

Objevování trigonometrických identit pomocí umělé inteligence

Discovering of trigonometric identities using artificial intelligence

Bc. Tomáš Urbánek



Univerzita Tomáše Bati ve Zlíně

Fakulta aplikované informatiky

akademický rok: 2010/2011

ZADÁNÍ DIPLOMOVÉ PRÁCE

(PROJEKTU, UMĚLECKÉHO DÍLA, UMĚLECKÉHO VÝKONU)

Jméno a příjmení: **Bc. Tomáš URBÁNEK**

Osobní číslo: **A09493**

Studijní program: **N 3902 Inženýrská informatika**

Studijní obor: **Informační technologie**

Téma práce: **Objevování trigonometrických identit pomocí umělé inteligence**

Zásady pro vypracování:

1. Seznamte se s problematikou evoluční syntézy struktur.
2. Seznamte se s nástrojem Analytické programování.
3. Pomocí Analytického programování naleznete matematické vzorce pro trigonometrické identity.
4. Zpracujte pro výukové účely jednotlivé kroky hledání příslušných vzorců na několika příkladech.
5. Zpracujte závěr.

Rozsah diplomové práce:

Rozsah příloh:

Forma zpracování diplomové práce: **tištěná/elektronická**

Seznam odborné literatury:

1. ZELINKA, I., OPLATKOVÁ, Z., OŠMERA, P., ŠEDA, M., VČELAŘ, F. Evoluční výpočetní techniky – principy a aplikace. BEN – technická literatura, Praha, 2008, ISBN 80-7300-218-3.
2. ZELINKA, I., OPLATKOVÁ, Z., NOLLE, L., Boolean Symmetry Function Synthesis by Means of Arbitrary Evolutionary Algorithms-Comparative Study, International Journal of Simulation Systems, Science and Technology, Volume 6, Number 9, August 2005, pages 44 – 56, ISSN: 1473-8031, online <http://ducati.doc.ntu.ac.uk/uksim/journal/Vol-6/No.9/cover.htm>, ISSN: 1473-804x.
3. OPLATKOVÁ, Z.: Metaevolution – Synthesis of Optimization Algorithms by means of Symbolic Regression and Evolutionary Algorithms, Lambert-Publishing, 2009, ISBN 978-8383-1808-0.
4. KOZA J. R.: Genetic Programming: on the programming of computers by means of natural selection, The Bradford Book, MIT Press, UK, 1992, ISBN 0-262-11170-5.
5. ONEILL M., CONOR R.: Grammatical Evolution, Kluwer Academic Publishers, 2003, ISBN 1-4020-7444-1.
6. BANZHAF W. (ed.): Genetic Programming and Evolvable Machines, Vol. 7, Nr. 1, March, 2006, Springer, ISSN: 1389-2576.

Vedoucí diplomové práce:

Ing. Zuzana Oplatková, Ph.D.

Ústav informatiky a umělé inteligence

Datum zadání diplomové práce:

24. února 2011

Termín odevzdání diplomové práce:

18. května 2011

Ve Zlíně dne 24. února 2011

prof. Ing. Vladimír Vašek, CSc.
děkan



doc. Mgr. Roman Jašek, Ph.D.
ředitel ústavu

ABSTRAKT

Tato diplomová práce pojednává o možnostech využití metod umělé inteligence a to využití evoluční syntézy struktur. V teoretické části diplomové práce budou uvedeny metody evoluční syntézy struktur, jejich popis a možnosti použití. Dále budou v teoretické části zpracovány evoluční algoritmy SOMA a diferenciální evoluce, které budou použity pro praktickou část diplomové práce. V praktické části je použita metoda analytické programování pro vyhledávání matematických vzorců trigonometrických identit. Součástí práce je také zpracování programu pro výukové účely zobrazující evoluci na nalezených identitách.

Klíčová slova: SOMA, diferenciální evoluce, evoluční algoritmy, symbolická syntéza struktur, gramatická evoluce, analytické programování, genetické programování, trigonometrické identity

ABSTRACT

This thesis discusses the possibilities of using artificial intelligence methods and structures using evolutionary synthesis. The theoretical part will be the methods of evolutionary synthesis structures, their descriptions and potential applications. There will also be treated in the theoretical part of the evolutionary algorithms and differential evolution of SOMA, which will be used for the practical part. The practical part of the analytical method used to find programming of mathematical formulas trigonometric identities. The work also developed a program for educational purposes showing the evolution of identities found.

Keywords: SOMA, differential evolution, evolutionary algorithms, symbolic synthesis of structures, grammatical evolution, analytical programming, genetic programming, trigonometric identities

Na tomto místě bych rád poděkoval vedoucí mé diplomové práce Ing. Zuzaně Oplatkové, Ph.D. za odborné vedení a za čas, který věnovala mé práci. Poděkování patří také mým rodičům a mé přítelkyni za morální podporu a pomoc během psaní této diplomové práce.

Prohlašuji, že

- beru na vědomí, že odevzdáním diplomové/bakalářské práce souhlasím se zveřejněním své práce podle zákona č. 111/1998 Sb. o vysokých školách a o změně a doplnění dalších zákonů (zákon o vysokých školách), ve znění pozdějších právních předpisů, bez ohledu na výsledek obhajoby;
- beru na vědomí, že diplomová/bakalářská práce bude uložena v elektronické podobě v univerzitním informačním systému dostupná k prezenčnímu nahlédnutí, že jeden výtisk diplomové/bakalářské práce bude uložen v příruční knihovně Fakulty aplikované informatiky Univerzity Tomáše Bati ve Zlíně a jeden výtisk bude uložen u vedoucího práce;
- byl/a jsem seznámen/a s tím, že na moji diplomovou/bakalářskou práci se plně vztahuje zákon č. 121/2000 Sb. o právu autorském, o právech souvisejících s právem autorským a o změně některých zákonů (autorský zákon) ve znění pozdějších právních předpisů, zejm. § 35 odst. 3;
- beru na vědomí, že podle § 60 odst. 1 autorského zákona má UTB ve Zlíně právo na uzavření licenční smlouvy o užití školního díla v rozsahu § 12 odst. 4 autorského zákona;
- beru na vědomí, že podle § 60 odst. 2 a 3 autorského zákona mohu užít své dílo – diplomovou/bakalářskou práci nebo poskytnout licenci k jejímu využití jen s předchozím písemným souhlasem Univerzity Tomáše Bati ve Zlíně, která je oprávněna v takovém případě ode mne požadovat přiměřený příspěvek na úhradu nákladů, které byly Univerzitou Tomáše Bati ve Zlíně na vytvoření díla vynaloženy (až do jejich skutečné výše);
- beru na vědomí, že pokud bylo k vypracování diplomové/bakalářské práce využito softwaru poskytnutého Univerzitou Tomáše Bati ve Zlíně nebo jinými subjekty pouze ke studijním a výzkumným účelům (tedy pouze k nekomerčnímu využití), nelze výsledky diplomové/bakalářské práce využít ke komerčním účelům;
- beru na vědomí, že pokud je výstupem diplomové/bakalářské práce jakýkoliv softwarový produkt, považují se za součást práce rovněž i zdrojové kódy, popř. soubory, ze kterých se projekt skládá. Neodevzdání této součásti může být důvodem k neobhájení práce.

Prohlašuji,

- že jsem na diplomové práci pracoval samostatně a použitou literaturu jsem citoval. V případě publikace výsledků budu uveden jako spoluautor.
- že odevzdaná verze diplomové práce a verze elektronická nahraná do IS/STAG jsou totožné.

Ve Zlíně

.....
podpis diplomanta

OBSAH

ÚVOD.....	9
I TEORETICKÁ ČÁST	10
1 ZÁKLADNÍ POJMY Z OBLASTI EVOLUČNÍCH ALGORITMŮ	11
1.1 JEDINEC	11
1.2 POPULACE	11
1.3 GENERACE	12
1.4 ÚČELOVÁ FUNKCE	12
1.5 EVOLUČNÍ CYKLUS	12
2 EVOLUČNÍ ALGORITMY	13
2.1 SOMA	13
2.1.1 Parametry SOMY	13
2.1.2 Popis funkce algoritmu SOMA	14
2.2 DIFERENCIÁLNÍ EVOLUCE	15
2.2.1 Parametry diferenciální evoluce	15
2.2.2 Popis funkce algoritmu diferenciální evoluce	15
3 METODY EVOLUČNÍ SYNTÉZY STRUKTUR	17
3.1 GENETICKÉ PROGRAMOVÁNÍ	17
3.1.1 Popis genetického programování	18
3.2 GRAMATICKÁ EVOLUCE.....	21
3.2.1 Backus-Naurova forma.....	21
3.2.2 Popis gramatické evoluce.....	21
3.2.3 Paralelní gramatická evoluce.....	23
3.3 ANALYTICKÉ PROGRAMOVÁNÍ.....	24
3.3.1 Princip Analytického programování	24
4 TRIGONOMETRICKÉ IDENTITY.....	26
II PRAKTICKÁ ČÁST	27
5 POUŽITÝ SOFTWARE	28
5.1 MATHEMATICA.....	28
5.2 ANALYTICKÉ PROGRAMOVÁNÍ.....	29
6 ZPRACOVÁNÍ ÚLOH	32
6.1 POSTUP.....	32
6.1.1 Účelová funkce.....	32
6.1.2 Výsledky.....	33
6.2 ÚLOHY SOMA	34
6.2.1 Úloha číslo 1	34
6.2.2 Úloha číslo 2	36
6.2.3 Úloha číslo 3	39

6.2.4	Úloha číslo 4	41
6.2.5	Úloha číslo 5	44
6.3	ÚLOHY DIFERENCIÁLNÍ EVOLUCE.....	47
6.3.1	Úloha číslo 1	47
6.3.2	Úloha číslo 2	49
6.3.3	Úloha číslo 3	52
6.3.4	Úloha číslo 4	54
6.3.5	Úloha číslo 5	57
7	VÝUKOVÝ SOFTWARE.....	60
7.1	POPIS APLIKACE.....	60
7.2	OBSLUHA APLIKACE	61
	ZÁVĚR	62
	ZÁVĚR V ANGLIČTINĚ.....	63
	SEZNAM POUŽITÝCH SYMBOLŮ A ZKRATEK	66
	SEZNAM OBRÁZKŮ	67
	SEZNAM TABULEK.....	69
	SEZNAM PŘÍLOH.....	70

ÚVOD

Stejně jako většina metod umělé inteligence, které jsou založeny na pozorování mechanismu přírody, jsou i evoluční syntézy struktur založeny na biologické evoluční teorii a přírodním výběru. Zakladatelem této teorie je Charles Darwin. Z této teorie vznikla celá řada metod a algoritmů, které jsou schopny řešit velmi složité problémy, jenž mnohdy nelze vyřešit v reálném čase. Vzpomeňme například evoluční algoritmy typu SOMA, popřípadě diferenciální evoluce, které se velmi úspěšně používají k nalézání globálních extrémů zadaných problémů.

Metody evoluční syntézy struktur se využívají k nalezení řešení problémů, které jsou složité nebo k nalezení alternativních řešení daných problémů. Tato práce vysvětluje čtenáři, jakým způsobem se s těmito algoritmy umělé inteligence pracuje. V této diplomové práci jsou objasněny a popsány základní algoritmy evoluční syntézy struktur, ale také dva evoluční algoritmy, a to SOMA a diferenciální evoluce. Evoluční syntézy struktur jsou mocným nástrojem při objevování nových řešení. Nicméně nemají jen samé výhody. Jednou z nevýhod je časová a výpočetní náročnost u složitějších problémů. Stejně jako v případě evolučních algoritmů lze nacházet řešení problémů, u kterých dokážeme popsat vhodnost jednotlivých řešení a jak moc se toto řešení liší od řešení, které akceptujeme jako vhodné.

Tato práce si klade za cíl prověřit metody evoluční syntézy struktur, hlavně schopnost analytického programování objevit k trigonometrické identitě identitu alternativní. Jednotlivé zadané trigonometrické identity budou postupně hledány s určitými nastavenými parametry. Každá identita bude hledána desetkrát, aby se zajistila rozmanitost a objektivita nalezených identit. Vypočtená data budou prezentována v této diplomové práci, a dále budou zpracována do výukového softwaru, který je naprogramován v softwaru Mathematica. Tato aplikace bude sloužit k prezentaci dosažených výsledků při hledání trigonometrických identit. Součástí výukové aplikace bude také možnost vizualizovat průběh výpočtu jednotlivých identit na zobrazených grafech.

I. TEORETICKÁ ČÁST

1 ZÁKLADNÍ POJMY Z OBLASTI EVOLUČNÍCH ALGORITMŮ

V oblasti evolučních algoritmů jsou zažité pojmy a označení, kterých se tato diplomová práce bude držet. Tyto základní pojmy jsou důležitou částí pro pochopení funkce evolučních algoritmů a evoluční syntézy struktur.

1.1 Jedinec

Jedinec je z pohledu evolučních algoritmů chápán jako jedno řešení daného problému. Jedinec má určitý počet parametrů. Počet těchto parametrů určuje také dimenzi problému. Každý jedinec je pomocí účelové funkce ohodnocen, tzn. že získá určitou vhodnost. Vhodnost říká, jak vhodní jsou jedinci pro výběr do další populace tak, aby byli do další populace vybráni jen kvalitnější jedinci.

1.2 Populace

Populace je typickým znakem evolučních algoritmů. Většina evolučních algoritmů využívá populaci jako matici $M \times N$. Jde vlastně o tabulku, kde sloupce tabulky představují jedince a řádky představují jednotlivé parametry viz Tabulka 1. Mezi řádky populace se vyskytuje také tzv. vhodnost. Vhodnost je parametr, který danému evolučnímu algoritmu sděluje, jak je ten který jedinec kvalitní, a jestli bude připuštěn do další generace.

	Jedinec 1	Jedinec 2	...	Jedinec N
Vhodnost (fitness)	22,6	42,8	...	50,3
Parametr 1	12,4	11,5	...	10,2
Parametr 2	65,7	73,8	...	84,3
...
Parametr N	10,3	11,7	...	15,8

Tabulka 1 : Ukázka populace

1.3 Generace

Generací se označuje cyklus průchodu populace celým evolučním cyklem. Jako první generace je označován cyklus, kde je prvotně a náhodně vygenerována populace jedinců. Po dokončení evolučního cyklu nastoupí nová generace kvalitnějších jedinců. Generací se tudíž označuje iterace evolučního cyklu. V případě algoritmu SOMA je jedna iterace označována jako migrační kolo.

1.4 Účelová funkce

Na účelové funkci stojí a padají evoluční algoritmy. K tomu, abychom byli schopni určit, kteří jedinci jsou vhodnější než jiní, nám slouží právě účelová funkce. Účelová funkce je vlastně určitá geometrická plocha, ve které pomocí evolučních algoritmů hledáme extrém, tzn. minimum popř. maximum. Každý jedinec je ohodnocen účelovou funkcí a získá tzv. vhodnost. Do další generace postupují jen kvalitnější jedinci.

1.5 Evoluční cyklus

Každý evoluční algoritmus se řídí určitým cyklem, typickým právě pro evoluční algoritmy.

- Definice parametrů vybraného evolučního algoritmu
- Vygenerování první náhodné populace
- Ohodnocení jedinců této prvotní populace
- Výběr jedinců do nové populace
- Operace křížení a mutace
- Ohodnocení jedinců populace
- Výběr nejkvalitnějších jedinců do nové populace
- Ukončení algoritmu

Po ukončení evolučního algoritmu získáme nejkvalitnější jedince, a tudíž řešení našeho zadaného problému [1] .

2 EVOLUČNÍ ALGORITMY

Evolučních algoritmů je velké množství. Vzhledem k tomu, že se tato diplomová práce zaměřuje na metody evoluční syntézy struktur, uvedu zde jen popis dvou vybraných algoritmů, a to Diferenciální evoluce a SOMA.

2.1 SOMA

SOMA je zkratka ze slovního spojení samoorganizující se migrační algoritmus. Činnost algoritmu je založena na vektorových operacích. Tento algoritmus byl vyvinut v roce 1999. Činnost algoritmu je založena na populaci jedinců, jako je tomu i u jiných evolučních algoritmů, avšak u SOMY nevznikají noví potomci, ale stávající se přemísťují po prostoru daného problému. Toto přemístění se děje v rámci jedné generace, u SOMY nazývané migrační kolo. Přesnější zařazení algoritmu je mezi algoritmy memetické či hejnové [1,7].

Hlavní myšlenka tohoto algoritmu je spolupráce mezi jedinci, kteří řeší společný problém, v našem případě hledají minimum resp. maximum daného problému. Podobné chování lze vidět v přírodě např. u mravenců, včel apod.

2.1.1 Parametry SOMY

Nejdůležitější je nastavit parametry SOMY, se kterými se bude pracovat během evoluce. Pokusy bylo zjištěno, že SOMA je velmi citlivá na počáteční nastavení parametrů. Proto je správné nastavení parametrů klíčovým momentem pro běh evoluce a dosažení správných výsledků.

