 10

	2D	5D	10D	25D	50D	100D
Čas	32 s	1 min	1 min 35 s	3min 30s	7 min 12 s	14 min
Směr. odchylka	4,7463E-07	0,865872	2,520806	5,894078	8,576431	13,95041
Medián	3,45884E-07	1,991149	9,961045	41,90466	104,4959	261,3903
Průměr	5,20365E-07	2,100647	9,999747	41,79887	104,6061	259,9874
Úspěšnost	100%	4%	0%	0%	0%	0%
Minimální hodnota	2,40155E-09	0,000732	3,987462	22,97959	81,9941	221,5756

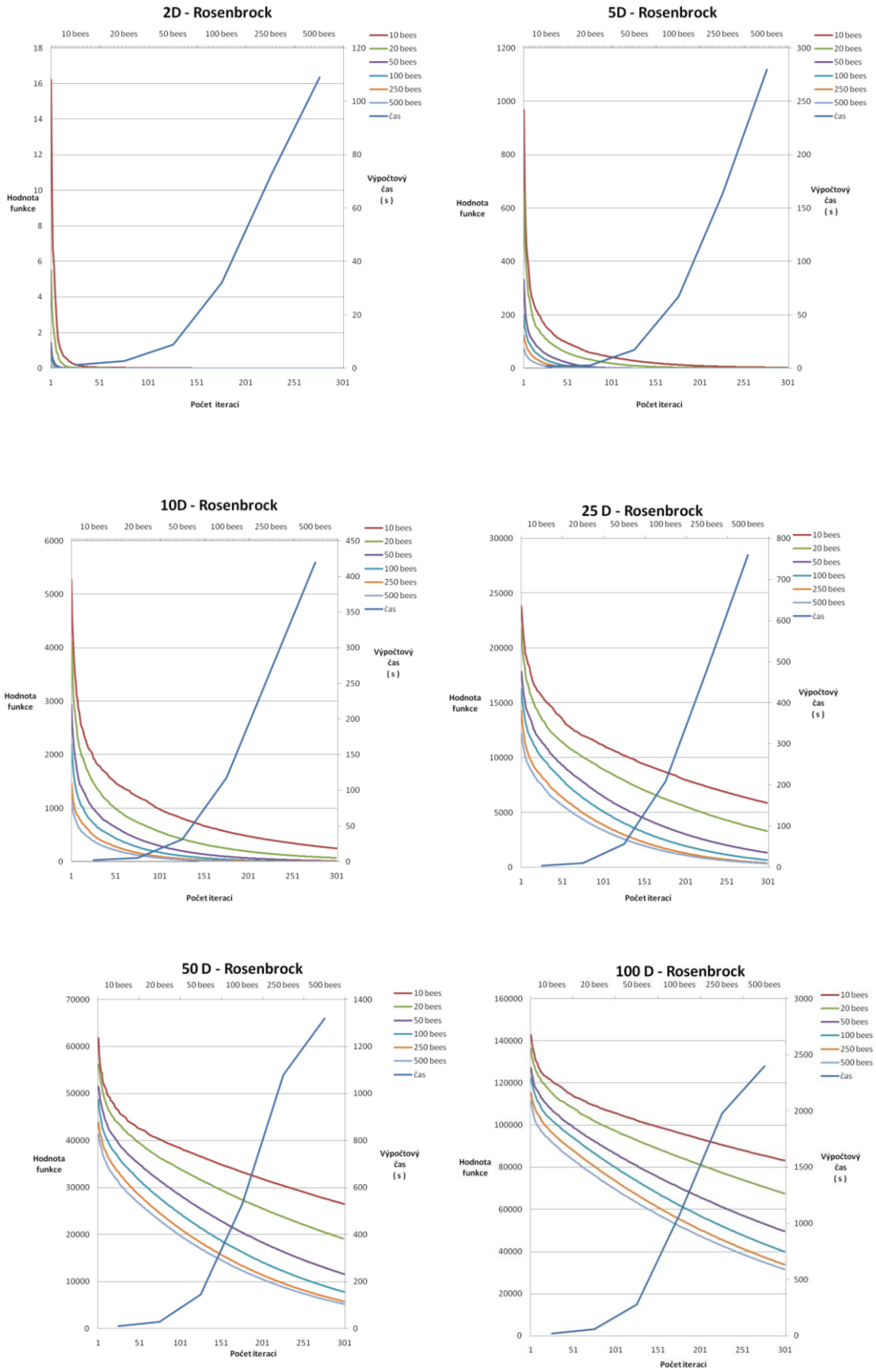
250 včel – elite places = 30, best places = 15, elite bees = 30, best bees = 15

