

# **Diskrétní evoluční algoritmy v prostředí Mathematica**

Discrete Evolutionary Algorithms in the Mathematica Environment.

Bc. Dalibor Klučka

---

Diplomová práce  
2013



Univerzita Tomáše Bati ve Zlíně  
Fakulta aplikované informatiky

---

Univerzita Tomáše Bati ve Zlíně

Fakulta aplikované informatiky

akademický rok: 2012/2013

## ZADÁNÍ DIPLOMOVÉ PRÁCE

(PROJEKTU, UMĚLECKÉHO DÍLA, UMĚLECKÉHO VÝKONU)

Jméno a příjmení: **Bc. Dalibor Klučka**  
Osobní číslo: **A11437**  
Studijní program: **N3902 Inženýrská informatika**  
Studijní obor: **Počítačové a komunikační systémy**  
Forma studia: **prezenční**

Téma práce: **Diskrétní evoluční algoritmy v prostředí Mathematica.**

Zásady pro vypracování:

1. Vypracujte literární rešerši na dané téma.
2. Popište způsoby úpravy a principy diskrétních evolučních algoritmů.
3. Naprogramujte vybrané strategie zvoleného diskrétního evolučního algoritmu v prostředí Mathematica.
4. Otestujte algoritmus na sadě vybraných testovacích problémů.
5. Výsledky testování přehledně graficky a tabulkově zobrazte.

Rozsah diplomové práce:

Rozsah příloh:

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

Seznam odborné literatury:

1. ZELINKA, Ivan. Umělá inteligence v problémech globální optimalizace. BEN, 2002, 190 s. ISBN 80-7300-069-5.
2. ZELINKA, Ivan. Evoluční výpočetní techniky: principy a aplikace. 1. vyd. Praha: BEN - technická literatura, 2009, 534 s. ISBN 978-80-7300-218-3.
3. DE JONG, Kenneth A. Evolutionary computation: a unified approach. Cambridge: MIT Press, 2006, ix, 256 s. ISBN 02-620-4194-4.
4. MAŘÍK, V., ŠTĚPÁNKOVÁ, O., LAŽANSKÝ, J.: Umělá inteligence, Academia, 1993, ISBN 80-200-0496-3.
5. MAŘÍK, V., ŠTĚPÁNKOVÁ, O., LAŽANSKÝ, J.: Umělá inteligence 4., Academia, 2003, ISBN 80-200-1044-0.
6. ZELINKA, Ivan, Zuzana OPLATKOVÁ a Roman ŠENKEŘÍK. Aplikace umělé inteligence. Vyd. 1. Zlín: Univerzita Tomáše Bati ve Zlíně, 2010, 151 s. ISBN 978-80-7318-898-6.
7. PRICE, Kenneth V, Rainer M STORN a Jouni A LAMPINEN. Differential evolution: a practical approach to global optimization [online]. Berlin: Springer, 2005.

Vedoucí diplomové práce:

**Ing. Roman Šenkeřík, Ph.D.**

Ústav informatiky a umělé inteligence


Datum zadání diplomové práce:

**26. února 2013**


Termín odevzdání diplomové práce:

**31. května 2013**

Ve Zlíně dne 26. února 2013

  
prof. Ing. Vladimír Vašek, CSc.  
*děkan*



  
prof. Ing. Karel Vlček, CSc.  
*ředitel ústavu*

## **ABSTRAKT**

Tato práce je zaměřena na testování diferenciální evoluce pracující s diskrétními hodnotami. Teoretická část je zaměřena na problematiku evolučních a optimalizačních algoritmů, je zde popsán princip diferenciální evoluce a její využití pro práci s diskrétními hodnotami a popsáno vývojové prostředí Wolfram Mathematica. Praktická část je zaměřena na samotné testování diferenciální evoluce a její využití při řešení problému obchodního cestujícího. Velká část práce je zaměřena na porovnání výsledků pro různou hranici oprav parametrů jedinců. Výsledky jsou přehledně zobrazeny v tabulkách a grafech.

Klíčová slova: Diferenciální evoluce, Evoluční algoritmy, Wolfram Mathematica, Obchodní cestující

## **ABSTRACT**

The thesis is focused on the testing of the differential evolution working with discrete values. The theoretical part of this thesis is focused on the issues of evolutionary and optimization algorithms, there is a describe of principal of the differential evolution and its application for working with discrete values. There is also a describe of Wolfram Mathematica development environment. The practical part is focused on the testing of the differential evolution and its usage in solving the traveling salesman problem. Significant part of the thesis is focused on the comparison of the results for different repairment boundary. The results are clearly displayed in the tables and graphs.

Keywords: Differential evolution, Evolutionary algorithms, Wolfram Mathematica, Traveling Salesman Problem

Velmi děkuji vedoucímu práce Ing. Romanu Šenkeříkovi, Ph.D. za odborné vedení, ochotu a cenné rady a připomínky při řešení problémů.