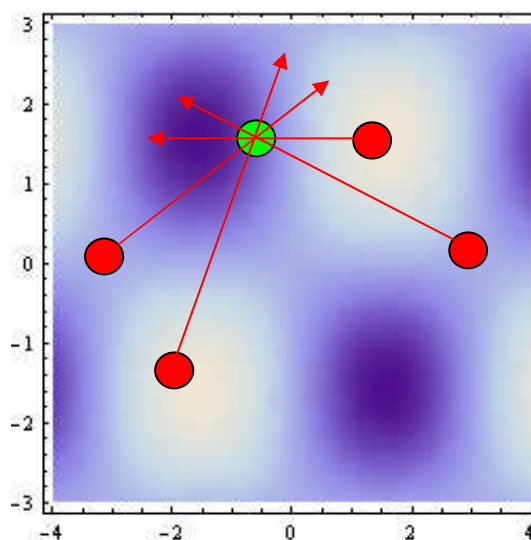
- PathLength – parametr určuje, v jaké vzdálenosti se jedinec zastaví od vedoucího jedince
- Step – parametr určuje, s jakou délkou kroku bude jedinec postupovat k vedoucímu jedinci
- PRT – jeden z nejdůležitějších parametrů ovlivňující citlivost algoritmu. Podle této hodnoty se jedinec rozhoduje, zda bude postupovat přímo k vedoucímu či nikoliv
- D – parametr určuje počet argumentů účelové funkce
- PopSize – parametr určuje, kolik jedinců bude vytvořeno v populaci

- Migrate – parametr určuje, kolik proběhne tzv. migračních kol, neboli kolik iterací bude uskutečněno
- MinDiv – parametr se používá pro ukončení algoritmu, definuje hodnotu rozdílu mezi nejlepším a nejhorším jedincem v populaci, který je povolen

2.1.2 Popis funkce algoritmu SOMA

Jak již bylo uvedeno, algoritmus pracuje v tzv. migračních kolech. Je to náhradní termín za iteraci algoritmu. U SOMY nejsou tvořeni noví jedinci. Filozofie algoritmu je postavena na pohybu jedinců po ploše problému, kde hledají extrém. Jedinci se pohybují k tzv. leaderovi (vedoucímu), což je jedinec s nejnižší hodnotou účelové funkce. I v tomto algoritmu nalezneme mutaci a křížení, ovšem v pozměněné formě. Mutace je v tomto algoritmu nazývána jako perturbace, a slouží ke změně směru pohybu jedince po ploše. Perturbace se řídí parametrem PRT popsáným výše. Vygeneruje se náhodné číslo od 0-1 a následně se porovná s parametrem PRT. Jestliže je číslo menší než PRT, do perturbačního vektoru uloží číslo 1, jinak 0. Perturbační vektor je použit při křížení, které je reprezentováno rovnicí směrového vektoru (1).

$$\vec{r} = \vec{r}_0 + m \cdot t \cdot PRT \vec{Vector} \quad (1)$$



Obrázek 1 : Ukázka migrace jedinců k leaderovi algoritmu SOMA

V této rovnici hraje úlohu PRTVector jako určitý druh mutace. Ze směrové rovnice se vypočítá směr pohybu jedince. Jedinec se pohybuje dle rovnice ke svému leaderovi nebo

v jiném směru po diskrétních krocích. Jedinec tak vytvoří několik svých potomků ve směru své cesty. Každý potomek je ohodnocen, a je vybrán nejlepší z nich. Každý jedinec v generaci takto cestuje po ploše zadaného problému a hledá extrém. V každém takovém cyklu dojde k několikanásobnému ohodnocení účelové funkce [1,7].

2.2 Diferenciální evoluce

Diferenciální evoluce vznikla v roce 1995 a jejími autory jsou Ken Price a Rainer Storm. Diferenciální evoluce sdílí určité aspekty s genetickými algoritmy, jako je např. tvorba potomků, generace apod. Diferenciální evoluce byla vyvinuta z genetického žíhání úpravou mutace, která se nazývá diferenciální mutace. První verze diferenciální evoluce nedostačovaly pro použití na široké množině optimalizačních úloh. Až třetí verze algoritmu diferenciální evoluce byla přijata s kladným ohlasem [1,10,14].

2.2.1 Parametry diferenciální evoluce

Algoritmus diferenciální evoluce se řídí pouze čtyřmi parametry. U diferenciální evoluce je nutné dávat si pozor na jev zvaný „stagnace“. Stagnace je jednou z největších nevýhod diferenciální evoluce. V podstatě jde o zastavení vývoje hodnot účelové funkce k nižším hodnotám, než je dosaženo globálního extrému. Aby bylo možné stagnaci minimalizovat, je třeba věnovat pozornost nastavení parametrů diferenciální evoluce. Těmito parametry jsou :

- NP – parametr udává velikost vytvořené populace
- F – parametr mutační konstanty
- CR – parametr určuje křížení v populaci a nazývá se práh křížení
- Generations – parametr určuje, kolik proběhne iterací algoritmu než bude ukončen

2.2.2 Popis funkce algoritmu diferenciální evoluce

Jako u všech podobných algoritmů je nejprve vytvořena prvotní generace. Počet vytvořených jedinců určuje parametr NP. Pro evoluční proces jsou vybráni čtyři jedinci. Jedinci vstupují do evolučního procesu mutací. Mutace probíhá ze tří náhodně vybraných jedinců. Mutace vytváří tzv. šumový vektor. Tento vektor je tvořen pomocí vzorce (2).

Rozdíl dvou náhodně vybraných jedinců je vynásoben mutační konstantou F , a tento výsledek je přičten k třetímu, zatím nepoužitému jedinci.

$$v_j = x_{r3,j}^G + F \cdot (x_{r1,j}^G - x_{r2,j}^G) \quad (2)$$

Při křížení hraje důležitou roli právě šumový vektor a náš čtvrtý, nepoužitý jedinec. Pro každou položku vektoru se vygeneruje náhodné číslo od 0-1. Toto číslo se porovná s parametrem CR . Jestliže je číslo menší než parametr CR , do nového vektoru, tzv. zkušebního vektoru, se vybere položka ze šumového vektoru. Jestliže je vygenerované číslo naopak větší, do šumového vektoru se vybere položka ze čtvrtého jedince.

CR=0,8			
Šumový vektor	Náhodné číslo	Čtvrtý jedinec	Zkušební vektor
12,69	0,31<0,8	14,75	12,69
5,39	0,58<0,8	-6,32	5,39
-7,54	0,96>0,8	2,94	2,94
1,38	0,85>0,8	-4,35	-4,35
9,65	0,47<0,8	7,26	9,65

Tabulka 2 : Ukázka křížení diferenciální evoluce

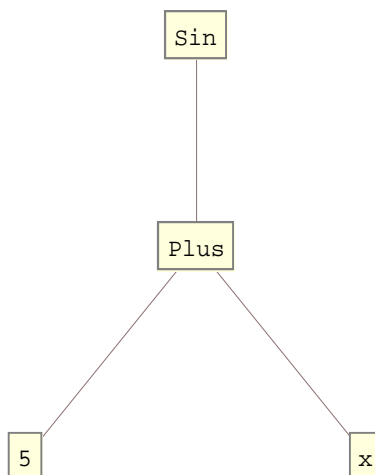
Jedinci dále vstupují do procesu ohodnocení účelovou funkcí. Každému jedinci je určena jeho vhodnost. Po určení vhodnosti všech jedinců jsou vybráni jedinci, kteří vstoupí jako rodiče do nové populace. Tato populace opět projde procesy mutací, křížení, ohodnocení a výběrem do nové populace, dokud není algoritmus zastaven počtem iterací, který udává parametr Generations [1,10,14].

3 METODY EVOLUČNÍ SYNTÉZY STRUKTUR

Postupem času bylo vyvinuto několik metod umělé inteligence, které nehledají jen parametry regresní funkce, ale jsou schopny získat samotnou regresní funkci. Důležitými pojmy v této oblasti umělé inteligence jsou pojmy terminální a neterminální symboly.

Neterminální symboly jsou funkce, ať už matematické, či uživatelsky definované. Každá funkce má nějaké své parametry. Tím se odlišuje od terminálního symbolu, který žádné parametry nemá. Z toho vyplývá, že do parametru neterminálního symbolu můžeme přiřadit další neterminální symbol, popř. terminální symbol.

Terminální symboly jsou symboly konečné a nelze je nadále dělit. Terminální symboly jsou parametry funkcí. Mohou to být proměnné, případně konstanty.



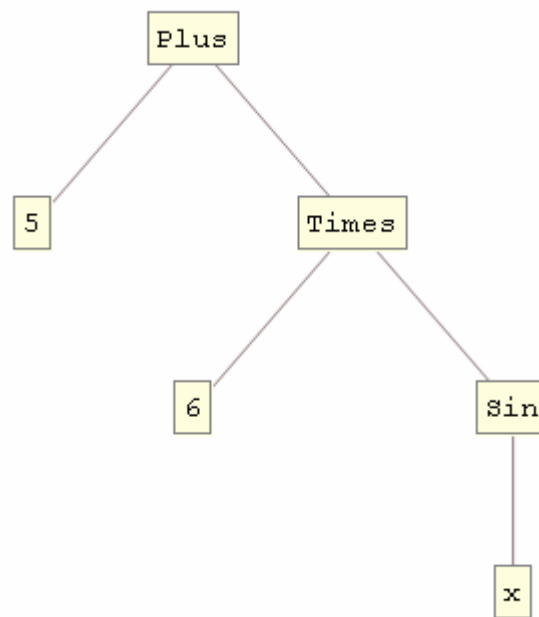
Obrázek 2 : Ukázka terminálních a neterminálních symbolů výrazu $\text{Sin}(5 + x)$

3.1 Genetické programování

Autorem genetického programování je John Koza, který v 80. letech představil tuto metodu. John Koza původně programoval metodu genetické programování ve funkcionálním programovacím jazyce LISP. Hlavní myšlenkou je, že počítač je schopný prohledat prostor možných řešení tak, že dostaneme určité řešení, které bude vyhovovat zadaným podmínkám. Počítač tedy navrhne program, který řeší zadaný problém. Tyto programy jsou šlechtěny pomocí symbolických struktur. To znamená, že programy, funkce a operátory jsou zapsány jako určité symboly a spojeny v určité struktuře. Touto strukturou je u klasického genetického programování datová struktura zvaná strom.[1,4,6]

3.1.1 Popis genetického programování

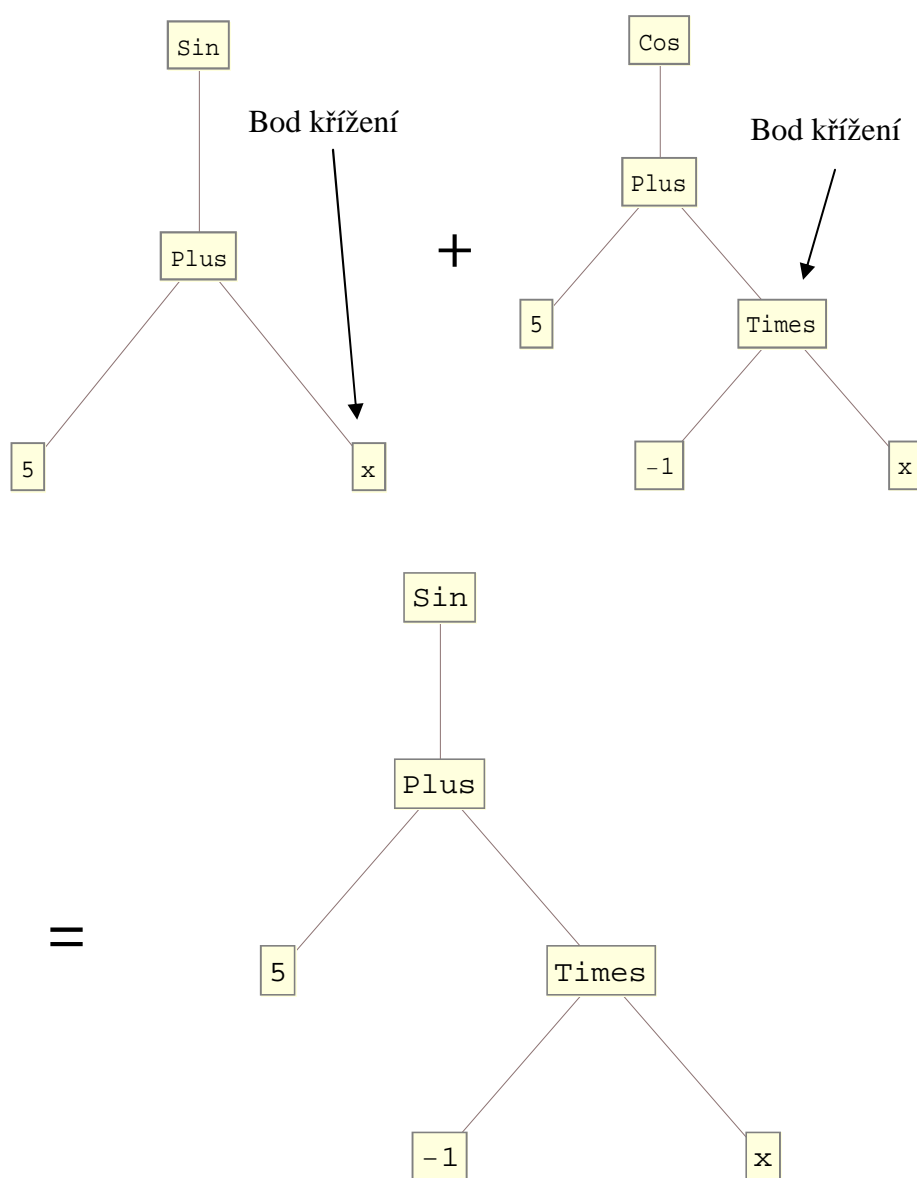
Nejvyšší člen stromu se nazývá kořen stromu. Členy, které se dále nedělí, se nazývají listy stromu. Členy, na které jsou připojeny listy stromu, se nazývají uzly stromu. Listy stromu jsou označovány jako terminální symboly, tedy dále nedělitelné. Představují parametry funkce. Uzly stromu a kořen stromu jsou označovány jako neterminální symboly. Představují samotné funkce programu.



Obrázek 3 : Příklad datové struktury stromu pro výraz $5 + 6 \sin(x)$

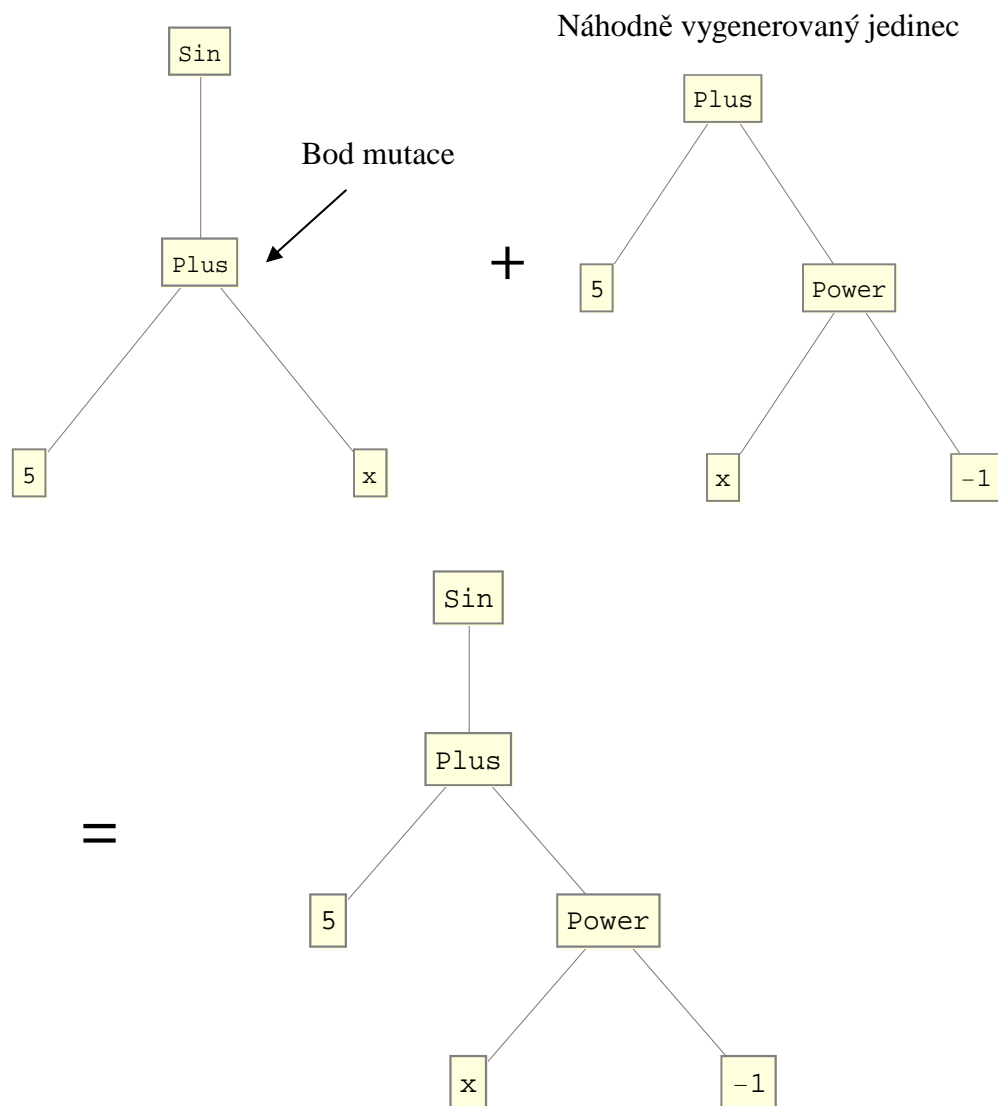
Celý proces začíná, stejně jako u většiny evolučních algoritmů, vygenerováním prvotní populace jedinců. Získáme tedy populaci náhodných stromů. Z této populace se náhodně vyberou dva jedinci, u kterých následně dochází ke křížení a mutaci. Dále jsou tyto noví jedinci ohodnoceni účelovou funkcí. Do další generace vstupují kvalitnější jedinci.[1,4,6]

Křížení probíhá tak, že se vyberou body křížení. V bodech křížení si dva jedinci vymění své podstromy. Stejně jako u jiných evolučních algoritmů, i zde hraje roli tzv. pravděpodobnost křížení. Tento parametr určuje, zda dojde ke křížení, či nikoliv.



Obrázek 4 : Ukázka křížení genetického programování

Mutace probíhá podobným způsobem, avšak zde se vybere jeden náhodný jedinec. Určí se bod mutace, a v tomto bodě bude podstrom nahrazen nově vygenerovaným podstromem. I zde existuje parametr, který ovlivňuje, zda mutace nastane, či nikoliv. Tento parametr se nazývá pravděpodobnost mutace.