	2D	5D	10D	25D	50D	100D
Čas	1 min 12s	2 min 03s	4 min 15s	8 min	16 min	31 min
Směr. odchylka	3,52624E-07	0,833475	2,680162	5,272946	8,164446	12,55247
Medián	2,41379E-07	1,990795	9,95984	40,88083	100,4544	238,409
Průměr	3,18206E-07	1,90143	9,850174	40,81151	99,86961	238,1846
Úspěšnost	100%	5%	0%	0%	0%	0%
Minimální hodnota	1,25026E-09	0,000354	0,996844	26,9373	77,18184	197,7005

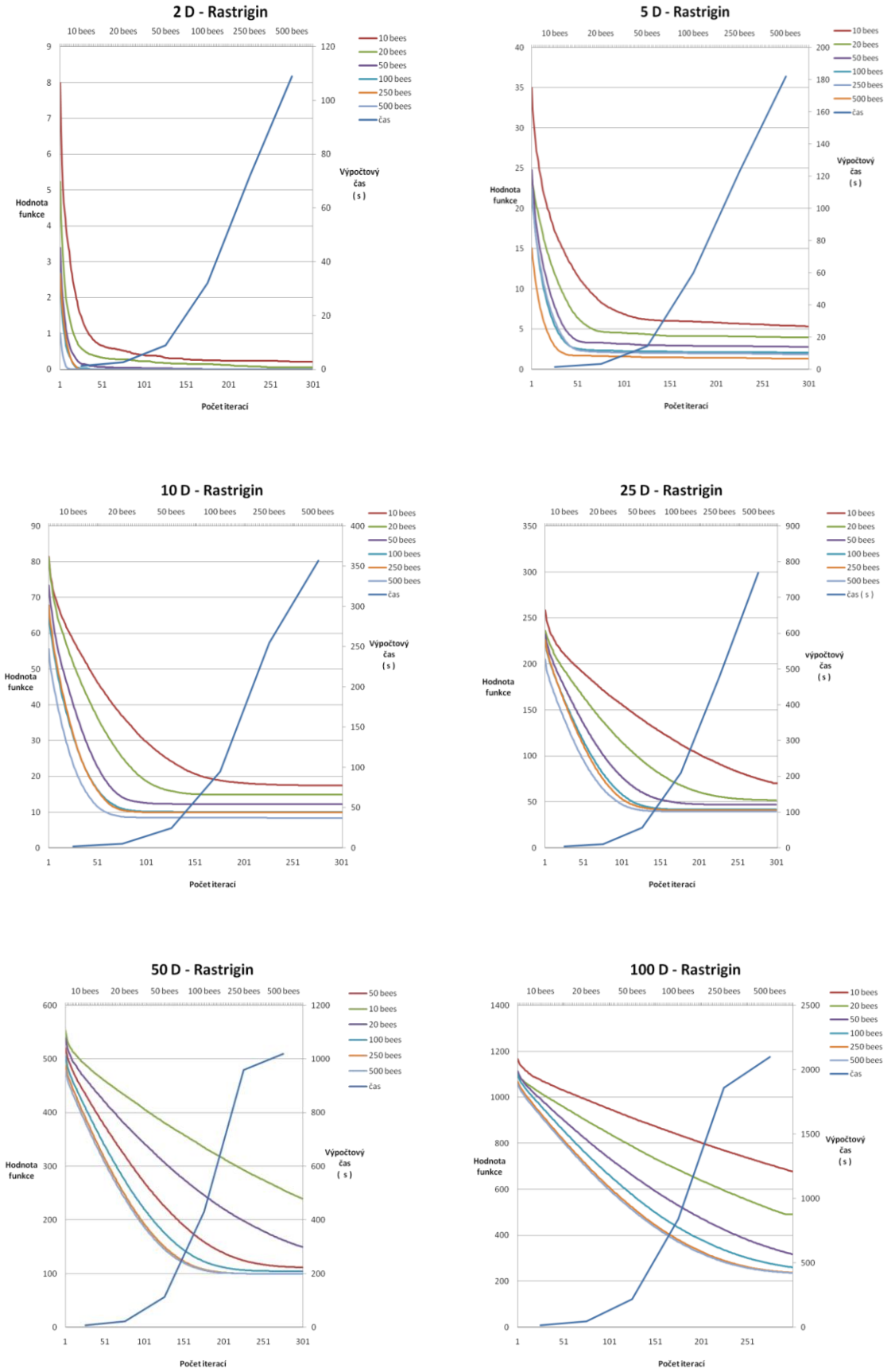
500 včel – elite places = 30, best places = 15, elite bees =30, best bees = 15