**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**

<b>ÚVOD</b> .....	<b>10</b>
<b>I TEORETICKÁ ČÁST</b> .....	<b>11</b>
<b>1 OPTIMALIZAČNÍ ALGORITMY</b> .....	<b>12</b>
1.1 SLOŽITÉ OPTIMALIZAČNÍ ÚLOHY .....	12
1.1.1 Problém batohu .....	12
1.1.2 Problém obchodního cestujícího .....	12
1.2 NO FREE LUNCH TEORÉM .....	13
<b>2 EVOLUČNÍ VÝPOČETNÍ TECHNIKY</b> .....	<b>14</b>
2.1 EVOLUČNÍ ALGORITMY .....	14
2.1.1 Výhody evolučních algoritmů.....	14
2.1.2 Diskrétní evoluční algoritmy.....	15
2.2 GENETICKÉ ALGORITMY .....	15
2.2.1 Reprezentace vlastností a informací.....	15
2.2.2 Princip genetických algoritmů .....	16
<b>3 DIFERENCIÁLNÍ EVOLUCE</b> .....	<b>17</b>
3.1 PARAMETRY A TERMINOLOGIE .....	17
3.2 MUTACE.....	18
3.3 KŘÍŽENÍ.....	18
3.4 PRINCIP .....	19
3.5 STAGNACE.....	20
<b>4 TESTOVACÍ FUNKCE</b> .....	<b>22</b>
4.1 UNIMODÁLNÍ FUNKCE .....	22
4.1.1 První de Jongova funkce .....	22
4.1.2 Třetí de Jongova funkce .....	23
4.2 MULTIMODÁLNÍ FUNKCE.....	24
4.2.1 Rossenbrockovo sedlo (druhá de Jongova funkce).....	24
4.2.2 Schwefelova funkce .....	26
4.2.3 Rastriginova funkce .....	27
<b>5 WOLFRAM MATHEMATICA</b> .....	<b>29</b>
5.1 VZNIK .....	29
5.2 VÝVOJOVÉ PROSTŘEDÍ .....	29
5.3 INTERAKTIVNÍ NÁPOVĚDA .....	30
<b>II PRAKTICKÁ ČÁST</b> .....	<b>32</b>
<b>6 ALGORITMUS DIFERENCIÁLNÍ EVOLUCE</b> .....	<b>33</b>
6.1 REPAIRMENT .....	33
<b>7 OBCHODNÍ CESTUJÍCÍ ŘEŠENÝ HRUBOU SILOU A DIFERENCIÁLNÍ EVOLUCÍ</b> .....	<b>35</b>
7.1 POROVNÁNÍ NALEZENÝCH ŘEŠENÍ.....	35
7.1.1 Pro 5 měst.....	36
7.1.2 Pro 6 měst.....	36
7.1.3 Pro 7 měst.....	37

7.1.4	Pro 8 měst.....	38
7.1.5	Pro 9 měst.....	38
7.2	SROVNÁNÍ.....	39
<b>8</b>	<b>TESTOVÁNÍ ZÁVISLOSTI NA REPAIRMENT.....</b>	<b>41</b>
8.1	DIMENZE 20, GENERACE 500 A VELIKOST POPULACE 30 .....	41
8.1.1	Repairment 0,3 .....	41
8.1.2	Repairment 0,5 .....	42
8.1.3	Repairment 0,7 .....	44
8.1.4	Porovnání průměrů a mediánů pro různou hranici oprav.....	46
8.1.5	Nízká hranice oprav .....	47
8.2	DIMENZE 20, GENERACE 5000 A VELIKOST POPULACE 50 .....	48
8.2.1	Repairment 0,3 .....	48
8.2.2	Repairment 0,5 .....	50
8.2.3	Repairment 0,7 .....	51
8.2.4	Porovnání průměrů a mediánů pro různou hranici oprav.....	54
8.3	DIMENZE 50, GENERACE 500 A VELIKOST POPULACE 30 .....	55
8.3.1	Repairment 0,3 .....	55
8.3.2	Repairment 0,5 .....	57
8.3.3	Repairment 0,7 .....	58
8.3.4	Porovnání průměrů a mediánů pro různou hranici oprav.....	60
8.4	DIMENZE 50, GENERACE 5000 A VELIKOST POPULACE 75 .....	61
8.4.1	Repairment 0,3 .....	61
8.4.2	Repairment 0,5 .....	63
8.4.3	Repairment 0,7 .....	64
8.4.4	Porovnání průměrů a mediánů pro různou hranici oprav.....	66
8.5	DIMENZE 100, GENERACE 500 A VELIKOST POPULACE 30 .....	67
8.5.1	Repairment 0,3 .....	67
8.5.2	Repairment 0,5 .....	68
8.5.3	Repairment 0,7 .....	70
8.5.4	Porovnání průměrů a mediánů pro různou hranici oprav.....	71
8.6	DIMENZE 100, GENERACE 5000 A VELIKOST POPULACE 75 .....	72
8.6.1	Repairment 0,3 .....	72
8.6.2	Repairment 0,5 .....	74
8.6.3	Repairment 0,7 .....	75
8.6.4	Porovnání průměrů a mediánů pro různou hranici oprav.....	76
<b>9</b>	<b>KOMPLETNÍ SROVNÁNÍ VÝSLEDKŮ.....</b>	<b>78</b>
9.1	NEJLEPŠÍ JEDINCI.....	78
9.2	PRŮMĚRNÉ HODNOTY NEJLEPŠÍCH JEDINCŮ.....	80
9.3	MEDIÁN NEJLEPŠÍCH JEDINCŮ .....	82
9.4	SMĚRODATNÉ ODCHYLKY NEJLEPŠÍCH JEDINCŮ .....	84
	<b>ZÁVĚR .....</b>	<b>87</b>
	<b>CONCLUSION .....</b>	<b>88</b>
	<b>SEZNAM POUŽITÉ LITERATURY.....</b>	<b>89</b>
	<b>SEZNAM POUŽITÝCH SYMBOLŮ A ZKRATEK.....</b>	<b>90</b>
	<b>SEZNAM OBRÁZKŮ .....</b>	<b>91</b>

---

SEZNAM TABULEK.....94

## ÚVOD

Evoluční algoritmy se nechaly inspirovat evoluční teorií a principy, které se běžně vyskytují v přírodě kolem nás. Jsou schopny řešit celou škálu problémů a vyznačují se vysokou robustností. Jejich největší uplatnění najdeme v oblasti optimalizace, jež má velký význam pro řešení mnoha úloh, jak teoretických, tak známých z reálného světa. K rozvoji optimalizačních algoritmů významně napomáhá rozvoj výpočetní techniky, která zvyšuje jejich efektivnost.

Evolučních algoritmů je velké množství, každý algoritmus je totiž vhodný k řešení jiného problému a neexistuje takový algoritmus, který by se spolehlivě hodil k řešení všech možných problémů. Jedním z takových algoritmů je diferenciální evoluce.