Obrázek 5 : Ukázka mutace genetického programování

Důležitou roli v této metodě hraje parametr maximální hloubky stromu D_{\max} . Tento parametr genetického programování nám říká, jakou hloubku stromu maximálně může mít jedinec. Pozorný čtenář si mohl povšimnout faktu, že ne vždy se při křížení nebo mutaci vyměňují, resp. nahrazují stejně hluboké podstromy. To znamená, že se může stát, že výsledný jedinec bude hlubší, než je naše maximální zadaná hloubka D_{\max} . Tento efekt se nazývá Bloat (z anglického výrazu pro nafouknutí). Jednou z možností řešení tohoto problému je penalizace dlouhých řešení tak, že tyto jedinci se nebudou vyskytovat v další generaci.[1,4,6]

3.2 Gramatická evoluce

Je to metoda genetického programování založená na gramatice. Pro tuto metodu se využívá tzv. Backus-Naurova forma, která nám dovoluje generovat programy v libovolném jazyce. Hlavní výhodou oproti genetickému programování je možnost generovat víceřádkové funkce v libovolném jazyce. Gramatická evoluce rovněž využívá jinou reprezentaci jedinců. Zde je jedinec reprezentován pomocí posloupnosti celých čísel. [1,5,17]

3.2.1 Backus-Naurova forma

Backus-Naurova forma je metasyntaxe používaná pro generování bezkontextové gramatiky. Autory této formy byli pánové John Backus a Peter Naur. Tato forma nám sděluje, jakými způsoby je možno vytvořit gramaticky správné a legální výrazy. Backus-Naurova forma se skládá z terminálních a neterminálních symbolů. [1,2,16]

3.2.2 Popis gramatické evoluce

Základem gramatické evoluce je gramatika. Gramatikou budeme nazývat čtveřici (3).

$$G = \{N, T, P, S\} \quad (3)$$

,kde

N – konečná množina neterminálních symbolů

T – konečná množina terminálních symbolů, kdy $N \cap T = \emptyset$

P – množina přepisovacích pravidel

S – počáteční symbol, kdy $S \in N$

Tedy, když je definována gramatika, je potřeba zapsat množinu přepisovacích pravidel. K tomuto účelu bude sloužit Backus-Naurova forma. Přepisovací pravidla se zapisují ve tvaru (4) :

$$\langle \text{symbol} \rangle ::= \langle \text{možnost 1} \rangle | \langle \text{možnost 2} \rangle | \langle \text{možnost 3} \rangle | \dots | \langle \text{možnost n} \rangle \quad (4)$$

Gramatická evoluce kóduje jedince do binárního řetězce. Osmibitové sekvence, tedy jeden bajt, je označován jako tzv. kodon. Samozřejmostí je, že maximální hodnota

kodonu je 255. [13] Aby bylo možné vybrat z binárního řetězce přepisovací pravidlo, byl zaveden vztah (5) :

Pravidlo = integer hodnoty kodonu MOD počet možností aktuálního neterminálu (5)

,kde MOD je matematická operace celočíselného dělení modulo.

Symbol	Možnosti	Pořadí
<expr>::=	<expr><op><expr>	0
	(<expr><op><expr>)	1
	<var>	2
<op>::=	+	0
	-	1
	*	2
	/	3
<var>::=	x	0
	1	1

Tabulka 3 : Ukázka tabulky přepisovacích pravidel

Celý proces začíná vygenerováním jedince. Tento jedinec je sestaven z tzv. kodonů popsaných výše. Jedinec se tedy skládá z většího počtu osmibitových čísel.

6	119	152	53	215	57	98	245	8	23	78	158	253	39	...
---	-----	-----	----	-----	----	----	-----	---	----	----	-----	-----	----	-----

Tabulka 4 : Ukázka jedince metody gramatické evoluce

Každý z kodonů je vlastně takový ukazatel do množiny přepisovacích pravidel. Začíná se obvykle prvním neterminálním symbolem. Tomuto symbolu je pomocí prvního kodonu jedince a vztahu (3) vybrán symbol, ať už terminální nebo neterminální. Jestliže je

vybrán symbol terminální, jedinec je hotov a postupuje do účelové funkce k ohodnocení. Jestliže je vybrán neterminální symbol, pokračuje se zleva doprava v postupné úpravě jednotlivých neterminálních symbolů. Tento proces pokračuje, dokud nejsou všechny symboly výrazu terminálními symboly. Jestliže dojdou kodony jedince, pokračuje se od začátku jedince, dokud není jedinec upraven a všechny jeho symboly jsou terminální.

Výraz	Kodon	Vybráno z tabulky přepisovacích pravidel
$\langle \text{expr} \rangle$	6 MOD 3 = 0	$\langle \text{expr} \rangle \langle \text{op} \rangle \langle \text{expr} \rangle$
$\langle \text{expr} \rangle \langle \text{op} \rangle \langle \text{expr} \rangle$	119 MOD 3 = 2	$\langle \text{var} \rangle$
$\langle \text{var} \rangle \langle \text{op} \rangle \langle \text{expr} \rangle$	152 MOD 2 = 0	x
$x \langle \text{op} \rangle \langle \text{expr} \rangle$	53 MOD 4 = 1	-
$x - \langle \text{expr} \rangle$	215 MOD 3 = 2	$\langle \text{var} \rangle$
$x - \langle \text{var} \rangle$	57 MOD 2 = 1	1
$x - 1$		

Tabulka 5 : Ukázka práce gramatické evoluce

Potom, co je jedinec upraven, je ohodnocena jeho kvalita účelovou funkcí. Takto se zpracují i ostatní jedinci. Stejně jako v jiných algoritmech, dochází před ohodnocením účelovou funkcí taktéž k mutacím a křížením, a až po té je jedinec sestavený z kodonů upraven a ohodnocen účelovou funkcí [1,5,17].

3.2.3 Paralelní gramatická evoluce

Stejně jako většina ostatních evolučních algoritmů a metod byla inspirována přírodou, je tomu tak i v případě paralelního výpočtu gramatické evoluce. Pojmem paralelní se zde uvažuje v tom smyslu, že jedinci jsou rozděleni mezi několik podmnožin. Paralelizace výpočtů vede k jejich rychlejšímu provádění a je možné řešit složitější problémy. Mohou být použity tři základní modely :

- „Farmářský model“
- Migrační model

- Difusní model

Jednotlivé modely popsány výše se liší jen v propojení jedinců mezi různými skupinami struktur. U paralelní gramatické evoluce existuje pojem migrace, kdy se pomocí migrace vnáší informace z cizího genového fondu, do jiného genového fondu [1,5,17].

3.3 Analytické programování

Analytické programování na rozdíl od ostatních metod evoluční syntézy struktur nevyužívá žádné reprezentace dat typu strom nebo gramatiky. Analytické programování lze chápat jako transformaci množiny dat, která bude předávána evolučnímu algoritmu, např. SOMA nebo diferenciální evoluce, k ohodnocení a řízení další evoluce. Využívá se schopnosti evolučních algoritmů pracovat s diskrétními hodnotami [1].

3.3.1 Princip Analytického programování

Stejně jako v ostatních algoritmech evoluční syntézy struktur se používají terminální a neterminální symboly. Všechny použité funkce se setřídí podle počtu jejich parametrů, tzn. např. funkce $\sin()$ obsahuje jeden parametr, funkce $+$ obsahuje dva parametry. Tímto získáme množinu funkcí, kde funkce s vyšším počtem parametrů jsou nadmnožinou funkcí s nižším počtem parametrů. Terminální symboly jako proměnné a konstanty jsou podmnožinou všech ostatních funkcí. Nemají žádný parametr.

Proces evoluce využívá schopnosti evolučních algoritmů pracovat s diskrétními hodnotami. Každý jedinec se skládá z celočíselných hodnot, které jsou ukazatelem do tabulky terminálních a neterminálních symbolů. Do účelové funkce tak vstupuje ne celočíselná reprezentace jedince, ale sestavený jedinec dle celočíselné reprezentace z tabulky symbolů.

1	5	6	7	8	8
---	---	---	---	---	---

Tabulka 6 : Ukázka jedince analytického programování

Symbol	+	-	*	/	$\sin()$	$\cos()$	$\tan()$	x	π
Počet parametrů	2	2	2	2	1	1	1	0	0

Tabulka 7 : Ukázka setříděných terminálních a neterminálních symbolů

Parametr 1: 5	Parametr 1: 7	Parametr 1: 8	Parametr 1: 8		
1: +	5: $\sin()$	6: $\cos()$	7: $\tan()$	8: x	8: x
Parametr 2: 6					

Tabulka 8 : Ukázka ukazatelů na jednotlivé symboly

Výraz
$Parametr1 + Parametr2$
$\sin(Parametr1) + Parametr2$
$\sin(Parametr1) + \cos(Parametr1)$
$\sin(\tan(Parametr1)) + \cos(Parametr1)$
$\sin(\tan(Parametr1)) + \cos(x)$
$\sin(\tan(x)) + \cos(x)$

Tabulka 9 : Ukázka sestavování výrazu analytického programování

Jakmile je jedinec sestaven, je ohodnocen účelovou funkcí některého z použitých evolučních algoritmů, např. SOMA nebo DE. Výhodou analytického programování je také vyplývající nezávislost na použití evolučního algoritmu. Jak již bylo naznačeno výše, analytické programování je určitým druhem transformace z celočíselného jedince na sestavený výraz pomocí terminálních a neterminálních symbolů [1].

4 TRIGONOMETRICKÉ IDENTITY

Trigonometrie je obor matematiky zabývající se goniometrickými funkcemi, které jsou využity pro výpočty v trojúhelníku. Trigonometrie se dříve využívala zejména v astronomii. Trigonometrické operace se taktéž používají k zjišťování délek a úhlů, které mohou být použity k tzv. triangulaci, což je určování pozice objektu.

V trigonometrických operacích se používají tzv. goniometrické funkce. Těmito funkcemi jsou např. známé funkce $\sin(x)$, $\cos(x)$, $\tan(x)$ a tak dále. Jednou z nejznámějších goniometrických funkcí je funkce $\sin(x)$, k níž se váže tzv. „Sinová věta“ (6)

$$\frac{\alpha}{\sin(\alpha)} = \frac{\beta}{\sin(\beta)} = \frac{\gamma}{\sin(\gamma)} \quad (6)$$

Trigonometrickou identitou je matematický vzorec, který je ekvivalentní hledanému matematickému vzorci, např. $\tan(x) = \frac{\sin(x)}{\cos(x)}$. Je zřejmé, že pro vzorce trigonometrických funkcí se používají goniometrické funkce. Tyto vzorce se učí studenti z paměti již na základní škole v matematice. Jsou důležité pro výpočty různých úloh. Zejména jde o úlohy související s výpočty v trojúhelníku, kde už hovoříme o trigonometrii [8,9].

Hledaná trigonometrické funkce	Trigonometrická identita	Trigonometrická identita
$\sin(x)$	$\pm \sqrt{1 - \cos^2(x)}$	$\pm \frac{\tan(x)}{\sqrt{1 + \tan^2(x)}}$
$\cos(x)$	$\pm \sqrt{1 - \sin^2(x)}$	$\pm \frac{1}{\sqrt{1 + \tan^2(x)}}$
$\tan(x)$	$\pm \frac{\sin(x)}{\sqrt{1 - \sin^2(x)}}$	$\pm \frac{\sqrt{1 - \cos^2(x)}}{\cos(x)}$

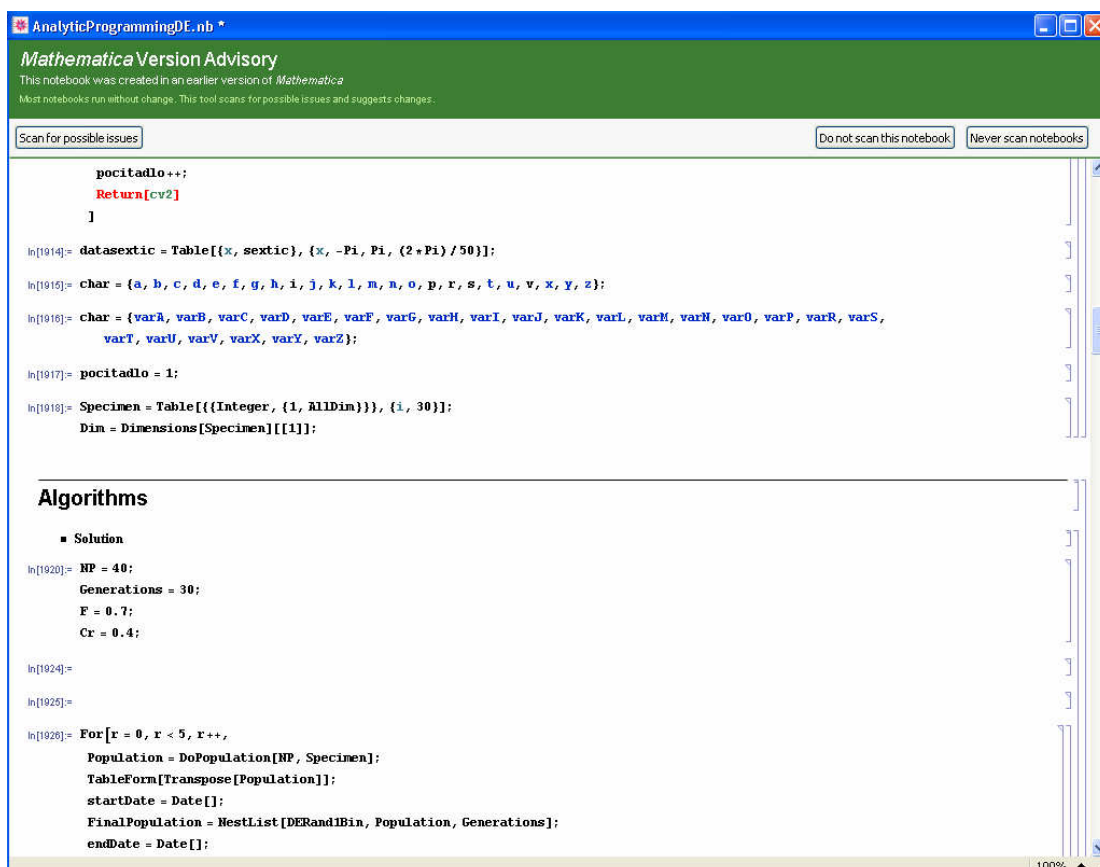
Tabulka 10 : Příklad trigonometrických identit

II. PRAKTICKÁ ČÁST

5 POUŽITÝ SOFTWARE

5.1 Mathematica

Mathematica je počítačový software pro matematické, inženýrské a vědecké výpočty. Je vyvíjen společností Wolfram Research. Tento software bude použit pro výpočet všech zadaných trigonometrických identit. V Mathematice je též naprogramováno analytické programování, evoluční algoritmus SOMA a evoluční algoritmus diferenciální evoluce. Tyto evoluční algoritmy jsou použity k objevování trigonometrických identit. Software Mathematica se skládá ze dvou částí, z jádra a frontendu. Jádro zpracovává zapsaný kód Mathematicy (výrazy) a vrací výsledky. Frontend zajišťuje Mathematice grafické rozhraní pro psaní kódu a prezentaci výsledku [11].



Obrázek 6 : Ukázka softwaru Mathematica

5.2 Analytické programování

Analytické programování je naprogramováno v softwaru Mathematica. Samotné analytické programování se skládá z evolučního algoritmu, v našem případě SOMA nebo diferenciální evoluce, dále z funkcí samotného analytického programování a z funkcí pro zadávání parametrů problému.

Jak již bylo řečeno a vysvětleno, analytické programování musí mít ke své činnosti externí evoluční algoritmus. V této práci jsou použity dva evoluční algoritmy, SOMA a DE. Implementace algoritmů není předmětem praktické části této diplomové práce. Implementace je uložena v notebooku Mathematicy. Předmětem je nastavení evolučních algoritmů, úprava účelové funkce, nastavení funkcí analytického programování a zpracování výsledku pro výukové a prezentační účely.

Zadávání hledané funkce se děje v sekci notebooku Problems proměnné hledana. Tato proměnná slouží k zadávání jednotlivých hledaných funkcí, k níž se hledají trigonometrické identity.

Problems

```
In[44]:= CriticalParameterSet[50, 10, 1000 000];  
  
In[45]:= TimeLimit  
  
Out[45]= 10  
  
In[46]:= hledana = Sqrt[1 - Power[Cos[x], 2]]  
  
Out[46]=  $\sqrt{1 - \cos^2[x]}$ 
```

Obrázek 7 : Zadání hledané funkce

Dalším krokem je nastavení analytického programování a jeho funkcí, které může použít při sestavování výrazů. Taktéž v sekci Problems.

```

In[48]:= BasicArithmetic = {
    {Plus, 2, {{}}},
    {Subtract, 2, {{}}},
    {Minus, 1, {{}}},
    {Times, 2, {{}}},
    {Divide, 2, {{}}},
    {Sqrt, 1, {{}}},
    {Tan, 1, {{}}},
    {Cos, 1, {{}}}
};

In[49]:= ElementaryFunctions = {
    {Power, 2, {{}}}
};

```

Obrázek 8 : Zadání funkcí použitých analytickým programováním

Jedním z nejdůležitějších kroků je nastavit účelovou funkci. Účelovou funkci pro tento typ úloh popisují v další kapitole. V podstatě se jedná o součet absolutních hodnot rozdílů hledané a nalezené funkce. Účelová funkce se taktéž nastavuje v sekci Problems.

```

CostValue[fce_] := Module[{RetFce, cv1, cv2},
  If[fce[[2]] > 0,
  {
    tmp = fce;
    CostFce[K_] := Evaluate[tmp[[1]]];
    var = Take[char, fce[[2]]];
    RetFce = CostFce[var] /. FindFit[datahledana, CostFce[var], var, {x}];
  },
  RetFce = fce[[1]];
  temp = Abs[Re[Table[hledana - N[Re[RetFce], {x, -Pi, Pi, (2 * Pi) / 50}]]] /.
    {Indeterminate -> 10 000., ComplexInfinity -> 10 000., Infinity -> 10 000.}];
  cv1 = If[ones.temp > 1000 000, 1000 000., ones.temp];
  cv2 = If[Head[cv1] == Real, 1.0 cv1, 1000 000., 1000 000.];
  vyslfce = RetFce;
  pocitadlo++;
  Return[cv2]
]

```

Obrázek 9 : Zadání účelové funkce

Následuje neméně důležitá činnost, a to nastavení parametrů použitého evolučního algoritmu. V tomto případě jako příklad uvedu parametry algoritmu SOMA. Tyto parametry se nastavují v sekci Algorithms.