	2D	5D	10D	25D	50D	100D
Čas	1 min 49 s	3 min 2s	5 min 57s	12 min 49 s	17 min	35 min
Směr. odchylka	1,00721E-07	0,627167	1,931895	4,918395	7,761545	13,37861
Medián	9,36699E-08	0,996893	8,961649	39,88441	99,46113	238,43
Průměr	1,15943E-07	1,314382	8,328418	39,70074	99,22124	236,8549
Úspěšnost	100%	9%	0%	0%	0%	0%
Minimální hodnota	9,16495E-09	0,001436	2,991764	35,90771	94,07644	230,1324

4.3 Grafické srovnání

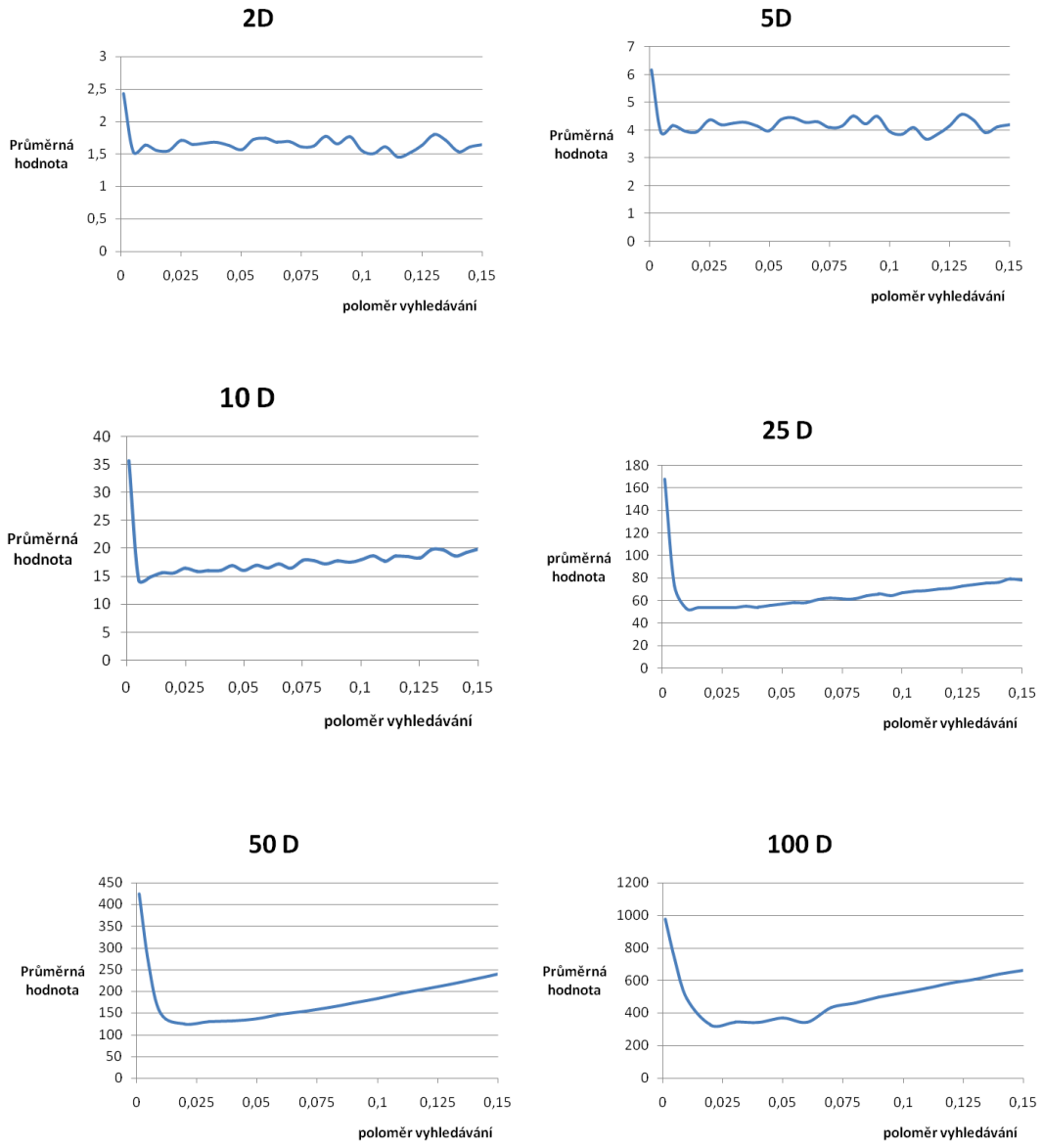


Obr. 8 Rosenbrock function



Obr. 9 Rastrigin function

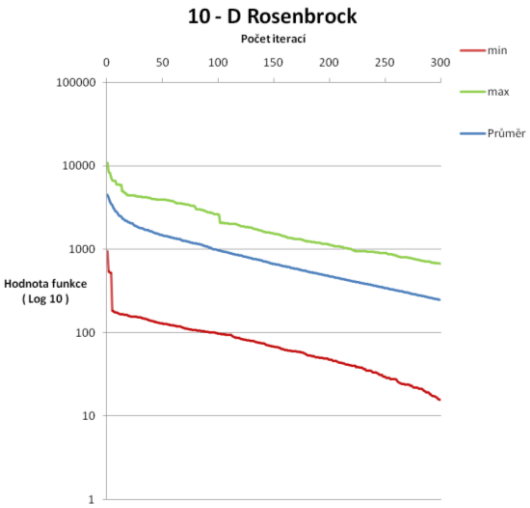
4.4 Vliv vyhledávajícího rádiusu



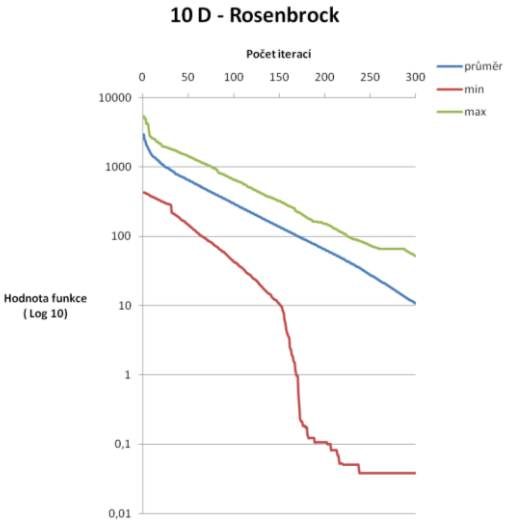
Obr. 8 Vyhledávání optimální hodnoty parametru ngh u funkce Rasrigin

Počet vzorků: 100,
Počet bees 20, elite bees 4, best bees 2, elite places 4, best places 2, ngh 0.001 až 0.15, krok v ngh 0.005. iterací 300.