Diferenciální evoluce je algoritmus inspirovaný chováním jedinců v přírodě, jejich vzájemným křížením a mutací. Existuje několik verzí diferenciální evoluce a je schopna pracovat jak v oboru reálných čísel, tak v oboru celočíselných, tedy diskrétních, hodnot.

V teoretické části práce jsou vysvětleny principy evolučních a genetických algoritmů, je zde podrobně vysvětlen princip diferenciální evoluce a také principy vybraných optimalizačních úloh. Dále je zde popis vybraných testovacích funkcí, které slouží k testování kvality algoritmu a popis vývojového prostředí Wolfram Mathematica.

V praktické části je popsán vytvořený algoritmus diferenciální evoluce upravený pro práci s diskrétními hodnotami a následné řešení problému obchodního cestujícího tímto algoritmem a jeho srovnání s řešením hrubou silou. Další část práce je věnována potřebě opravit část parametrů jedinců při jejich křížení a mutaci a možnost nastavit hranici těchto oprav. Následně jsou srovnána řešení pro různá nastavení této hranice, jejich vykreslení do grafů a pojednání o jejich závislosti na nastavení jednotlivých parametrů diferenciální evoluce.

## **I. TEORETICKÁ ČÁST**

## 1 OPTIMALIZAČNÍ ALGORITMY

Optimalizační algoritmy slouží k řešení problémů optimalizačních úloh, u kterých je řešení analytickou metodou nevhodné, časově náročné nebo ve velké většině případů dokonce nemožné ani za využití nejmodernější výpočetní techniky.

Optimalizační algoritmy najdou uplatnění tam, kde je třeba nalézt optimální řešení daného problému, například optimální trajektorie, optimální nastavení parametrů apod. a kde řešení analytickou metodou (např. útok hrubou silou) je složité, případně nemožné. Řešený problém můžeme převést na matematické vyjádření účelové funkce a podle ní hledat optimální řešení problému.

### 1.1 Složité optimalizační úlohy

Jedná se o úlohy, které jsou pro větší množství vstupních parametrů často neřešitelné hrubou silou (*brute-force*), tzn. nalezením nejlepšího řešení ze všech možných kombinací (resp. permutací) řešení daného problému.

Mezi tyto úlohy patří například problém Obchodního cestujícího a problém Batohu, které spadají do kategorie složitých optimalizačních úloh

#### 1.1.1 Problém batohu

Problém batohu je složitá kombinatorická optimalizační úloha, jejíž cílem je vybrat z množiny předmětů, které jsou určeny svojí hodnotou a hmotností, maximální množství předmětů, které batoh unese (tzn. nebude přetížen), tak, aby hodnota předmětů uvnitř batohu byla maximální možná.

Je zřejmé, že pro větší množství předmětů je tento problém neřešitelný hrubou silou. Proto je vhodné tuto úlohu vyřešit právě některým z optimalizačních algoritmů.

#### 1.1.2 Problém obchodního cestujícího

Problém obchodního cestujícího (*traveling salesman problem* - TSP) je složitá diskretní optimalizační úloha, jejíž cílem je nalezení nejkratší možné cesty mezi  $n$  body (městy) tak, aby algoritmus prošel všemi zadanými body přesně jedenkrát a na závěr se vrátil do počátečního bodu.

Matematicky lze tento problém vyjádřit vztahem:

$$L = \sum_{i=1}^{N-1} \Delta(C_i, C_{i+1}) + \Delta(C_N, C_1) \quad (1)$$

kde  $\{C_1, C_2, \dots, C_N\}$  je množina všech měst,  $N$  je jejich počet a  $\Delta$  je vzdálenost mezi městy. Délka cesty  $L$  je pak dána sumou těchto vzdáleností.

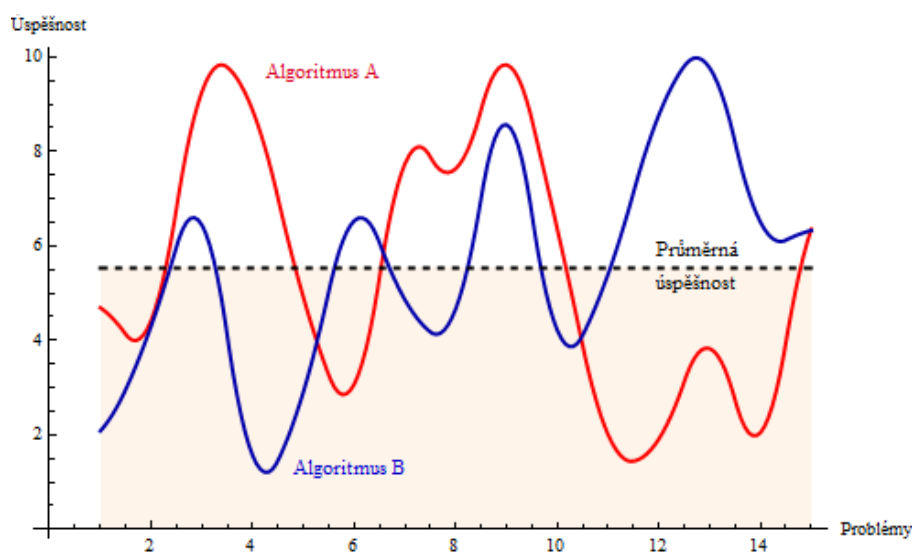
Složitost tohoto algoritmu je  $n!$ , což je překážkou pro řešení tohoto problému hrubou silou. Proto se také zde uplatňuje řešení pomocí optimalizačních (evolučních) algoritmů, které jsou schopny vyřešit úlohu TSP pro mnohonásobně větší počet zadaných bodů v reálném čase.

Právě na problému obchodního cestujícího a testování jeho řešení je založena podstatná část této diplomové práce a podrobně je vše popsáno v praktické části této práce.