Algorithms

■ Solution

```
In[71]:= PathLength = 3.;
Step = 0.11;
PRT = .1;
PopSize = 20;
Migrations = 10;
AcceptedError = -.01;
```

Obrázek 10 : Nastavení algoritmu SOMA

Po nastavení všech těchto náležitostí můžeme začít se spuštěním výpočtu a sběru výsledků. Po spuštění výpočtu se provede inicializace evolučního algoritmu. Dojde k vytvoření prvotní populace a spuštění evolučního procesu. Po dokončení výpočtu Mathematica vypíše výsledky. Průběh výpočtů a jejich výsledky jsou řešeny v další kapitole.

```
LinearAlgebra`BLAS`TRSV::oflow : Machine overflow encountered during computations.
General::stop : Further output of LinearAlgebra`BLAS`TRSV::oflow will be suppressed during this calculation. »
Divide::infty : Infinite expression  $\frac{1}{0}$  encountered. »

Cas : {0, 0, 0, 0, 6, -1.0781250}
Best individual is on position 2 with cost value +0.00000<E>0 and parameters {7.95, 8.97, 2, 6, 7,
6, 2, 9.6038, 8, 4, 6, 9, 1, 10, 6, 10, 3, 1.03, 6.94, 7, 8.96, 9.01, 7, 7, 1.08, 2.9904, 2., 5, 5, 8.96}
Cos[x]
Cos[x]
Cos[x]
Cos[x]
Cos[x]
Cos[x]
Cos[x]
{Cos[(0.0495468 - x) Cos[x - 1.03331 Tan[1.00611 x (0.520067 + x) Cos[x] Cos[Tan[x]]]],
-7.77156 × 10-16 + 4.44089 × 10-16 x - Cos[3.14159 + x], Cos[x], Cos[x], Cos[x], Cos[x], Cos[x], Cos[x], Cos[x], Cos[x]]
{23.3474, 5.14033 × 10-14, 0., 0., 0., 0., 0., 0., 0., 0., 0.}

-----
Divide::indet : Indeterminate expression  $\frac{0}{0}$  encountered. »

Cas : {0, 0, 0, 0, 5, -0.3750000}
Best individual is on position 4 with cost value +0.00000<E>0 and parameters {8, 8.92, 5.3466, 10,
4, 9, 9, 6, 9, 1.06, 9, 10, 7.14, 5, 5, 3.06, 10, 5.9484, 4, 6, 10, 1, 4.817, 2, 4, 5, 4, 1., 1.01, 1}
Cos[x]
Cos[x]
Cos[x]
Cos[x]
Cos[x]
Cos[x]
Cos[x]
{Sqrt[Cos[x]], Sqrt[Cos[x]], -Cos[3.14159 - x], Cos[x], Cos[x], Cos[x], Cos[x], Cos[x], Cos[x], Cos[x]]
{20.1122, 20.1122, 7.21645 × 10-15, 0., 0., 0., 0., 0., 0., 0., 0., 0.}

-----
```

Obrázek 11 : Příklad hotového výpočtu hledané identity $\cos(x)$

6 ZPRACOVÁNÍ ÚLOH

Všechny úlohy jsou zpracovány v softwaru Mathematica, kde bylo naprogramováno také analytické programování. U všech úloh jsou zadávané parametry a výsledky, kterých bylo dosaženo. Každá identita byla spočtena 10x, aby byla zajištěna objektivita nálezu dané identity, a možnost nalézt více identit daného vzorce s tím, že nejlepší je nalézt alternativní identitu ke známým matematickým vzorcům.

6.1 Postup

Nejprve je nezbytné určit hledanou trigonometrickou identitu. Těmito identitami jsou různé trigonometrické funkce např. $\cos\left(\frac{\pi}{2} - x\right)$ nebo např. $\sqrt{1 - \cos^2(x)}$. Dalším úkolem je nastavit množinu funkcí, pomocí nichž budeme identity hledat. Těmito funkcemi jsou např. $\sin(\)$, $\cos(\)$, ale i $\sqrt{\ }$ a základní matematické operace sčítání, odčítání a tak dále. Určíme účelovou funkci, která je popsána v následující kapitole. Následně bude úkolem nastavit parametry použitého evolučního algoritmu. Pro každý algoritmus je nutné nastavit rozdílné parametry, viz teoretická část o evolučních algoritmech. Pro praktickou část byly použity dva evoluční algoritmy, a to SOMA a diferenciální evoluce. Následuje spuštění evolučního procesu, a po určitém čase, dáno nastavením algoritmu, Mathematica vypíše výsledky evoluce. Jestliže výpočty proběhnou v pořádku, výsledky jsou zapsány do tabulek viz níže.

6.1.1 Účelová funkce

Účelová funkce je jednou z nejdůležitějších součástí analytického programování, které se musíme věnovat. Kvalita účelové funkce určuje také kvalitu výsledného řešení. U úloh typu objevování trigonometrických identit bude jako účelová funkce sloužit absolutní hodnota vzdálenosti mezi bodem funkce hledané a bodem funkce nalezené. Tyto hodnoty nadále sečteme, a tím získáme kvalitu našeho řešení, viz obrázek. Matematický popis účelové funkce (7)

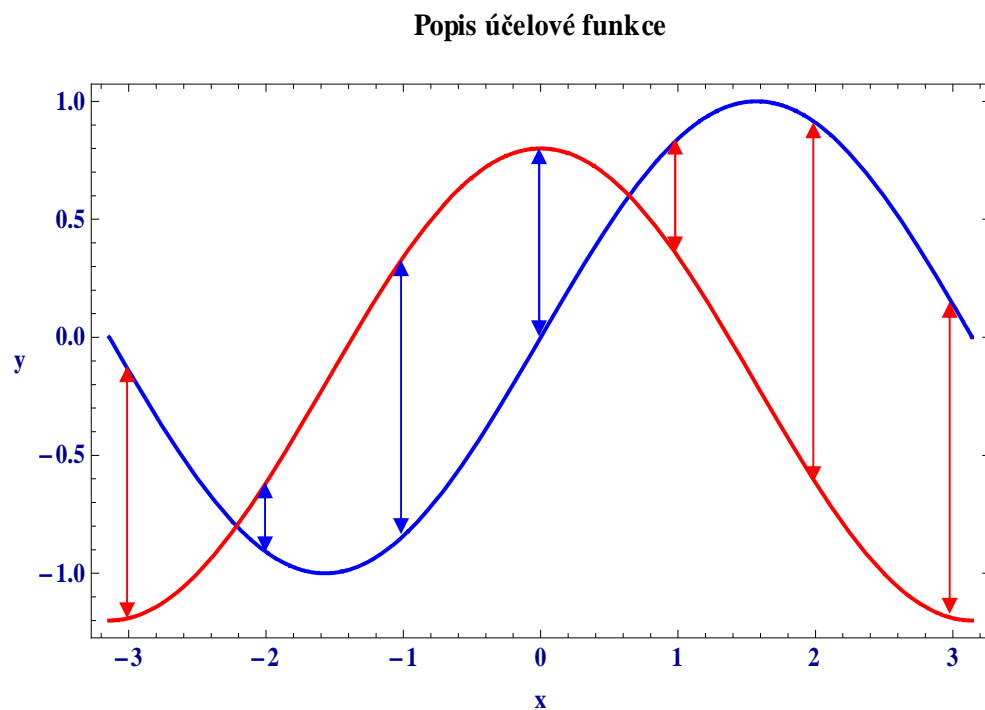
$$F_{\text{cost}}(x) = \sum_{i=\text{dolní hranice}}^{\text{horní hranice}} |f(x_i) - g(x_i)| \quad (7)$$

,kde

$F_{\text{cost}}(x)$ - účelová funkce

$f(x)$ - hledaná funkce

$g(x)$ - nalezená funkce



Obrázek 12 : Popis účelové funkce

6.1.2 Výsledky

Výsledkem evoluce je nalezená funkce, která by měla být identická s hledanou funkcí. Tuto míru identity udává účelová funkce, tzn. jak vzdálená je funkce hledaná od funkce nalezené. V tomto případě by měly být účelové funkce co nejblíže nebo rovny 0. Jestliže je účelová funkce rovna nule, znamená to, že na daném intervalu funkce nalezená kopíruje funkci hledanou ve všech bodech. Součástí výsledku je i orientační čas výpočtu evoluce. Tento čas je různý v závislosti na řešeném problému a nastavení evolučního algoritmu. Součástí řešení jsou i grafy zobrazující porovnání funkce hledané a funkce nalezené. K dispozici je také graf vývoje nejlepších jedinců v každé generaci použitého evolučního algoritmu. Každá z trigonometrických identit byla zjišťována deseti pokusy.

6.2 Úlohy SOMA

Zde jsou vypočteny úlohy hledání identit s pomocí analytického programování a použitého algoritmu SOMA.

6.2.1 Úloha číslo 1

Zadání : Nalezněte identitu funkce $\sin(x)$ pomocí funkcí základních matematických operací a trigonometrických funkcí

Nastavení parametrů

Parametr	Hodnota
Zadaný vzorec	$\sin(x)$
Hledáno na intervalu	$\langle -\pi, \pi \rangle$
Maximální počet členů vzorce	30
PathLength	3
Step	0.11
PRT	1
PopSize	10
Migrations	10
AcceptedError	-0.1

Tabulka 11 : Nastavení parametrů úlohy 1 algoritmu SOMA

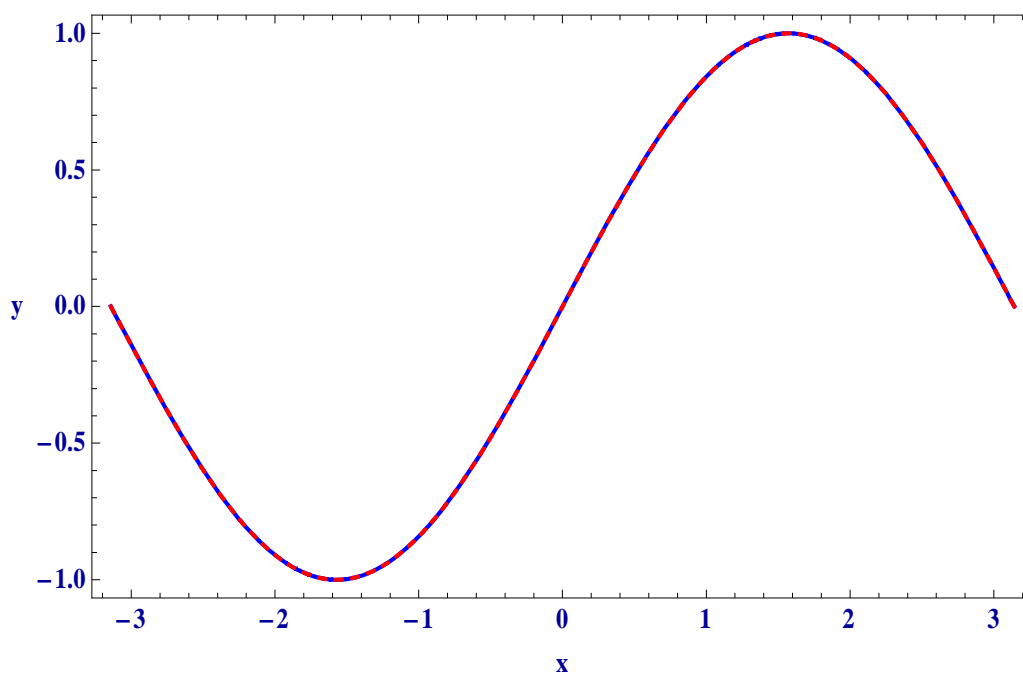
Výsledky hledání

Výsledky				
Pokus č.	Nalezená identita	Čas evoluce [mm:ss]	Počet ohodnocení účelové funkce	Hodnota účelové funkce
1	$\sin(x)$	2:20	2455	0

2	$\sin(x)$	4:10	2455	0
3	$\sin(x)$	3:15	2455	$1,23 \cdot 10^{-15}$
4	$\sin(x)$	2:05	2455	0
5	$\cos(0,0018795(835,74246 - x) - 0,9981 \cdot x)$	3:20	2455	$2,65 \cdot 10^{-15}$
6	$\sin(x)$	3:30	2455	0
7	$\sin(x)$	2:25	2455	0
8	$\sin(x)$	3:05	2455	0
9	$\sin(x)$	4:15	2455	$2,13 \cdot 10^{-15}$
10	$\cos(1,5707963267 - x)$	4:05	2455	$4,87 \cdot 10^{-11}$

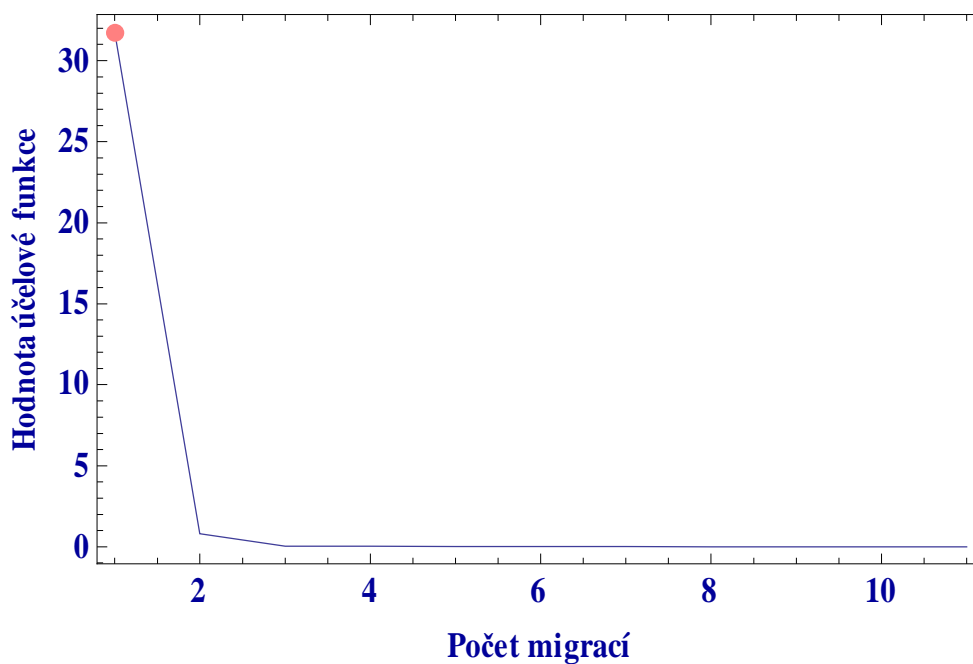
Tabulka 12 : Výsledky úlohy 1 algoritmu SOMA

Porovnání hledaného trigonometrického vzorce s nalezenou identitou



Obrázek 13 : Graf porovnání úlohy 1 algoritmu SOMA

Historie nejlepších jedinců



Obrázek 14 : Graf historie jedinců úlohy 1 algoritmu SOMA

6.2.2 Úloha číslo 2

Zadání : Nalezněte identitu funkce $1 - \sin^2(x)$ pomocí funkcí základních matematických operací a trigonometrických funkcí

Nastavení parametrů

Parametr	Hodnota
Zadaný vzorec	$1 - \sin^2(x)$
Hledáno na intervalu	$\langle -\pi, \pi \rangle$
Maximální počet členů vzorce	30
PathLength	3
Step	0.11
PRT	1

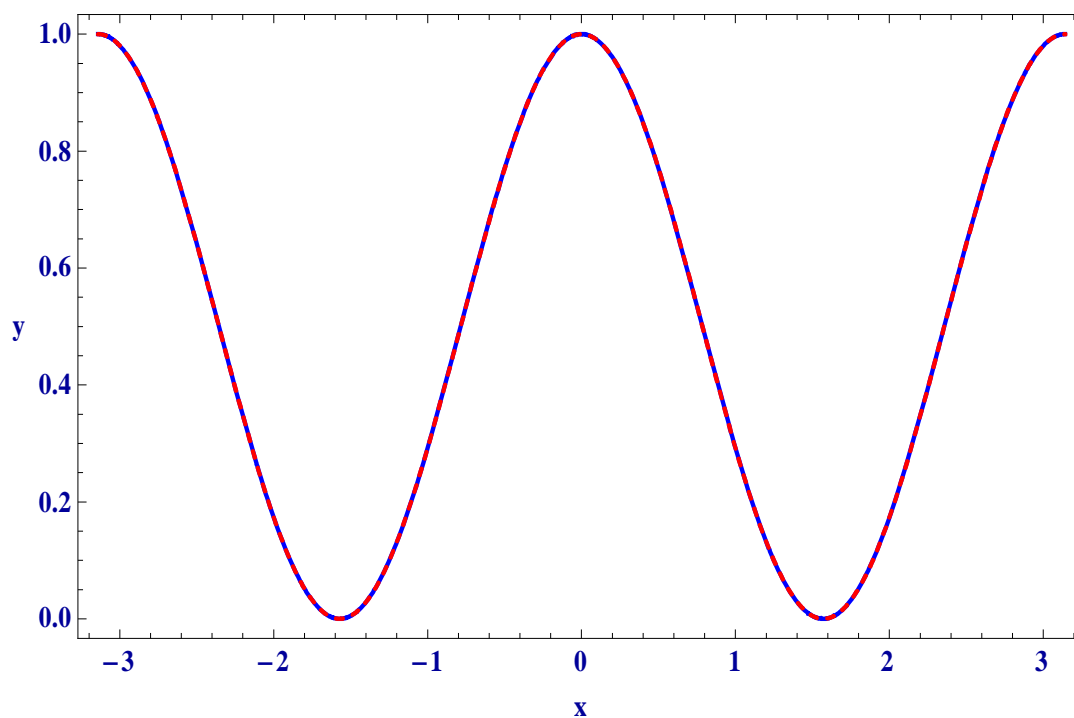
PopSize	20
Migrations	10
AcceptedError	-0.1