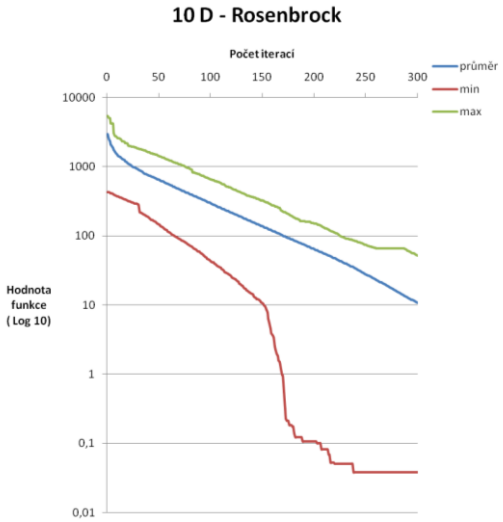
4.5 Minima a maxima funkce



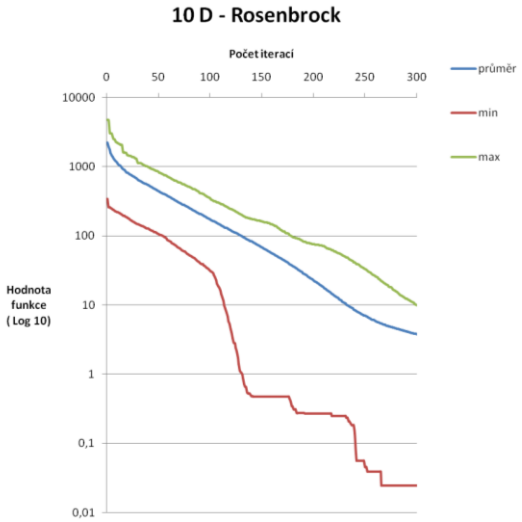
10 bees



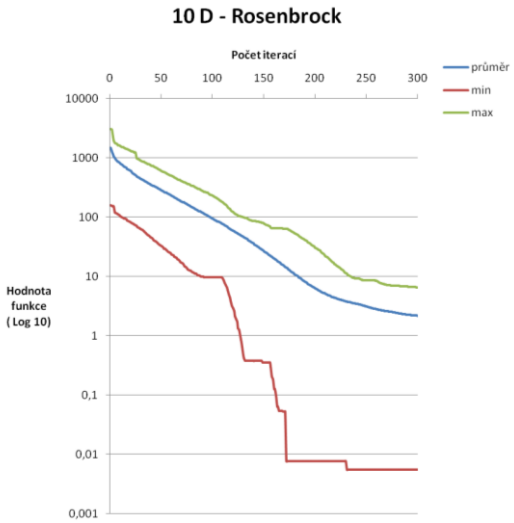
20 bees



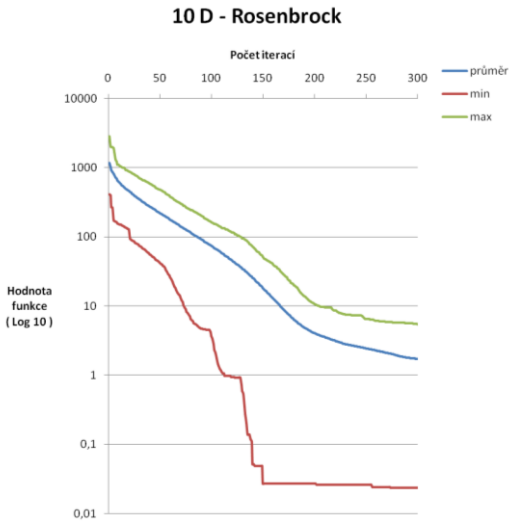
50 bees



100 bees



250 bees



500 bees

5. ZÁVĚR

Implementace algoritmu pro platformu Java a Matlab pro potřeby jeho testování na funkcích Rosenbrock a Rastrigin proběhla úspěšně. Naprogramovaný algoritmus byl otestován na platformě Java a výsledky ukazují, že i když se jedná o náhodné vyhledávání, hodnota mediánu funkce se velmi blíží hodnotě aritmetického průměru a jejich rozdíl se pohybuje kolem 2-3%. Další zajímavou vlastností je nelineární závislost času na efektivitě algoritmu.

Když zvýšíme počet včel dvojnásobně, výpočtový čas nestoupne dvakrát, ale čtyřikrát. A efektivita algoritmu vzroste asi o 5 - 50% v závislosti na počtu proměnných. Počet nejlepších výsledků, které se zvolí k dalšímu rozvoji by se měl pohybovat kolem 20 – 30, z důvodu obrovského nárůstu výpočetního času, pokud by se jejich počet volil jako procento z celkového počtu včel. Kdyby nebyl nastaven tento strop, výpočet pro 500 včel by trval přes 7 hodin.Poloměr vyhledávání okolo nejlepších výsledků byl měřen u funkce Rastrigin pro 10 proměnných, při měření se ukázala jako nejvýhodnější hodnota v okolí 0.01. pro funkci Rosenbrock je nejvýhodnější pokud možno co největší rádius.