## 1.2 No Free Lunch Teorém

Pro optimalizační algoritmy existuje jistá statisticky významná skutečnost, která se v odborné terminologii nazývá *No Free Lunch Teorém* (NFL). Velmi stručně řečeno, je to teorém, ve kterém se tvrdí, že neexistuje algoritmus, který by dokázal řešit všechny problémy lépe než jiné algoritmy, neboli existuje podmnožina problémů, pro které je algoritmus A lepší než algoritmus B a naopak. [2]

Lze tedy říct, že nelze použít jeden algoritmus k řešení jakéhokoliv problému, ale pro řešení různých problémů je vhodné použít různých algoritmů určených speciálně pro daný typ problému. Graficky je tato skutečnost vyjádřena na Obr. 1. na dvou algoritmech, jejichž úspěšnost je sice stejná, ale každý je vhodný pro řešení jiného typu úloh.



Obr. 1. No Free Lunch Teorém v grafickém vyjádření

## 2 EVOLUČNÍ VÝPOČETNÍ TECHNIKY

Evoluční výpočetní techniky (EVT) jsou numerické algoritmy, které vycházejí ze základních principů Darwinovy a Mendelovy teorie evoluce, jejíž hlavní ideou je předávání rodičovského genomu novým potomkům a následné uvolnění životního prostoru potomkům. [2]

Evoluční výpočetní techniky stojí v drtivé většině na existenci evolučních algoritmů.

### 2.1 Evoluční algoritmy

Biologicky inspirované algoritmy nazývané evoluční algoritmy (EA) jsou založeny na podobnosti se základními principy evoluce tak, jak ji známe z běžného prostředí a jak byla popsána Ch. Darwinem či J. G. Mendelem.

Obecným principem EA je vytváření nových generací jedinců (množinou možných řešení) za pomoci generací předchozích. Takto nově vytvoření jedinci získávají vlastnosti ovlivněné vlastnostmi jedinců z generace předchozí a tím je také ovlivněna jejich kvalita a pravděpodobnost jejich dalšího "přežití". Jedná se tedy o "přírozený výběr", kdy jsou ze skupiny jedinců vybíráni ti, kteří nejlépe vyhovují požadovaným kritériím. Starší a méně kvalitní jedinci pak uvolňují své místo těm novějším a kvalitnějším. Jedná se tedy využití technik v analogii s přírodními zákony, kdy podle Darwinovi teorie přežijí jen ti nejsilnější. Po dostatečném množství opakování tohoto cyklu dospějeme tím pádem ke generaci jedinců, kteří mají nejlepší vlastnosti a jsou tedy nejlepším řešením problému.

Evoluční algoritmy se vyznačují vysokou robustností a schopností řešit širokou škálu obecně složitých problémů. Mají schopnost nalézt nejlepší možné řešení daného problému, ale také lze s jejich pomocí nalézt více optimálních řešení a to jak v oblasti reálných čísel, tak v oblasti diskretních (celočíslných) hodnot. Největší využití mají EA v oblasti optimalizace, klasifikace a rozhodování.

#### 2.1.1 Výhody evolučních algoritmů

Evoluční algoritmy lze obvykle velmi jednoduše naprogramovat, jsou hybridní, tedy mohou pracovat s kombinacemi různých typů čísel jako real, integer, množina čísel. Jedinec se nemusí převádět do binárního kódu, jako je tomu například u genetických algoritmů, ale lze pracovat přímo s dekadickými čísly.

V porovnání s klasickými metodami je výhodná jejich rychlost, jelikož dané řešení dokážou nalézt mnohem rychleji. Další výhodou je schopnost nalézt i obtížné extrémy (například plocha s dírou v rovině) a schopnost nalézt vícenásobné řešení. [6]

### 2.1.2 Diskrétní evoluční algoritmy

Evoluční algoritmy lze použít jak pro řešení problémů v oboru reálných hodnot, tak pro řešení problémů v oboru diskrétních, tedy celočíselných hodnot, případně hodnot jen v určitém intervalu.. To znamená, že vývoj jedinců neprobíhá plynule po spojitě křivce, ale hodnota účelové funkce "skáče" v různých hodnotách po nespojitých křivkách.

Převážná část této práce je řešena právě pro diskrétní algoritmus diferenciální evoluce (kapitola 3). Podrobně je popsána v praktické části této práce.

## 2.2 Genetické algoritmy

Genetické algoritmy (GA) jsou založeny na myšlenkách vycházejících z Darwinovy teorie o vývoji druhů a také většina termínů a technik používaných u GA je převzata z biologie (generace, populace, jedinec, rodič, křížení, dědičnost, mutace). Genetické algoritmy jsou jednoduché a obecně slouží k řešení optimalizačních problémů.

### 2.2.1 Reprezentace vlastností a informací

Základní pojmy pro popis

- Gen - jsou to všechny parametry jedince, které určují jeho vlastnosti. Jedná se o nejmenší nedělitelnou část chromozómu.
- Chromozóm - jedná se o řetězec informací tvořený ze všech genů jedince, který nese vlastnosti a chování každého jedince. Většinou je vyjádřen v binární podobě, ale není to podmínkou (lze použít také reálná čísla, vektory, matice, křivky, apod.).
- Populace - je to skupina jedinců v rámci jedné generace. Každý jedinec v ní je popsán svými chromozómy.
- Vhodnost (fitness) - číselné ohodnocení každého jedince. Pro každý problém je třeba sestavit vhodnou účelovou (fitness) funkci, která ohodnotí jedince číselnou hodnotou. Vhodné je toto číslo vyjadřovat v rozmezí 0 až 1, ale lze použít i číslo z libovolného intervalu.