Tabulka 13 : Nastavení parametrů úlohy 2 algoritmu SOMA

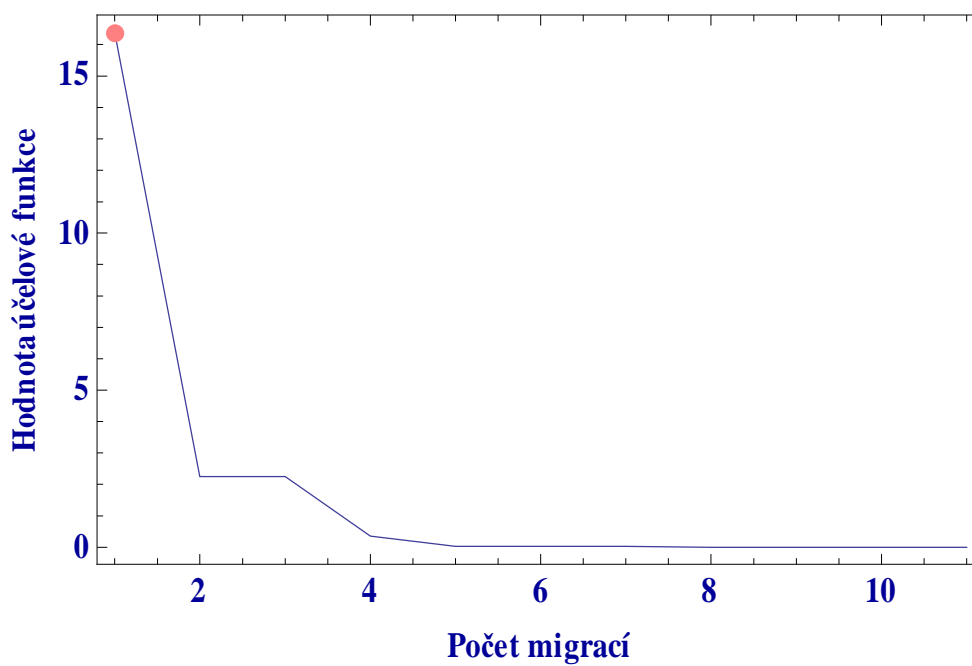
Výsledky hledání

Výsledky				
Pokus č.	Nalezená identita	Čas evoluce [mm:ss]	Počet ohodnocení účelové funkce	Hodnota účelové funkce
1	$\cos^2(x)$	7:00	5182	$2,48 \cdot 10^{-15}$
2	$\cos^2(x)$	5:20	5182	0
3	$\cos^2(x)$	9:30	5182	0
4	$0,5 + 0,5 \cdot \cos(2 \cdot x)$	5:40	5182	$6,65 \cdot 10^{-15}$
5	$\cos^2(x)$	3:30	5182	0
6	$0,708073 + \cos(x) \cdot \sin(0,570796 + x)$	6:00	5182	$7,23 \cdot 10^{-15}$
7	$0,5 + 0,5 \cdot \cos(2 \cdot x)$	6:20	5182	$4,32 \cdot 10^{-15}$
8	$\cos^2(x)$	5:10	5182	0
9	$\cos^2(x)$	4:05	5182	0
10	$\cos^2(x)$	5:05	5182	$2,12 \cdot 10^{-15}$

Tabulka 14 : Výsledky úlohy 2 algoritmu SOMA

Porovnání hledaného trigonometrického vzorce s nalezenou identitou

Obrázek 15 : Graf porovnání úlohy 2 algoritmu SOMA

Historie nejlepších jedinců

Obrázek 16 : Graf historie jedinců úlohy 2 algoritmu SOMA

6.2.3 Úloha číslo 3

Zadání : Nalezněte identitu funkce $\cos\left(\frac{\pi}{2} - x\right)$ pomocí funkcí základních matematických operací a trigonometrických funkcí

Nastavení parametrů

Parametr	Hodnota
Zadaný vzorec	$\cos\left(\frac{\pi}{2} - x\right)$
Hledáno na intervalu	$\langle -\pi, \pi \rangle$
Maximální počet členů vzorce	30
PathLength	3
Step	0.11
PRT	1
PopSize	20
Migrations	10
AcceptedError	-0.1

Tabulka 15 : Nastavení parametrů úlohy 3 algoritmu SOMA

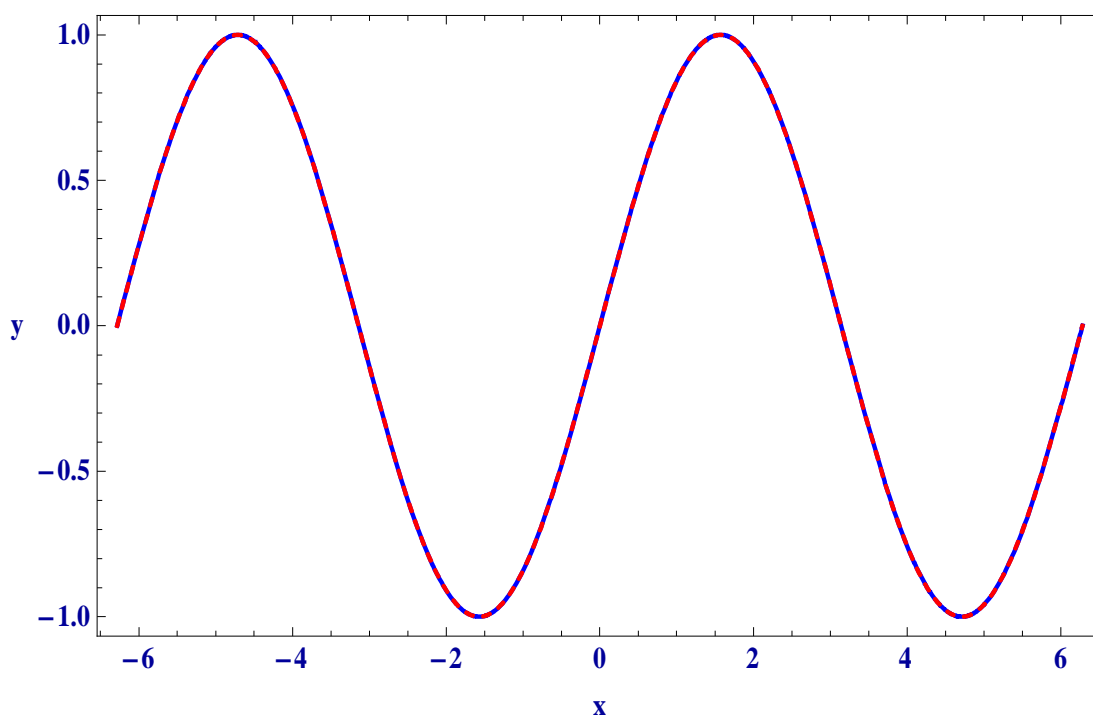
Výsledky hledání

Výsledky				
Pokus č.	Nalezená identita	Čas evoluce [mm:ss]	Počet ohodnocení účelové funkce	Hodnota účelové funkce
1	$0 - (0 - \cos(x) \cdot \tan(x))$	6:15	5182	$2,88 \cdot 10^{-15}$
2	$\sin(x)$	6:10	5182	0
3	$\cos(1,5707963267 - x)$	5:05	5182	$5,69 \cdot 10^{-15}$

4	$\sin(x)$	10:30	5182	0
5	$\sin(x)$	7:10	5182	$4,26 \cdot 10^{-15}$
6	$\sin(x)$	7:30	5182	0
7	$\cos(1,5707963267 - x)$	7:50	5182	$4,32 \cdot 10^{-15}$
8	$\sin(x)$	9:50	5182	$4,36 \cdot 10^{-15}$
9	$\sin(x)$	8:40	5182	0
10	$\sin(x)$	9:15	5182	0

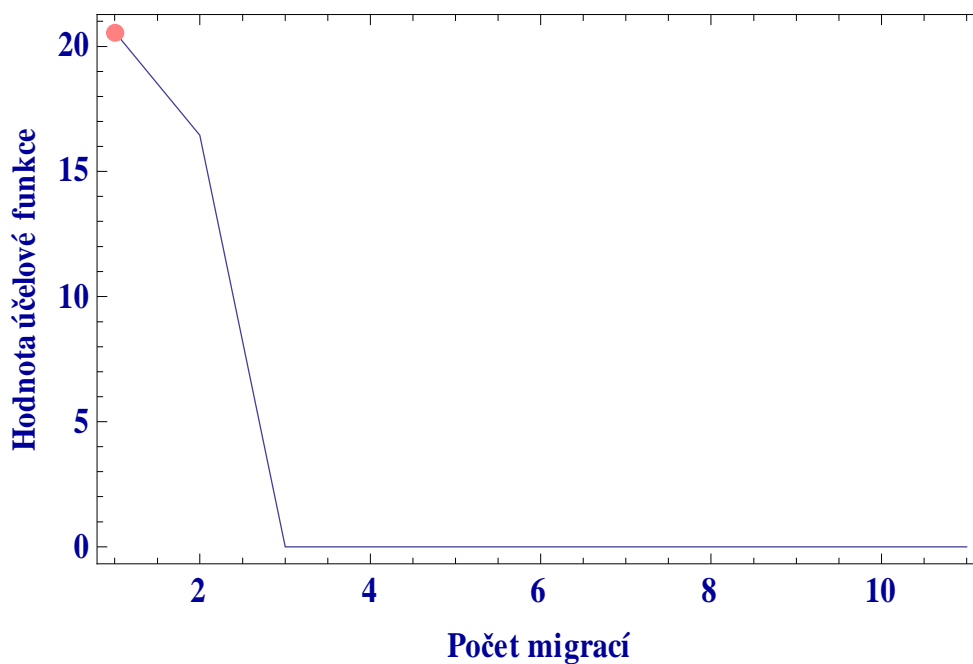
Tabulka 16 : Výsledky úlohy 3 algoritmu SOMA

Porovnání hledaného trigonometrického vzorce s nalezenou identitou



Obrázek 17 : Graf porovnání úlohy 3 algoritmu SOMA

Historie nejlepších jedinců



Obrázek 18 : Graf historie jedinců úlohy 3 algoritmu SOMA

6.2.4 Úloha číslo 4

Zadání : Nalezněte identitu funkce $\cos^2(x)$ pomocí funkcí základních matematických operací a trigonometrických funkcí

Nastavení parametrů

Parametr	Hodnota
Zadaný vzorec	$\cos^2(x)$
Hledáno na intervalu	$\langle -\pi, \pi \rangle$
Maximální počet členů vzorce	30
PathLength	3
Step	0.11
PRT	1
PopSize	20

Migrations	10
AcceptedError	-0.1

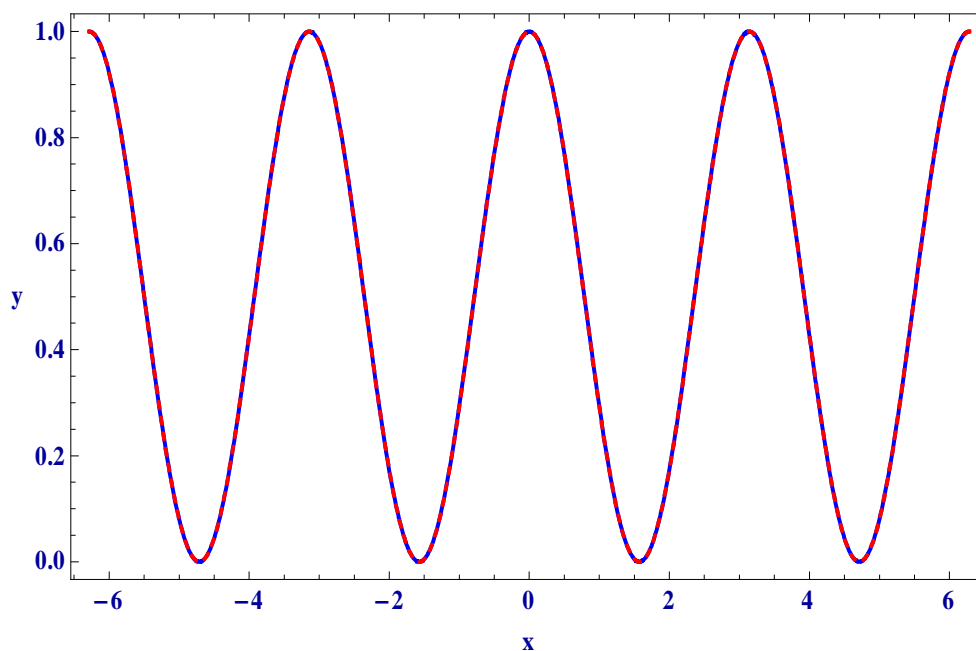
Tabulka 17 : Nastavení parametrů úlohy 4 algoritmu SOMA

Výsledky hledání

Výsledky				
Pokus č.	Nalezená identita	Čas evoluce [mm:ss]	Počet ohodnocení účelové funkce	Hodnota účelové funkce
1	$0,5 + 0,5 \cdot \cos(2 \cdot x)$	6:00	5182	$1,09 \cdot 10^{-15}$
2	$0,5 + 0,5 \cdot \cos(2 \cdot x)$	8:10	5182	$1,96 \cdot 10^{-15}$
3	$\cos^2(x)$	7:15	5182	0
4	$0,5 + 0,5 \cdot \cos(2 \cdot x)$	5:35	5182	$4,35 \cdot 10^{-15}$
5	$\cos^2(x)$	6:35	5182	0
6	$\cos^2(x)$	6:05	5182	0
7	$\sqrt{1 - \cos^2(x)}$	5:15	5182	$2,32 \cdot 10^{-15}$
8	$\cos^2(x)$	5:10	5182	0
9	$\cos^2(x)$	5:20	5182	0
10	$0,5 + 0,5 \cdot \cos(2 \cdot x)$	7:15	5182	$4,37 \cdot 10^{-15}$

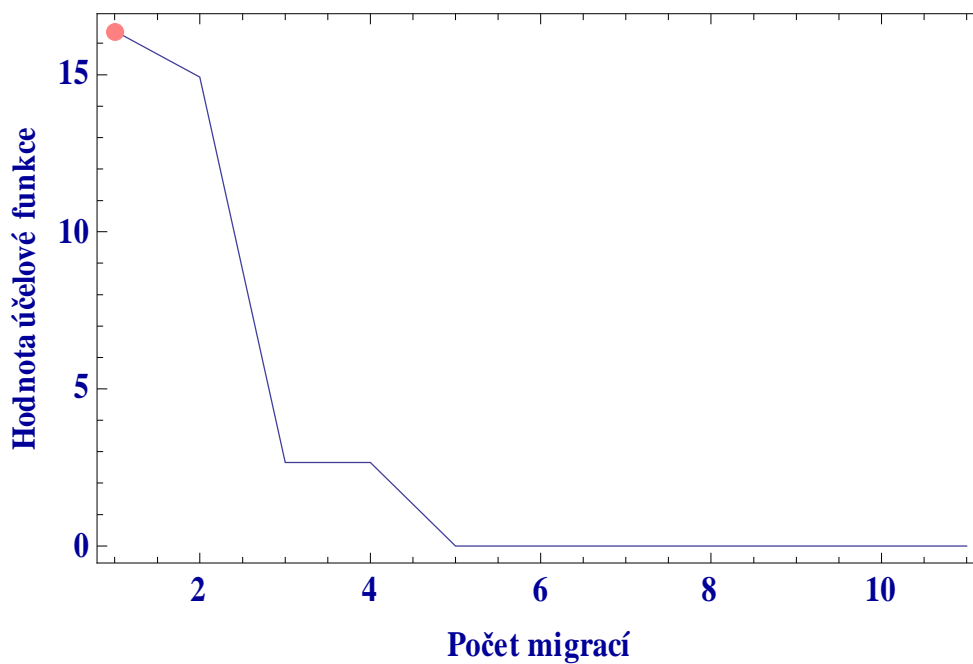
Tabulka 18 : Výsledky úlohy 4 algoritmu SOMA

Porovnání hledaného trigonometrického vzorce s nalezenou identitou



Obrázek 19 : Graf porovnání úlohy 4 algoritmu SOMA

Historie nejlepších jedinců



Obrázek 20 : Graf historie jedinců úlohy 4 algoritmu SOMA

6.2.5 Úloha číslo 5

Zadání : Nalezněte identitu funkce $\sqrt{1 - \cos^2(x)}$ pomocí funkcí základních matematických operací a trigonometrických funkcí.