Při měření se vyskytla neočekávaná chyba omezující počet vzorků, které je možné měřit. Výsledky jsou exportovány do xls souboru pro MS Excel. Počet sloupců v xls je na jednom listu maximálně 255. Touto hodnotou je tedy limitován počet vzorků. Při nastavení většího množství program proběhne v pořádku, grafy se vykreslí, ale pro následné statistické zpracování nejsou potřebná data. Ze stejného důvodu je omezen i počet iterací, které se ukládají do sloupců a jejich limit je 65 536. Toto omezení by se dalo odstranit lepším přepsáním třídy ToExcel, kde by se spočítal počet potřebných listů aby nedocházelo ke ztrátě dat.

Použitá literatura

[1] – HEYLIGHEN, Francis. *COLLECTIVE INTELLIGENCE AND ITS IMPLEMENTATION ON THE WEB: ALGORITHMS TO DEVELOP A COLLECTIVE MENTAL MAP*. [online]. 1999, [cit. 2010-05-03]. (Computational and Mathematical Organization Theory 5(3).) S. 253-280 Dostupné online Lang: (En)

http://pespmc1.vub.ac.be/Papers/CollectiveWebIntelligence.pdf

[2] – KARABOGA, Dervis, *AN IDEA BASED ON HONEY BEE SWARM FOR NUMERICAL OPTIMIZATION* [online] 2005 [cit. 2010-5-3] Dostupné online Lang (En)

http://mf.erciyes.edu.tr/abc/pub/tr06_2005.pdf

[3] – JÜRGEN TAUTZ. Fenomenální včely. 1. Vyd. Brázda 2009, 270 s. ISBN 978-80-209-0376-1

[4] – Web zaměřený na Bees Algorithm [online] Lang: (En)

http://www.bees-algorithm.com

[5] – PÍŠA, Pavel, MATLAB, LABORATOŘ NEJEN PRO MATEMATIKY [online], [cit. 2010-05-07]. Dostupné online Lang: (Cz)

http://cmp.felk.cvut.cz/~pisa/Public/ST_matlab.html

PŘÍLOHY:

CD s elektronickou verzí práce, Java aplikací a Matlab skriptu.