### 2.2.2 Princip genetických algoritmů

Principem GA je tvorba nových generací, tzv. potomků z populace rodičů. Nově vytvořené populace probíhají evolucí v každé další generaci a tím se zlepšují vlastnosti jedinců, kteří reprezentují jednotlivá řešení daného problému. K tomuto slouží určité operace, které probíhají nad celou populací v každém cyklu a vytváří tak evolucí vylepšenou novou generaci. Tyto operace jsou:

- Selekce - neboli výběr rodičů k vytvoření dalšího potomka. Výběr může probíhat několika způsoby. Rodiče se mohou volit například podle jejich kvality (hodnoty účelové funkce), kdy lepší jedinci mají větší pravděpodobnost stát se rodičem (tzv. *ruleta*); výběrem nejlepšího jedince z náhodně vybrané skupiny (*turnajová metoda*); rozdělením jedinců podle hodnoty účelové funkce a jejich následným vyloučením nejméně vhodných jedince (*ořezávání*); či nejjednodušším způsobem náhodného výběru z celé populace jedinců, bez ohledu na jejich kvalitu (*náhodný výběr*).
- Křížení - po výběru rodičů následuje křížení, tedy výměna jejich chromozómů. Tato výměna může mít různý rozsah. Nejjednodušší je rozdělení chromozómů na dvě části a jejich výměna (*křížení jednobodové*), čímž vzniknout dva noví jedinci, dojde k jejich ohodnocení a rozhodnutí, zda do nové populace postoupí oba nebo pouze jeden z nich. Podobným typem je rozdělení chromozómů na tři části a výměna prostřední z těchto částí (*křížení dvoubodové*). Dále je možno použít složitějšího křížení, kdy se chromozómy rozdělí na více částí a výměna chromozómů probíhá různě, případně náhodně (*křížení vícebodové*).
- Mutace - poslední operací je mutace, kdy se invertují hodnoty některých méně vhodných genů v chromozómu jedince. Je to z důvodu možnosti výskytu nové vlastnosti, která se u rodičů nevykytovala a tím pádem může vést k nalezení lepšího řešení.

### 3 DIFERENCIÁLNÍ EVOLUCE

Diferenciální evoluce (DE) je poměrně nový typ evolučního algoritmu (od r. 1995), který vyvinuli a poprvé použili Ken Price a Rainer Storm. Jeho schéma je dost podobné algoritmům genetickým, s nimiž má několik společných rysů, jako je například tvorba potomků (např. pomocí čtyř rodičů a ne dvou, jak je tomu u genetických algoritmů), opakování se v generacích apod. [2]

#### 3.1 Parametry a terminologie

Stejně jako je tomu u jiných evolučních algoritmů, je i celá činnost a výsledná kvalita průběhu diferenciální evoluce určena dle nastavení jejích řídicích parametrů. Těmito parametry jsou:

- Dimenze (*Dim*) - jedná se o počet argumentů účelové funkce. Jinak řečeno, jde o počet genů jedince, které určují jeho kvalitu (hodnotu účelové funkce). Velikost hodnoty dimenze je dána formulací řešeného problému.
- Velikost populace (*NP*) - hodnota *NP* by po zkušenostech s diferenciální evolucí měla nabývat hodnot od deseti do stonásobku hodnoty *Dim* (pro vysoce multimodální funkce). Pro diskrétní algoritmus lze však použít menší populaci v intervalu 25 až 75. Nejmenší možná hodnota *NP* by však neměla být nižší než 4, jelikož to je nejmenší možná velikost populace, při které je DE ještě schopna pracovat.
- Práh křížení (*CR*) - jinak nazýván též jako crossover. Je to hodnota v rozmezí 0 až 1, která výrazně ovlivňuje evoluční proces. V případě, že se *CR* nastaví na 0, dojde k tomu, že se mutace nedostane do zkušebního jedince, který díky tomu bude čistou kopií aktuálního (čtvrtého rodiče). Vývoj evoluce se pak zastaví. V případě, že *CR* bude nastaven na 1, bude zkušební jedinec tvořen pouze ze tří náhodně vybraných rodičů - jedinců z populace a diferenciální evoluce se bude spíše podobat náhodnému hledání nežli evolučnímu algoritmu (všichni tři jedinci, byť evolučně vytvoření, jsou náhodně vybíráni bez ohledu na jejich kvalitu). Je proto vhodné, aby *CR* nikdy nenabývalo těchto hodnot. [6]
- Mutační konstanta (*F*) - je to číselná hodnota v rozmezí 0 až 2, pomocí které dochází k mutaci při tvorbě nového jedince.

- Generace (*Gen*) - udává počet generací, tedy evolučních cyklů, během kterých dochází k vývoji populace. Horní hranice je neomezená a je třeba nastavit na takovou hodnotu, aby DE dospěla k optimálnímu řešení za optimální čas.

### 3.2 Mutace

V evolučních algoritmech stejně jako v klasické evoluční teorii hraje životně důležitou úlohu tzv. mutace. Diferenciální evoluce má tu zvláštnost, že k vytvoření dalšího potomka je potřeba ne dvou, ale čtyř rodičů. Pro každého jedince jsou náhodně vybráni tři další nestejní jedinci z populace ( $r_1, r_2, r_3$ ). Pomocí těchto tří jedinců se pak vytvoří tzv. šumový vektor  $v$ , který není ničím jiným nežli mutací kombinace oněch tří náhodně vybraných rodičů. [6]

Existuje několik variant, jak lze mutaci provést. Základní variantou je DE/Rand/1/Bin, který je popsán rovnicí:

$$v = x_{r_1,j}^G + F \cdot (x_{r_2,j}^G - x_{r_3,j}^G) \quad (2)$$

Zde je mutace provedena tak, že se rozdíl druhého a třetího náhodně vybraného rodiče vynásobí mutační konstantou  $F$  a takto získaný výsledný vektor se přičte ke prvnímu náhodně vybranému rodiči. Obecně na pořadí nezáleží, jelikož jsou stejně všichni tři rodiče vybírání náhodně.