Nastavení parametrů

Parametr	Hodnota
Zadaný vzorec	$\sqrt{1 - \cos^2(x)}$
Hledáno na intervalu	$\langle -\pi, \pi \rangle$
Maximální počet členů vzorce	30
PathLength	3
Step	0.11
PRT	1
PopSize	20
Migrations	10
AcceptedError	-0.1

Tabulka 19 : Nastavení parametrů úlohy 5 algoritmu SOMA

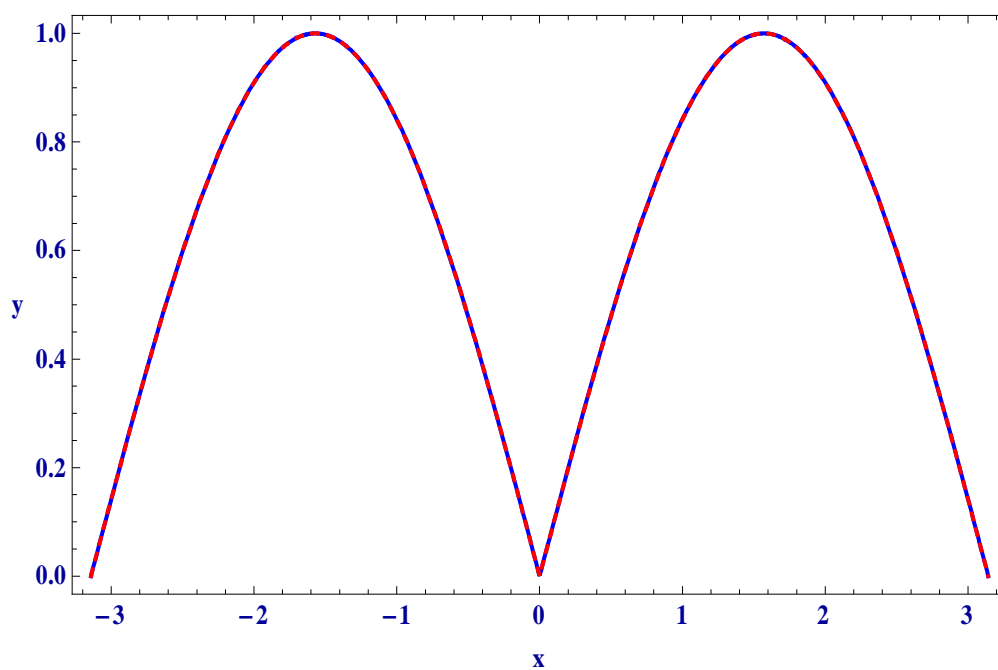
Výsledky hledání

Výsledky				
Pokus č.	Nalezená identita	Čas evoluce [mm:ss]	Počet ohodnocení účelové funkce	Hodnota účelové funkce
1	$\sqrt{\sin^2(x)}$	5:05	5182	$2,77 \cdot 10^{-15}$
2	$\sin(x)$	7:10	5182	0
3	$\sin(x)$	6:15	5182	$1,72 \cdot 10^{-15}$

4	$\text{Cos}(1,5707963267 - x)$	7:35	5182	$2,62 \cdot 10^{-15}$
5	$\text{Sin}(x)$	6:30	5182	0
6	$\sqrt{\text{Sin}^2(x)}$	6:25	5182	0
7	$\text{Sin}(x)$	5:10	5182	0
8	$\text{Sin}(x)$	6:15	5182	$1,33 \cdot 10^{-15}$
9	$\text{Cos}(1,57079633 - \sqrt{x^2})$	5:20	5182	$2,53 \cdot 10^{-15}$
10	$\text{Sin}(x)$	6:15	5182	0

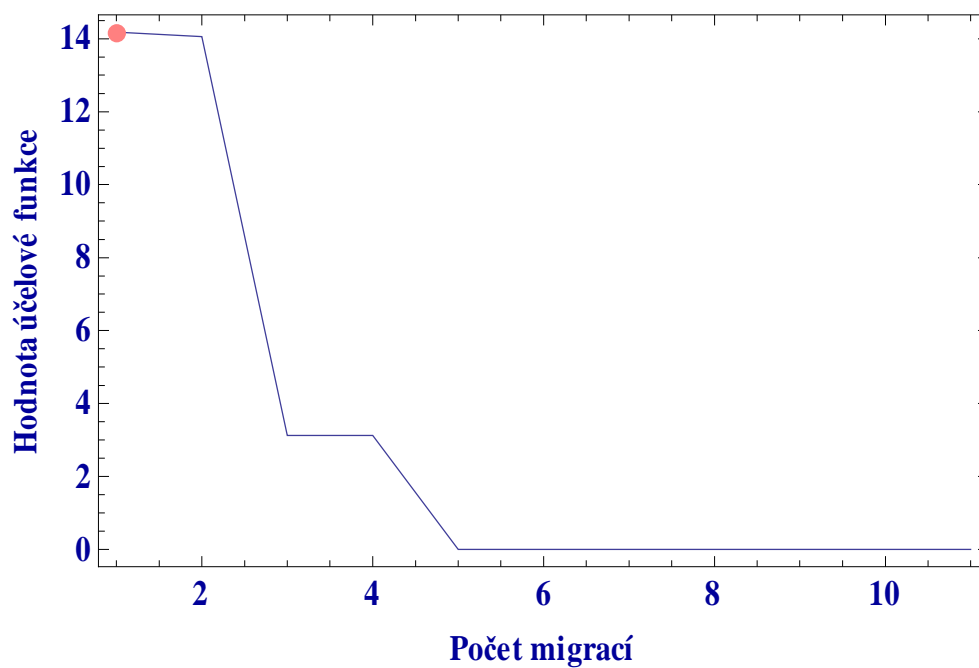
Tabulka 20 : Výsledky úlohy 5 algoritmu SOMA

Porovnání hledaného trigonometrického vzorce s nalezenou identitou



Obrázek 21 : Graf porovnání úlohy 5 algoritmu SOMA

Historie nejlepších jedinců



Obrázek 22 : Graf historie jedinců úlohy 5 algoritmu SOMA

6.3 Úlohy Diferenciální evoluce

Zde jsou vypočteny úlohy hledání identit s pomocí analytického programování a použitého algoritmu diferenciální evoluce.

6.3.1 Úloha číslo 1

Zadání : Nalezněte identitu funkce $\sin(x)$ pomocí funkcí základních matematických operací a trigonometrických funkcí

Nastavení parametrů

Parametr	Hodnota
Zadaný vzorec	$\sin(x)$
Hledáno na intervalu	$\langle -\pi, \pi \rangle$
Maximální počet členů vzorce	30
NP	20
Generation	30
F	0,7
Cr	0,4

Tabulka 21 : Nastavení parametrů úlohy 1 algoritmu Diferenciální evoluce

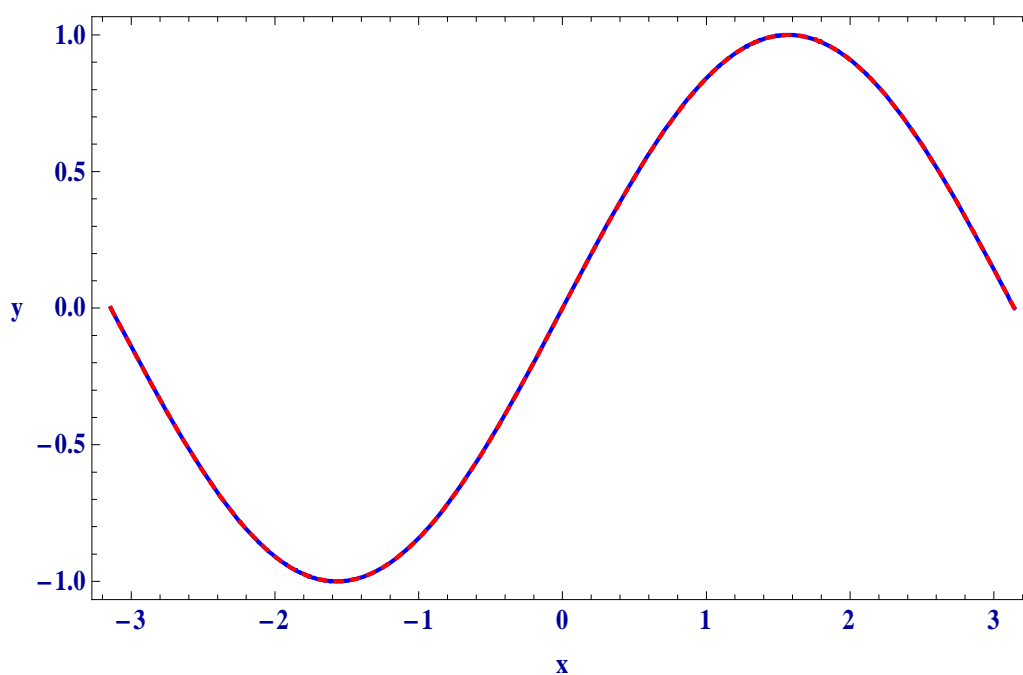
Výsledky hledání

Výsledky				
Pokus č.	Nalezená identita	Čas evoluce [mm:ss]	Počet ohodnocení účelové funkce	Hodnota účelové funkce
1	$\sin(x)$	1:30	600	$2,48 \cdot 10^{-15}$
2	$\cos(1,5707963267 - x)$	0:50	600	$3,12 \cdot 10^{-15}$
3	$\sin(x)$	1:15	600	0

4	$\text{Cos}(1,5707963267 - x)$	1:05	600	$5,4 \cdot 10^{-15}$
5	$\text{Cos}(1,5707963267 - x)$	0:40	600	$1,28 \cdot 10^{-15}$
6	$\text{Sin}(x)$	1:20	600	0
7	$\text{Sin}(x)$	1:25	600	$1,23 \cdot 10^{-15}$
8	$\text{Sin}(x)$	1:05	600	0
9	$\text{Cos}(1,5707963267 - x)$	0:55	600	$2,12 \cdot 10^{-15}$
10	$\text{Cos}(1,5707963267 - x)$	1:00	600	$3,34 \cdot 10^{-15}$

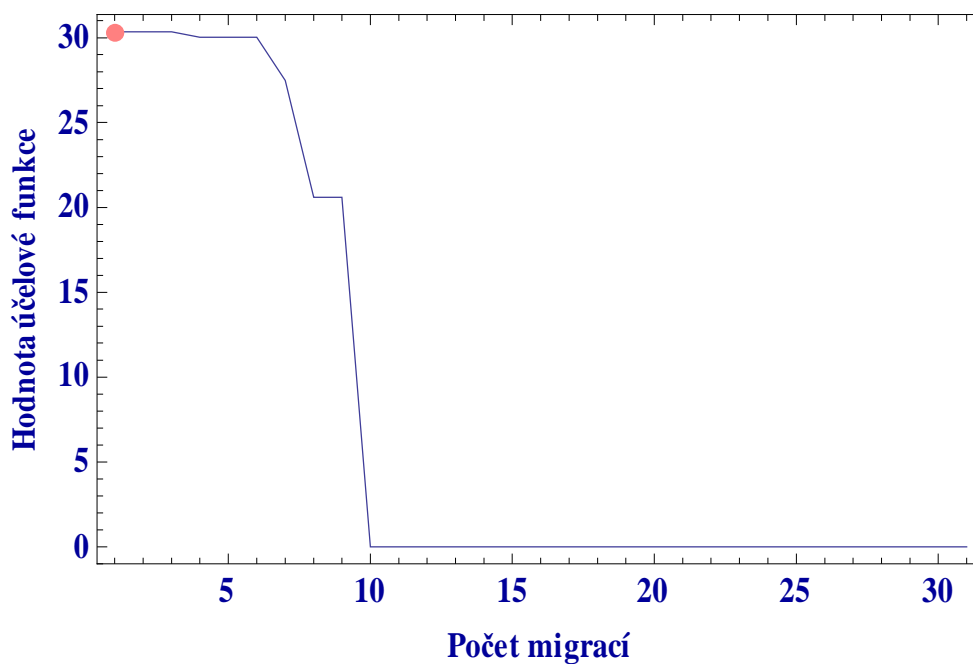
Tabulka 22 : Výsledky úlohy 1 algoritmu Diferenciální evoluce

Porovnání hledaného trigonometrického vzorce s nalezenou identitou



Obrázek 23 : Graf porovnání úlohy 1 algoritmu Diferenciální evoluce

Historie nejlepších jedinců



Obrázek 24 : Graf historie jedinců úlohy 1 algoritmu Diferencialní evoluce

6.3.2 Úloha číslo 2

Zadání : Nalezněte identitu funkce $1 - \sin^2(x)$ pomocí funkcí základních matematických operací a trigonometrických funkcí

Nastavení parametrů

Parametr	Hodnota
Zadaný vzorec	$1 - \sin^2(x)$
Hledáno na intervalu	$\langle -\pi, \pi \rangle$
Maximální počet členů vzorce	30
NP	30
Generation	50
F	0,7

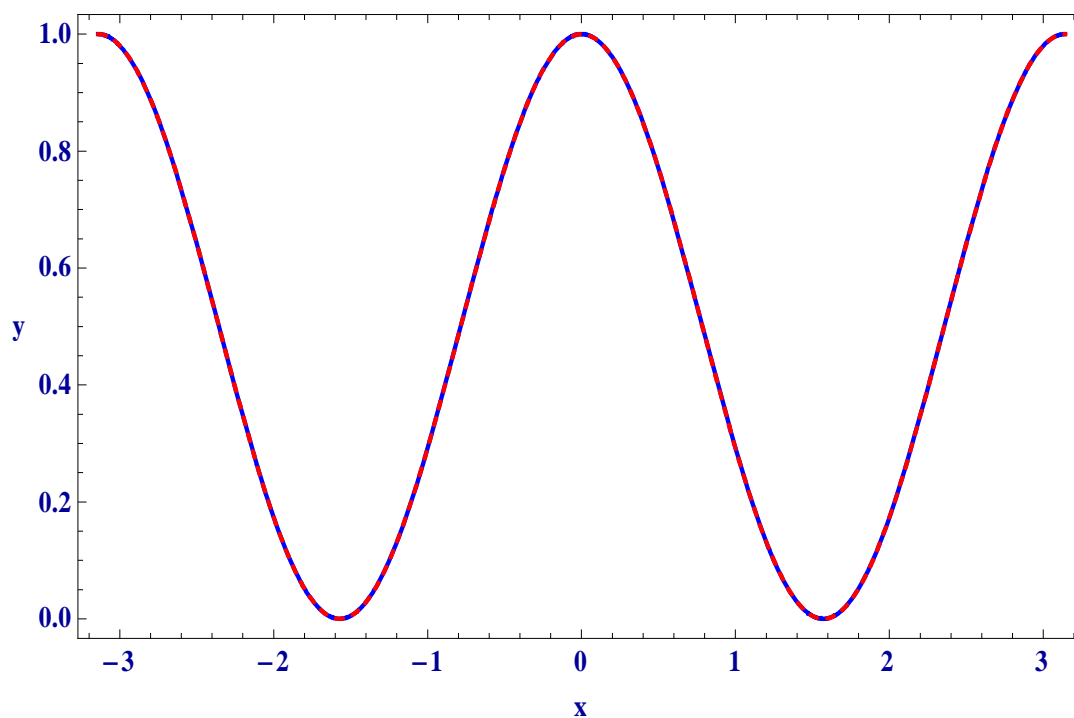
Cr	0,4
----	-----

Tabulka 23 : Nastavení parametrů úlohy 2 algoritmu Diferenciální evoluce

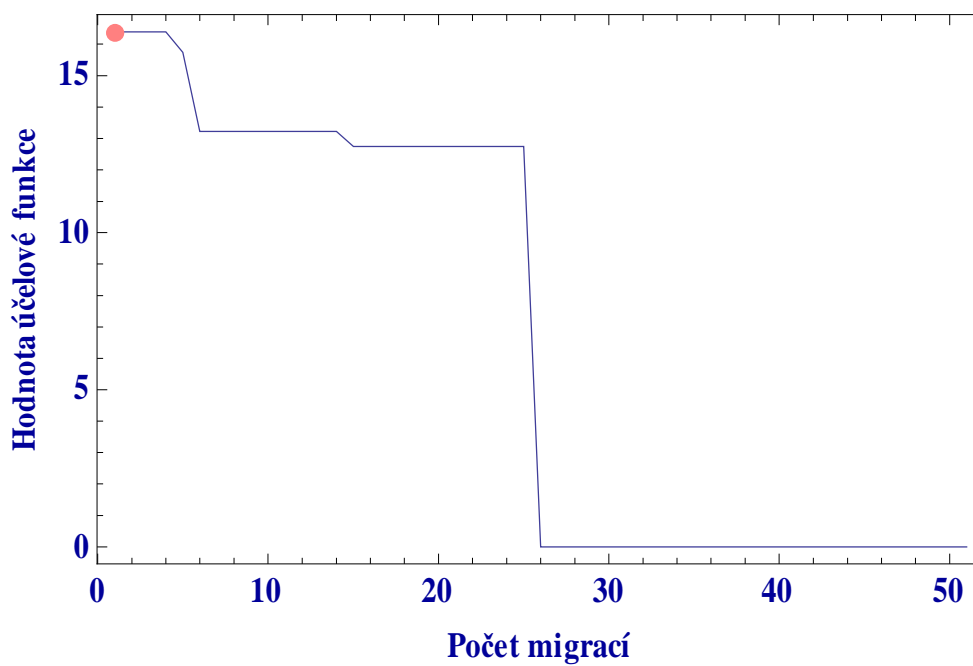
Výsledky hledání

Výsledky				
Pokus č.	Nalezená identita	Čas evoluce [mm:ss]	Počet ohodnocení účelové funkce	Hodnota účelové funkce
1	$\cos(x) \cdot \cos(x)$	2:15	1500	$1,12 \cdot 10^{-15}$
2	$\cos^2(x)$	1:55	1500	$2,56 \cdot 10^{-15}$
3	$1 - \frac{\cos(1,57079632 - 2 \cdot x) \cdot \tan(x)}{2 \cdot x}$	2:00	1500	$3,36 \cdot 10^{-15}$
4	$\cos^2(x)$	2:30	1500	0
5	$\cos^2(x)$	1:45	1500	0
6	$\cos^2(x)$	1:55	1500	$2,12 \cdot 10^{-15}$
7	$\cos^2(x)$	1:50	1500	0
8	$\cos^2(x)$	2:20	1500	0
9	$\cos(x) \cdot \cos(x)$	2:15	1500	$1,89 \cdot 10^{-15}$
10	$\cos^2(x)$	2:00	1500	0

Tabulka 24 : Výsledky úlohy 2 algoritmu Diferenciální evoluce

Porovnání hledaného trigonometrického vzorce s nalezenou identitou

Obrázek 25 : Graf porovnání úlohy 2 algoritmu Diferencialní evoluce

Historie nejlepších jedinců

Obrázek 26 : Graf historie jedinců úlohy 2 algoritmu Diferencialní evoluce

6.3.3 Úloha číslo 3

Zadání : Nalezněte identitu funkce $\cos\left(\frac{\pi}{2} - x\right)$ pomocí funkcí základních matematických operací a trigonometrických funkcí

Nastavení parametrů

Parametr	Hodnota
Zadaný vzorec	$\cos\left(\frac{\pi}{2} - x\right)$
Hledáno na intervalu	$\langle -\pi, \pi \rangle$
Maximální počet členů vzorce	30
NP	30
Generation	50
F	0,7
Cr	0,4

Tabulka 25 : Nastavení parametrů úlohy 3 algoritmu Diferenciální evoluce

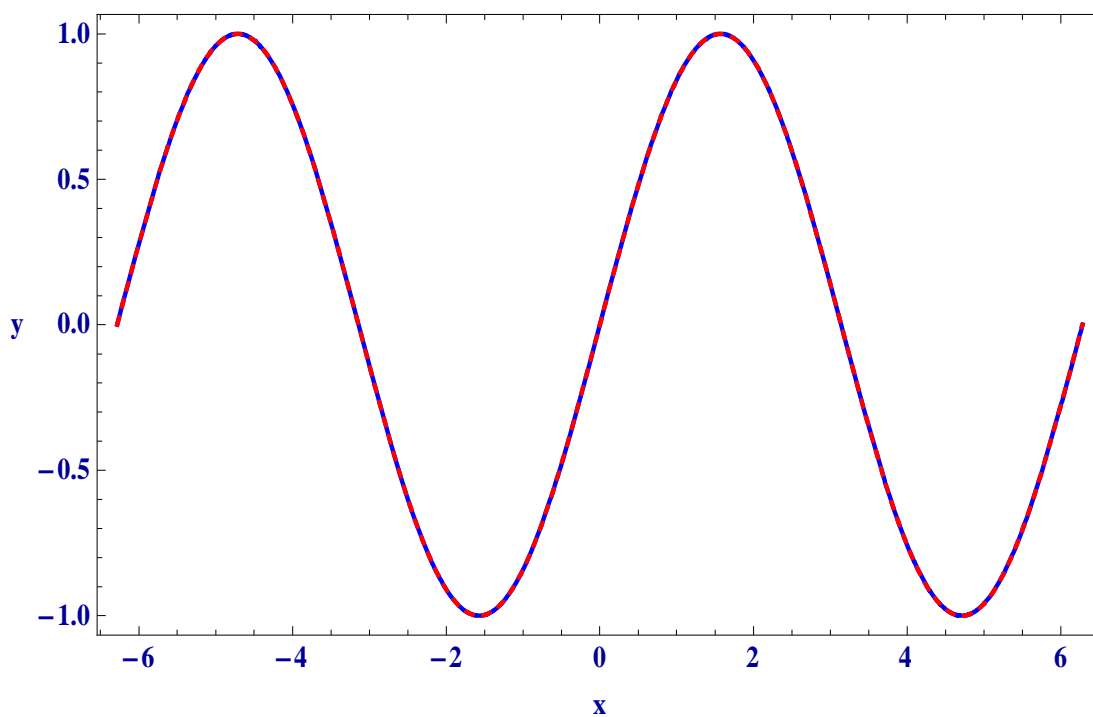
Výsledky hledání

Výsledky				
Pokus č.	Nalezená identita	Čas evoluce [mm:ss]	Počet ohodnocení účelové funkce	Hodnota účelové funkce
1	$\cos(1,5707963267 - x)$	2:50	1500	$2,49 \cdot 10^{-15}$
2	$\sin(x)$	3:15	1500	0
3	$\sin(x)$	2:30	1500	$3,12 \cdot 10^{-15}$
4	$\cos(1,5707963267 - x)$	2:55	1500	$2,42 \cdot 10^{-15}$
5	$\cos(1,5707963267 - x)$	2:40	1500	$5,35 \cdot 10^{-15}$

6	$\sin(x)$	3:30	1500	$1,86 \cdot 10^{-15}$
7	$\sin(x)$	2:45	1500	$2,47 \cdot 10^{-15}$
8	$\sin(x)$	2:55	1500	0
9	$\sin(x)$	2:50	1500	0
10	$\cos(1,5707963267 - x)$	3:10	1500	$5,23 \cdot 10^{-15}$

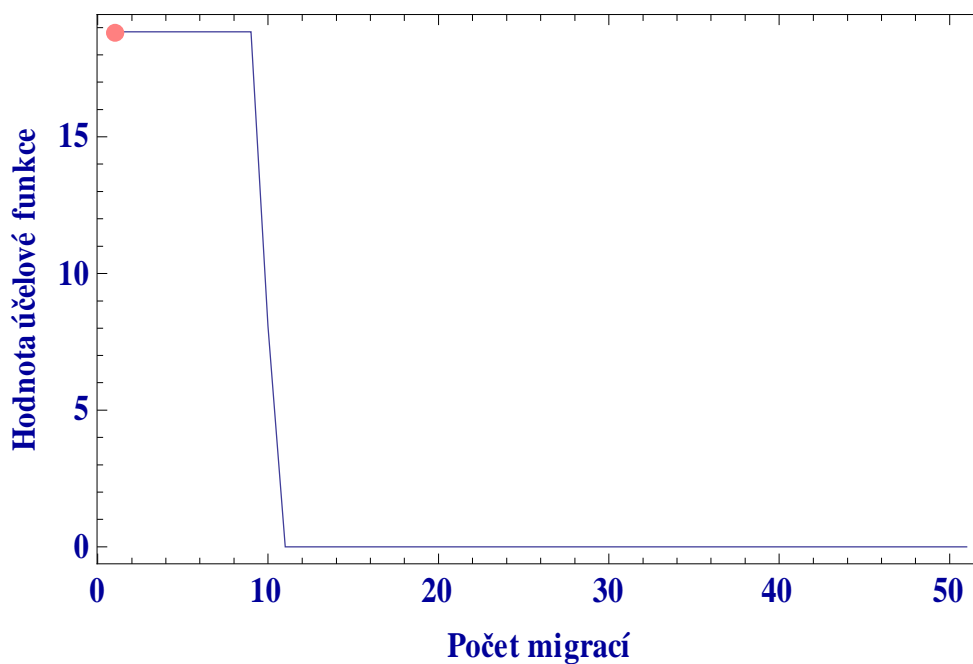
Tabulka 26 : Výsledky úlohy 3 algoritmu Diferenciální evoluce

Porovnání hledaného trigonometrického vzorce s nalezenou identitou



Obrázek 27 : Graf porovnání úlohy 3 algoritmu Diferenciální evoluce

Historie nejlepších jedinců



Obrázek 28 : Graf historie jedinců úlohy 3 algoritmu Diferencialní evoluce

6.3.4 Úloha číslo 4

Zadání : Nalezněte identitu funkce $\cos^2(x)$ pomocí funkcí základních matematických operací a trigonometrických funkcí

Nastavení parametrů

Parametr	Hodnota
Zadaný vzorec	$\cos^2(x)$
Hledáno na intervalu	$\langle -\pi, \pi \rangle$
Maximální počet členů vzorce	30
NP	40
Generation	50
F	0,7
Cr	0,4

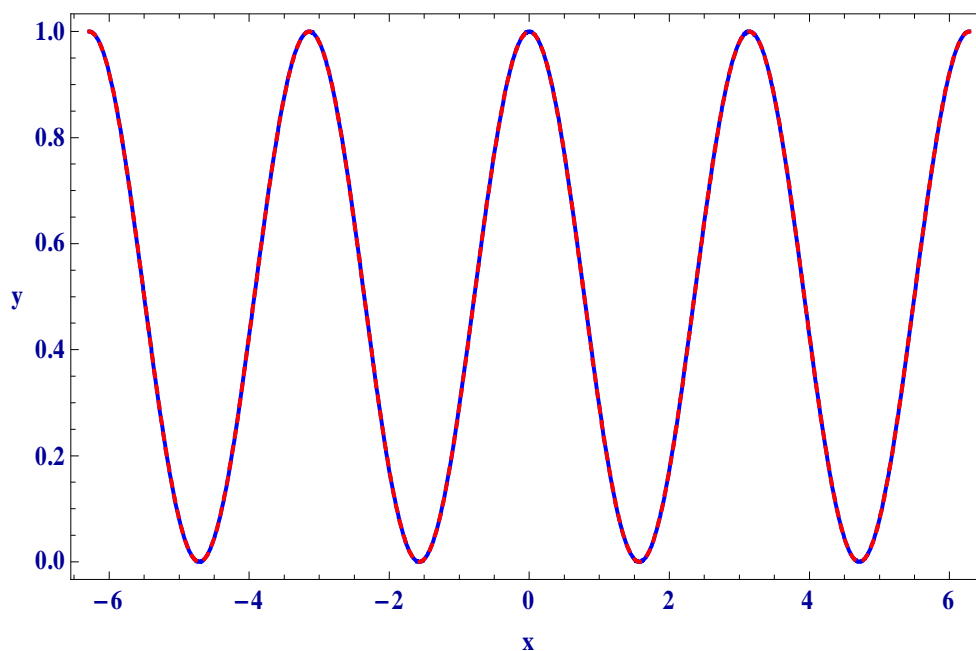
Tabulka 27 : Nastavení parametrů úlohy 4 algoritmu Diferenciální evoluce

Výsledky hledání

Výsledky				
Pokus č.	Nalezená identita	Čas evoluce [mm:ss]	Počet ohodnocení účelové funkce	Hodnota účelové funkce
1	$\cos^2(x)$		2000	0
2	$0,5(1 + 1 \cdot \cos(2 \cdot x))$		2000	$4,74 \cdot 10^{-15}$
3	$\cos^2(x)$		2000	0
4	$\cos(x) \cdot \cos(x)$		2000	$3,23 \cdot 10^{-15}$
5	$\cos^2(x)$		2000	0
6	$\cos^2(x)$		2000	$2,48 \cdot 10^{-15}$
7	$\cos^2(x)$		2000	$1,71 \cdot 10^{-15}$
8	$\cos^2(x)$		2000	0
9	$\cos^2(x)$		2000	$3,42 \cdot 10^{-15}$
10	$\cos^2(x)$		2000	0

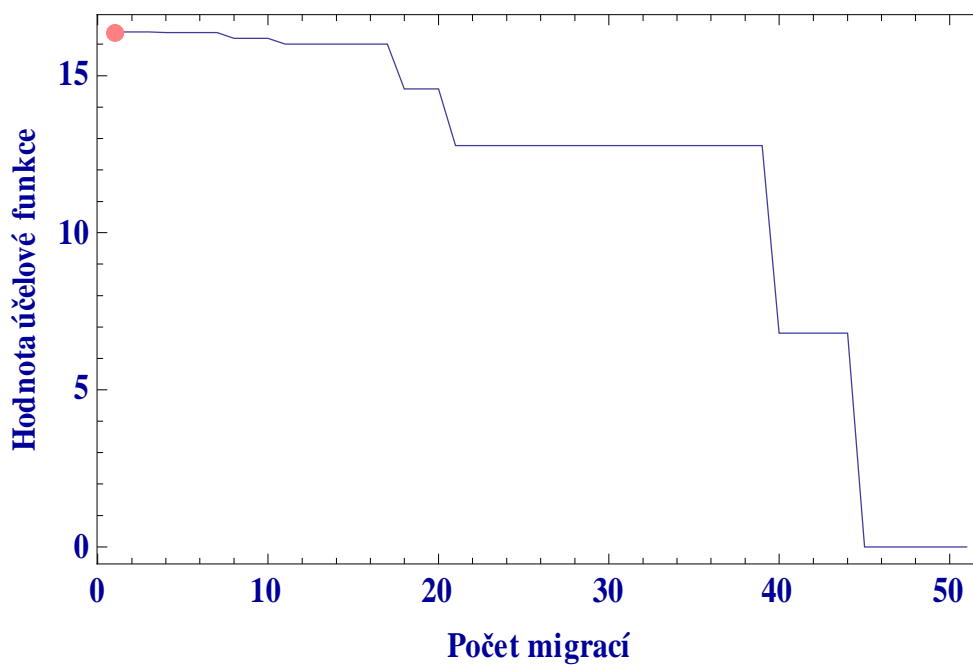
Tabulka 28 : Výsledky úlohy 4 algoritmu Diferenciální evoluce

Porovnání hledaného trigonometrického vzorce s nalezenou identitou



Obrázek 29 : Graf porovnání úlohy 4 algoritmu Diferencialní evoluce

Historie nejlepších jedinců



Obrázek 30 : Graf historie jedinců úlohy 4 algoritmu Diferencialní evoluce

6.3.5 Úloha číslo 5

Zadání : Nalezněte identitu funkce $\sqrt{1 - \cos^2(x)}$ pomocí funkcí základních matematických operací a trigonometrických funkcí.

Nastavení parametrů

Parametr	Hodnota
Zadaný vzorec	$\sqrt{1 - \cos^2(x)}$
Hledáno na intervalu	$\langle -\pi, \pi \rangle$
Maximální počet členů vzorce	30
NP	40
Generation	70
F	0,7
Cr	0,4

Tabulka 29 : Nastavení parametrů úlohy 5 algoritmu Diferenciální evoluce

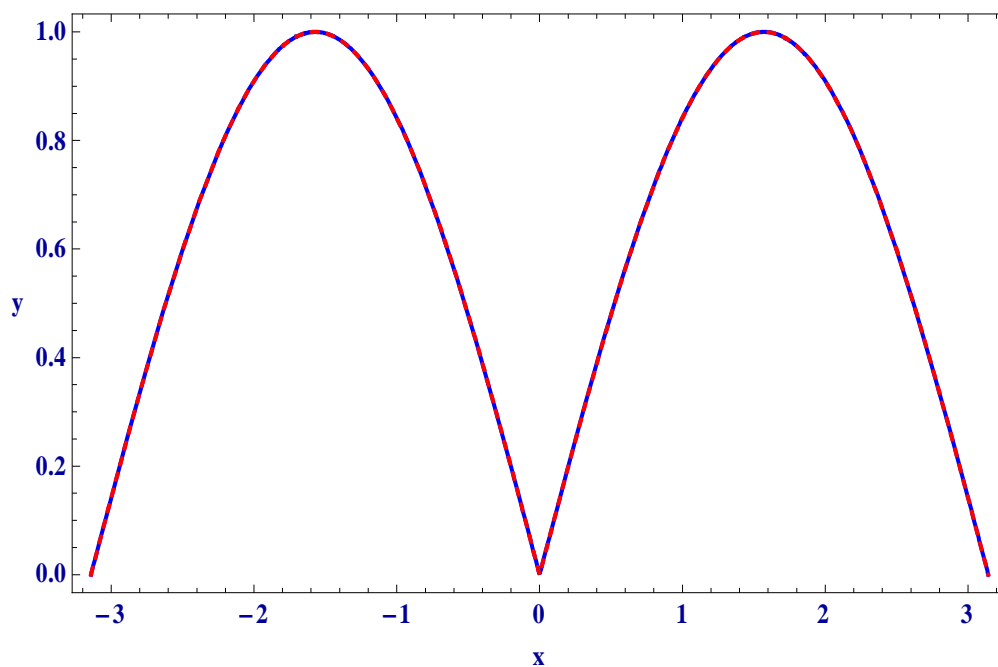
Výsledky hledání

Výsledky				
Pokus č.	Nalezená identita	Čas evoluce [mm:ss]	Počet ohodnocení účelové funkce	Hodnota účelové funkce
1	$\sin(x)$	3:40	2800	$1,35 \cdot 10^{-15}$
2	$\cos(1,5707963267 - x)$	3:35	2800	$3,36 \cdot 10^{-15}$
3	$\sin(x)$	2:55	2800	0
4	$-\cos(1,570796326 + \sqrt{x^2})$	2:40	2800	$4,75 \cdot 10^{-15}$
5	$\sin(x)$	3:15	2800	0

6	$\sin(x)$	3:10	2800	0
7	$\cos(1,5707963267 - x)$	3:20	2800	$4,23 \cdot 10^{-15}$
8	$\sin(x)$	3:25	2800	$3,56 \cdot 10^{-15}$
9	$\sin(x)$	3:00	2800	0
10	$\sin(x)$	2:50	2800	0

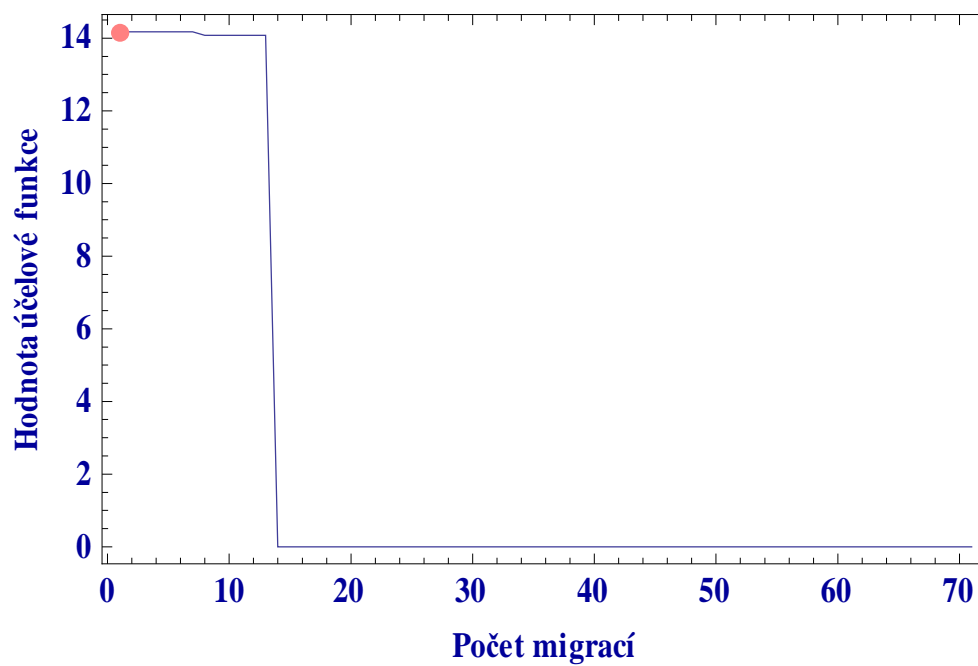
Tabulka 30 : Výsledky úlohy 5 algoritmu Diferenciální evoluce

Porovnání hledaného trigonometrického vzorce s nalezenou identitou



Obrázek 31 : Graf porovnání úlohy 5 algoritmu Diferenciální evoluce

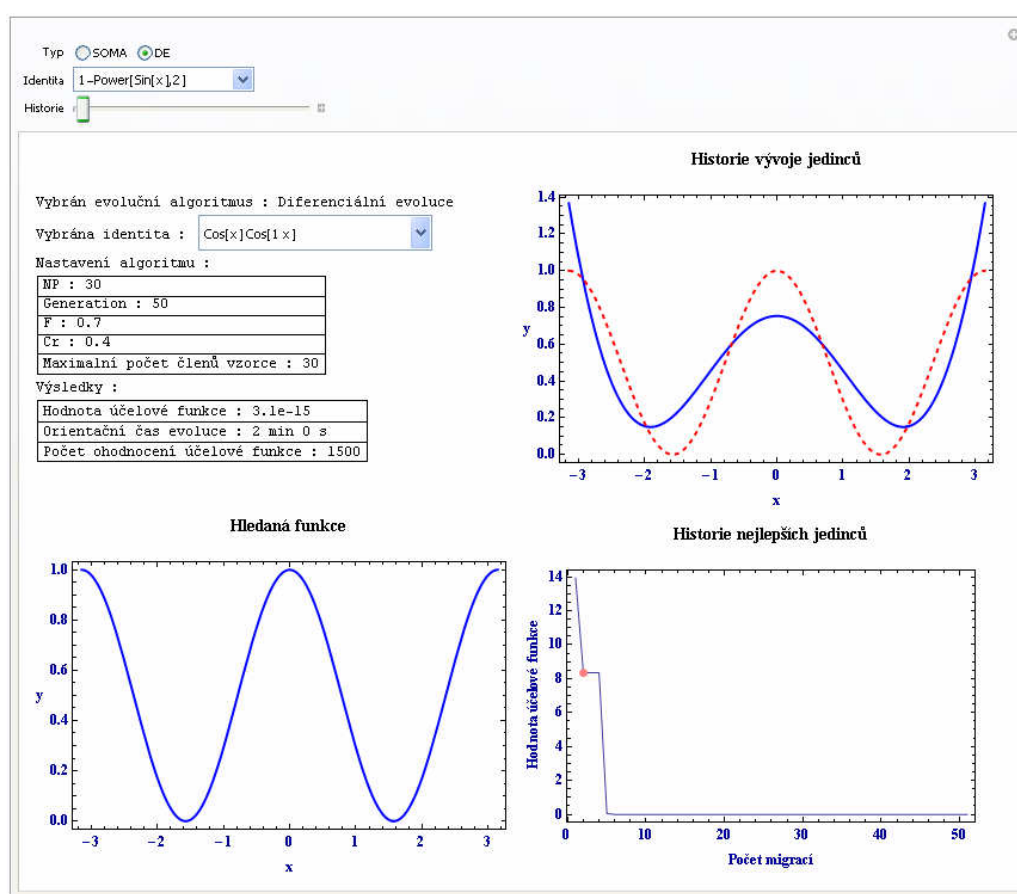
Historie nejlepších jedinců



Obrázek 32 : Graf historie jedinců úlohy 5 algoritmu Diferencialní evoluce

7 VÝUKOVÝ SOFTWARE

Součástí této diplomové práce je i výukový software sloužící jako názorná ukázka a prezentace nalezených identit pomocí analytického programování. Tato aplikace je napsána v softwaru Mathematica a jsou v ní uloženy historie výpočtů mnou nalezených identit. Student má možnost zvolit si algoritmus použitý pro hledání identity, a to algoritmus SOMA, popř. diferenciální evoluci. V aplikaci je uvedeno, s jakými parametry byla identita hledána, a jakých výsledků bylo dosaženo. V grafu lze pozorovat průběh evoluce na nejlepších jedincích.



Obrázek 33 : Ukázka výukové aplikace

7.1 Popis aplikace

Data aplikace jsou rozdělena do dvou datových struktur pojmenovaných „databazeDE“ a „databazeSOMA“. Data jsou uložena v maticové struktuře viz. tabulka.

Hledaná Identita 1	Nalezená identita 1 + Vstupní parametry algoritmu 1 + Výsledky výpočtu 1	...	Nalezená identita N1 + Vstupní parametry algoritmu N1 + Výsledky výpočtu N1
Hledaná Identita 2	Nalezená identita 1 + Vstupní parametry algoritmu 1 + Výsledky výpočtu 1	...	Nalezená identita N2 + Vstupní parametry algoritmu N2 + Výsledky výpočtu N2
Hledaná Identita 3	Nalezená identita 1 + Vstupní parametry algoritmu 1 + Výsledky výpočtu 1	...	Nalezená identita N3 + Vstupní parametry algoritmu N3 + Výsledky výpočtu N3
...
Hledaná Identita N	Nalezená identita 1 + Vstupní parametry algoritmu 1 + Výsledky výpočtu 1	...	Nalezená identita N4 + Vstupní parametry algoritmu N4 + Výsledky výpočtu N4

Tabulka 31 : Obecné schéma struktury dat

Z tabulky vyplývá, že počet nalezených identit jedné z hledaných identit nemusí odpovídat počtu ostatních hledaných identit. Data se do aplikace vkládají zapisováním vypočtených hodnot ve stanoveném pořadí.

7.2 Obsluha aplikace

Obsluha aplikace je velmi intuitivní. Po vybrání evolučního algoritmu a hledané identity se zobrazí údaje první nalezené identity. Tuto lze změnit, jestliže k hledané identitě bylo nalezeno více identit. Ke každé nalezené identitě jsou uvedeny parametry nastavení algoritmu a výsledky hledání. K dispozici jsou také grafy, na nichž lze pozorovat evoluční proces zobrazený z historie výpočtů.

ZÁVĚR

Tato práce se zabývá evoluční syntézou trigonometrických identit. V teoretické části diplomové práce jsou objasněny tři hlavní algoritmy evoluční syntézy struktur, a to genetické programování, gramatická evoluce a analytické programování. Analytické programování potřebuje pro svou činnost externí evoluční algoritmus. Z tohoto důvodu jsou popsány dva evoluční algoritmy, SOMA a diferenciální evoluce, které jsou taktéž použity v praktické části diplomové práce.

Analytické programování bylo shledáno jako velmi úspěšný algoritmus pro evoluční syntézu struktur. I když analytické programování je formou jisté transformace množin dat. V podstatě více záleží na kvalitě použitého evolučního algoritmu, a jak je tento algoritmus nastaven. Společně tvoří analytické programování a úspěšné evoluční algoritmy, jakými SOMA a diferenciální evoluce bezesporu jsou, velmi silný, a přitom relativně jednoduchý nástroj evoluční syntézy struktur.

Praktická část diplomové práce byla věnována objevování trigonometrických identit. Objevování bylo časově i výpočetně náročné, viz jednotlivé výpočty. Každá trigonometrická identita byla vypočtena desetkrát. Z tabulek výpočtů vyplývá, že algoritmus upřednostňuje již známe trigonometrické identity. Studenti se v matematice učí trigonometrické identity zpaměti. Výhodou tohoto přístupu je fakt, že není třeba si identity pamatovat, ale je možné je objevit pomocí metod evoluční syntézy struktur s možností objevit alternativní identity. Bylo jen pár případů, kdy byla nalezena alternativní neznámá identita. Většinu výsledků výpočtů bylo možné matematicky upravit do známých matematických vzorců. Hledání neznámých identit by bylo možné posílit vhodnou úpravou účelové funkce, a to penalizací známých identit.

Data ze zajímavých výpočtů byla ukládána a stala se součástí softwaru určeného pro výukové a prezentační účely. Tato aplikace byla naprogramována v softwaru Mathematica. Byly ukládány pouze zajímavé identity, hlavně z důvodů výpočetní náročnosti aplikace. Výuková a prezentační aplikace je intuitivní na ovládání a jsou v ní přehledně zaznamenány důležité informace o průběhu evoluce jednotlivých trigonometrických výpočtů.

ZÁVĚR V ANGLIČTINĚ

This master thesis deals with evolutionary synthesis of trigonometry identities. In theoretical part, are described three main algorithms of evolutionary synthesis of symbolic structures and it is genetic programming, grammatical evolution and analytical programming. Analytic programming needs for its work external evolutionary algorithm. For this reason, are describe two evolutionary algorithms, SOMA and differential evolution, which are also used in the practical part of this thesis.

Analytic programming was found as very successful algorithm for evolutionary synthesis of symbolic structures. Although this method is a form of transformation of data sets. Basically, it depends more on the quality of the used evolutionary algorithm, and how this algorithm set. Together, analytical programming and successful evolutionary algorithms, such as SOMA and differential evolution, are very powerful and relatively simple tools for evolutionary synthesis of symbolic structures.

The practical part was given to discovering of trigonometric identities. The discovery was time-consuming and computationally expensive, see the calculations. Each trigonometric identity was calculated ten times. Tables of calculations show that the algorithm inclination to already known trigonometric identities. Students learn mathematics trigonometric identities from memory. The advantage of this approach is that there is no need to remember the identities, but it can be discovered by the evolutionary synthesis of symbolic structures with the opportunity to discover the alternative identities. It was only a few cases where was found an alternative unknown identity. Most of the results of calculation can be mathematically adjust to known mathematical formulas. Finding unknown identity would be enhanced by appropriate adjustment cost function with penalty of known identities.

Data from the interesting calculations have been saved and become part of the software designed for teaching and presentation purposes. This application was programmed in software called Mathematica. They were saved only interesting identities, mainly due to computational complexity of this application. Training and presentation application is intuitive to use and there are recorded important information about the evolution of trigonometric identities.

SEZNAM POUŽITÉ LITERATURY

- [1] ZELINKA, I., OPLATKOVÁ, Z., OŠMERA, P., ŠEDA, M., VČELAŘ, F. Evoluční výpočetní techniky - principy a aplikace. BEN - technická literatura, Praha, 2008, ISBN 80-7300-218-3.
- [2] ZELINKA, I., OPLATKOVÁ, Z., NOLLE, L., Boolean Symmetry Function Synthesis by Means of Arbitrary Evolutionary Algorithms-Comparative Study, International Journal of Simulation Systems, Science and Technology, Volume 6, Number 9, August 2005, pages 44 - 56, ISSN: 1473-8031, online <http://ducati.doc.ntu.ac.uk/uksim/journal/Vol-6/No.9/cover.htm>, ISSN: 1473-804x.
- [3] OPLATKOVÁ, Z.: Metaevolution - Synthesis of Optimization Algorithms by means of Symbolic Regression and Evolutionary Algorithms, Lambert-Publishing, 2009, ISBN 978-8383-1808-0.
- [4] KOZA J. R.: Genetic Programming: on the programming of computers by means of natural selection, The Bradford Book, MIT Press, UK, 1992, ISBN 0-262-11170-5.
- [5] O'NEILL M., CONOR R.: Grammatical Evolution, Kluwer Academic Publishers, 2003, ISBN 1-4020-7444-1.
- [6] BANZHAF W. (ed.): Genetic Programming and Evolvable Machines, Vol. 7, Nr. 1, March, 2006, Springer, ISSN: 1389-2576.
- [7] ZELINKA, Ivan, New Optimization Techniques in Engineering / kap.7 "SOMA - Self Organizing Migrating Algorithm, Springer-Verlag
- [8] REKTORYS, K. a spol.: Přehled užití matematiky I.. Prometheus, Praha, 2003, 7. vydání. ISBN 80-7196-179-5
- [9] ODVÁRKO, O.: Goniometrie pro gymnázia. Prometheus, Praha 1994
- [10] KVASNIČKA, V., POSPÍCHAL, J. , TIŇO, P.: Evolučné algoritmy. Bratislava, STU 2000.
- [11] WOLFRAM S.: The Mathematica Book 5, Wolfram Media, 2003, ISBN 1-57955-022-3.

- [12] LANGDON, W. B., Genetic Programming and Data Structures, Springer ISBN 0-7923-8135-1
- [13] HUGOSSON J., HEMBERG E., BRABAYON A., O'NEILL M. Genotype Representations in Grammatical Evolution. Applied Soft Computing, pp.36-43 Vol.10
- [14] LAMPINEN, J., ZELINKA, I., New Ideas in Optimization & Mechanical Engineering Design Optimization by Differential Evolution. Volume 1. London: McGraw-Hill, 1999. 20 p. ISBN 007-709506-5
- [15] Science World - Programy, které se píší samy [online]. 2001 [cit. 2011-05-15]. Dostupný z WWW: < <http://scienceworld.cz/technologie/programy-ktere-se-pisi-samy-4528>>.
- [16] EBNF [online]. [s.l.] : [s.n.], 2005 [cit. 2011-05-15]. Dostupné z WWW: <<http://www.cs.cmu.edu/~pattis/misc/ebnf.pdf>>
- [17] OŠMERA, Pavel. Paralelní gramatická evoluce pro optimalizaci elektronických obvodů [online]. [s.l.] : [s.n.], [2009] [cit. 2011-05-15]. Dostupné z WWW: <<http://dai.fmph.uniba.sk/events/kuz2009/prispevky-pdf/osmera.pdf>>

SEZNAM POUŽITÝCH SYMBOLŮ A ZKRATEK

DE	Diferenciální evoluce
SOMA	Samoorganizující se migrační algoritmus
PRT	Parametr algoritmu SOMA ovlivňující perturbaci algoritmu
D	Dimenze problému
NP	Parametr algoritmu diferenciální evoluce určující počet jedinců v populaci
F	Parametr algoritmu diferenciální evoluce určující mutace v populaci
CR	Parametr algoritmu diferenciální evoluce ovlivňující křížení – práh křížení
LISP	Funkcionální programovací jazyk
D_{\max}	Parametr genetického programování určující maximální hloubku stromu
N	Určuje konečnou množinu neterminálních symbolů gramatiky
T	Určuje konečnou množinu terminálních symbolů gramatiky
P	Určuje přepisovací pravidla gramatiky
S	Určuje počáteční symbol gramatiky
F_{cost}	Označení účelové funkce
$f(x)$	Označení hledané funkce
$g(x)$	Označení nalezené funkce

SEZNAM OBRÁZKŮ

Obrázek 1 : Ukázka migrace jedinců k leaderovi algoritmu SOMA	14
Obrázek 2 : Ukázka terminálních a neterminálních symbolů výrazu $\sin(5 + x)$	17
Obrázek 3 : Příklad datové struktury stromu pro výraz $5 + 6\sin(x)$	18
Obrázek 4 : Ukázka křížení genetického programování	19
Obrázek 5 : Ukázka mutace genetického programování	20
Obrázek 6 : Ukázka softwaru Mathematica	28
Obrázek 7 : Zadání hledané funkce	29
Obrázek 8 : Zadání funkcí použitých analytickým programováním	30
Obrázek 9 : Zadání účelové funkce	30
Obrázek 10 : Nastavení algoritmu SOMA	31
Obrázek 11 : Příklad hotového výpočtu hledané identity $\cos(x)$	31
Obrázek 12 : Popis účelové funkce	33
Obrázek 13 : Graf porovnání úlohy 1 algoritmu SOMA	35
Obrázek 14 : Graf historie jedinců úlohy 1 algoritmu SOMA	36
Obrázek 15 : Graf porovnání úlohy 2 algoritmu SOMA	38
Obrázek 16 : Graf historie jedinců úlohy 2 algoritmu SOMA	38
Obrázek 17 : Graf porovnání úlohy 3 algoritmu SOMA	40
Obrázek 18 : Graf historie jedinců úlohy 3 algoritmu SOMA	41
Obrázek 19 : Graf porovnání úlohy 4 algoritmu SOMA	43
Obrázek 20 : Graf historie jedinců úlohy 4 algoritmu SOMA	43
Obrázek 21 : Graf porovnání úlohy 5 algoritmu SOMA	45
Obrázek 22 : Graf historie jedinců úlohy 5 algoritmu SOMA	46
Obrázek 23 : Graf porovnání úlohy 1 algoritmu Diferencialní evoluce	48
Obrázek 24 : Graf historie jedinců úlohy 1 algoritmu Diferencialní evoluce	49
Obrázek 25 : Graf porovnání úlohy 2 algoritmu Diferencialní evoluce	51
Obrázek 26 : Graf historie jedinců úlohy 2 algoritmu Diferencialní evoluce	51
Obrázek 27 : Graf porovnání úlohy 3 algoritmu Diferencialní evoluce	53
Obrázek 28 : Graf historie jedinců úlohy 3 algoritmu Diferencialní evoluce	54
Obrázek 29 : Graf porovnání úlohy 4 algoritmu Diferencialní evoluce	56
Obrázek 30 : Graf historie jedinců úlohy 4 algoritmu Diferencialní evoluce	56
Obrázek 31 : Graf porovnání úlohy 5 algoritmu Diferencialní evoluce	58

Obrázek 32 : Graf historie jedinců úlohy 5 algoritmu Diferencialní evoluce.....	59
Obrázek 33 : Ukázka výukové aplikace.....	60

SEZNAM TABULEK

Tabulka 1 : Ukázka populace.....	11
Tabulka 2 : Ukázka křížení diferenciální evoluce	16
Tabulka 3 : Ukázka tabulky přepisovacích pravidel	22
Tabulka 4 : Ukázka jedince metody gramatické evoluce.....	22
Tabulka 5 : Ukázka práce gramatické evoluce	23
Tabulka 6 : Ukázka jedince analytického programování.....	24
Tabulka 7 : Ukázka seříděných terminálních a neterminálních symbolů	24
Tabulka 8 : Ukázka ukazatelů na jednotlivé symboly	25
Tabulka 9 : Ukázka sestavování výrazu analytického programování	25
Tabulka 10 : Příklad trigonometrických identit	26
Tabulka 11 : Nastavení parametrů úlohy 1 algoritmu SOMA	34
Tabulka 12 : Výsledky úlohy 1 algoritmu SOMA	35
Tabulka 13 : Nastavení parametrů úlohy 2 algoritmu SOMA	37
Tabulka 14 : Výsledky úlohy 2 algoritmu SOMA	37
Tabulka 15 : Nastavení parametrů úlohy 3 algoritmu SOMA	39
Tabulka 16 : Výsledky úlohy 3 algoritmu SOMA	40
Tabulka 17 : Nastavení parametrů úlohy 4 algoritmu SOMA	42
Tabulka 18 : Výsledky úlohy 4 algoritmu SOMA	42
Tabulka 19 : Nastavení parametrů úlohy 5 algoritmu SOMA	44
Tabulka 20 : Výsledky úlohy 5 algoritmu SOMA	45
Tabulka 21 : Nastavení parametrů úlohy 1 algoritmu Diferenciální evoluce	47
Tabulka 22 : Výsledky úlohy 1 algoritmu Diferenciální evoluce	48
Tabulka 23 : Nastavení parametrů úlohy 2 algoritmu Diferenciální evoluce	50
Tabulka 24 : Výsledky úlohy 2 algoritmu Diferenciální evoluce	50
Tabulka 25 : Nastavení parametrů úlohy 3 algoritmu Diferenciální evoluce	52
Tabulka 26 : Výsledky úlohy 3 algoritmu Diferenciální evoluce	53
Tabulka 27 : Nastavení parametrů úlohy 4 algoritmu Diferenciální evoluce	54
Tabulka 28 : Výsledky úlohy 4 algoritmu Diferenciální evoluce	55
Tabulka 29 : Nastavení parametrů úlohy 5 algoritmu Diferenciální evoluce	57
Tabulka 30 : Výsledky úlohy 5 algoritmu Diferenciální evoluce	58
Tabulka 31 : Obecné schéma struktury dat	61

SEZNAM PŘÍLOH

P 1 CD-ROM

PŘÍLOHA P I: CD - ROM

Přiložené CD obsahuje tuto práci v elektronické podobě. Dále obsahuje notebooky softwaru Mathematica, pomocí nichž byly prováděny všechny výpočty. Obsažen je i výukový program, taktéž napsán v softwaru Mathematica.