### 3.3 Křížení

V případě diferenciální evoluce je další zvláštností to, že křížení nastává až po mutačním procesu, zatímco např. u genetických algoritmů je první vytvořen potomek křížením a teprve po té dochází k jeho mutaci. Proces křížení u diferenciální evoluce znamená, že se ze čtvrtého doposud nepoužitého rodiče a šumového - zmutovaného vektoru vytvoří nový jedinec, tzv. zkušební vektor. Ten se vytváří za pomoci tzv. konstanty prahu křížení CR a to tak, že se v cyklu vybírají korespondující parametry ze čtvrtého a šumového jedince (oba první parametry, oba druhé parametry, ...) a pro každou takto vybranou dvojici je generováno náhodné číslo. Pokud je menší než CR, do příslušného parametru ve zkušebním vektoru se přesune parametr z jedince šumového a naopak. Podmínku pro přiřazování parametrů do zkušebního vektoru lze obrátit. Výkon algoritmu se nezmění. [2]

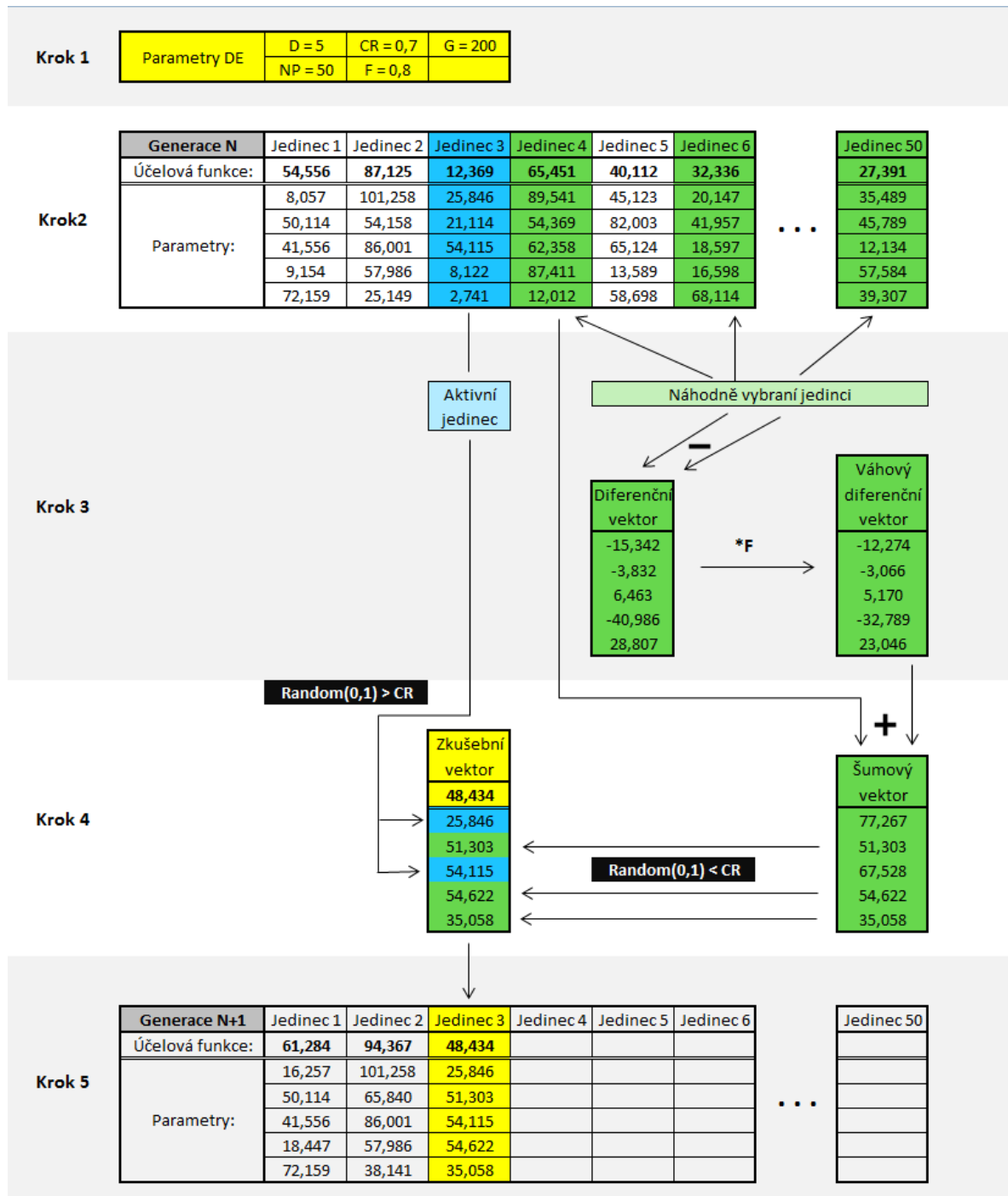
Výsledkem tohoto procesu je nový jedinec, který pak soutěží o zařazení do nové populace s aktivním jedincem.

### 3.4 Princip

Diferenciální evoluce probíhá v cyklech nazývaných generace. Každá generace prochází evolučním procesem a cílem je získat populaci jedinců, kteří jsou účelovou funkcí ohodnoceni jako co nejlepší. Algoritmus DE postupuje v těchto krocích:

1. Nejprve je třeba stanovit parametry určující chod diferenciální evoluce. Těmito parametry jsou v předchozí kapitole (3.1) popsány: dimenze, velikost populace, práh křížení, mutační konstanta a počet generací. Také je třeba nadefinovat vzorového jedince (tzv. specimen).
2. Podle vzorového jedince se vygeneruje populace a každý jedinec je ohodnocen účelovou funkcí.
3. Postupným výběrem všech jedinců v populaci je s každým z nich proveden evoluční cyklus, v němž je prováděna mutace a křížení. Takto vybraný prvek se nazývá aktivní jedinec. Poté se náhodně vyberou další 3 jedinci z populace, rozdíl dvou z nich vytvoří tzv. diferenční vektor a vynásobením s mutační konstantou  $F$  se získá tzv. váhový diferenční vektor. Přičtením k třetímu náhodně zvolenému jedinci získáme tzv. šumový vektor.
4. Následně se bere jeden prvek za druhým z šumového či aktivního vektoru (jedince) a generuje se náhodné číslo v rozsahu 0 až 1, které se porovnává s hodnotou  $CR$ . Pokud je hodnota náhodného čísla menší než  $CR$ , vezme se prvek z šumového vektoru, pokud je větší, vezme se prvek z aktivního jedince. Takto získané prvky spolu vytvoří tzv. zkušební vektor. Poté se porovná hodnota účelové funkce zkušebního vektoru a aktivního jedince. Lepší jedince postupuje do další generace.
5. Tímto je ukončena první generace. Ukončení celé DE nastane až v momentě, kdy je vykonán zadaný počet generací. Lze také vytvořit nějakou ukončovací podmínku - například pokud se hodnota nejlepšího jedince už několik generací nemění, není třeba čekat na konečný počet provedených generací, ale můžeme algoritmus ukončit, jelikož lze očekávat, že následný vývoj nejlepšího jedince už probíhat nebude.

Graficky je celý tento proces znázorněn na obrázku Obr.2.



Obr. 2. Příklad jednoho generačního cyklu diferenciální evoluce

### 3.5 Stagnace

Stagnace je jedna z nevýhod Diferenciální evoluce. Jedná se o jev, kdy se vývoj hodnoty účelové funkce zastaví a nepokračuje dále do globálního extrému., tzn. optimalizační proces dále nepokračuje. Stagnace může vzniknout bez zřejmých důvodů, obecně však vzniká za těchto příčin:

- a) Populace se dostala do lokálního extrému namísto globálního
- b) Populace ztratila diverzibilitu
- c) Optimalizace neprobíhá nebo probíhá velmi pomalu

Všechny tři body jsou vyjádřením téhož pohledu na věc. Jestliže populace je zavedena do lokálního extrému, neboli je "na jednom místě", znamená to, že parametry jedinců se od sebe takřka neliší a tudíž populace ztratila svou rozmanitost. Optimalizační proces díky tomu pak probíhá pomalu nebo vůbec neprobíhá. [2]

Zkoumáním však bylo zjištěno, že u diferenciální evoluce může za určitých podmínek dojít ke stagnaci i přesto, že nedošlo k ani jednomu z výše jmenovaných bodů.

## 4 TESTOVACÍ FUNKCE

Testování optimalizačních algoritmů je důležité pro ověření správné funkčnosti algoritmu, jeho robustnosti, pro porovnání jeho výkonnosti v závislosti na parametrech účelové funkce (počet lokálních extrémů, dimenze) a lze jím zjistit vhodné kombinace řídicích parametrů. Testování může probíhat jednak tím způsobem, že porovnáme výsledky s již existujícím řešením daného problému. Druhým způsobem je testování za pomoci testovacích funkcí, jejichž analytické vztahy známe a je tedy velmi jednoduché zjistit polohu a hodnotu extrému pro libovolnou dimenzi a porovnat ji s řešením nalezeným testovaným algoritmem.

Testovací funkce záměrně trpí různými vlastnostmi, jako jsou nelinearita, různé patologie v okolí extrému apod. Díky těmto vlastnostem můžeme zjistit, na jaký typ problému se daný algoritmus hodí či naopak nehodí.

### 4.1 Unimodální funkce

Unimodální funkce jsou takové, které na daném intervalu nabývají jen jednoho extrému. Slouží k testování převážně jednodušších problémů, jelikož zde algoritmus nemusí překonávat problémy spjaté s uvíznutím v lokálním extrému.

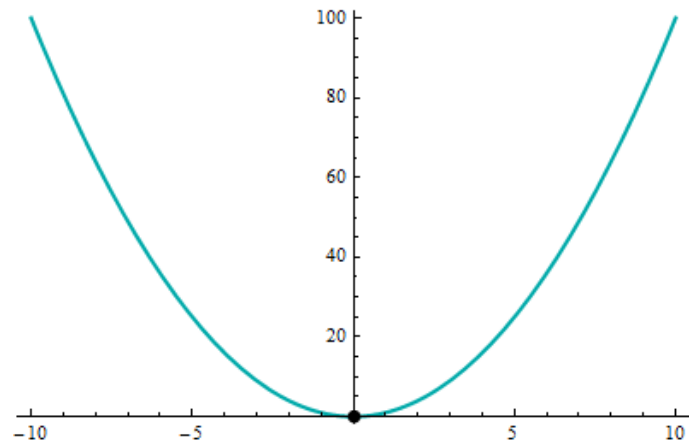
#### 4.1.1 První de Jongova funkce

Nejjednodušší testovací funkce je první de Jongova funkce. Je to spojitá, konvexní a unimodální funkce, která je popsána vztahem:

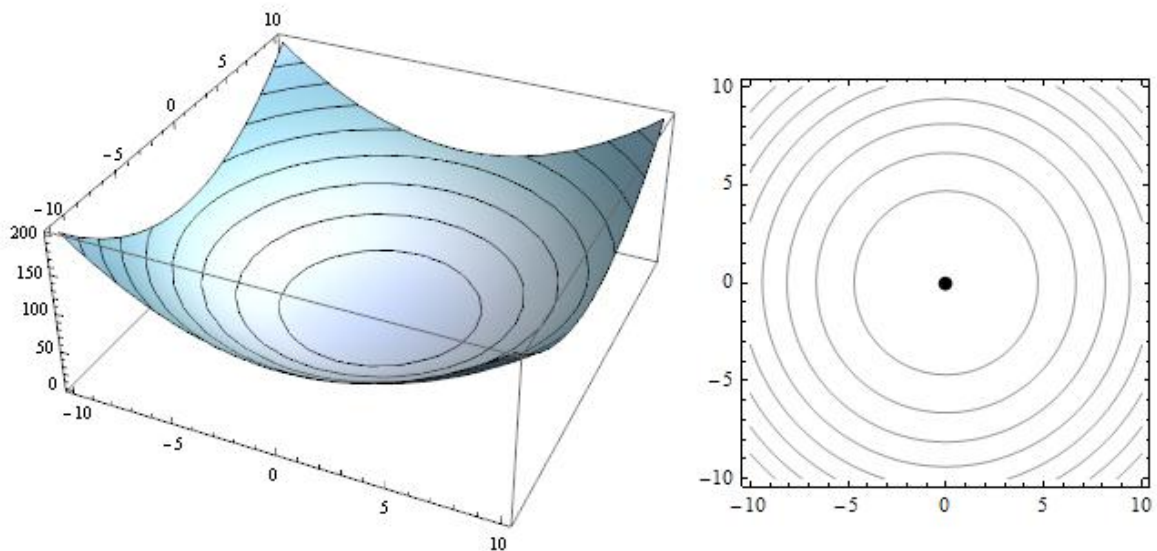
$$f(x) = \sum_{i=1}^D x_i^2 \quad (3)$$

Globální minimum je na pozici nula (v průsečíku os). Tuto testovací funkci lze použít pro  $N$  rozměrný prostor:

- Ve  $E_2$  má globální minimum na pozici  $(x_1, x_2) = (0, 0)$  a nabývá hodnoty  $y = 0$  (graficky znázorněno na obrázku Obr. 3.).
- V  $E_n$  má globální minimum na pozici  $(x_1, x_2, \dots, x_n) = (0, 0, \dots, 0)$  a nabývá hodnoty  $y = 0 \times n = 0$  (graficky znázorněno na obrázku Obr. 4.).



Obr. 3. První de Jongova funkce ve 2D s vyznačeným extrémem



Obr. 4. První de Jongova funkce s vyznačeným globálním minimem

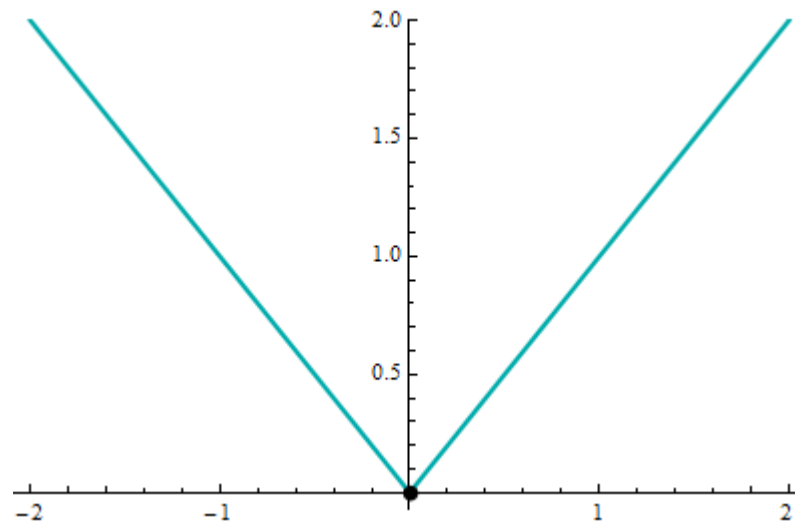
#### 4.1.2 Třetí de Jongova funkce

Třetí de Jongova funkce je jednoduchá testovací funkce absolutní hodnoty. Je vyjádřena vztahem:

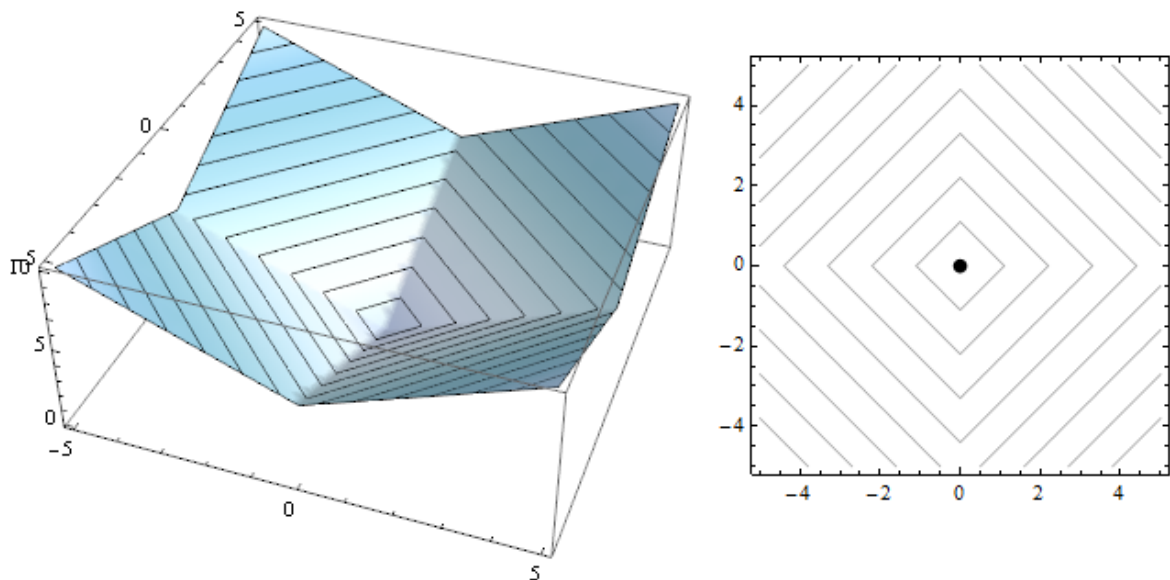
$$\sum_{i=1}^D |x_i| \quad (4)$$

Globální minimum se nachází:

- Ve  $E_2$  na pozici  $(x_1, x_2) = (0, 0)$  a nabývá hodnoty  $y = 0$  (graficky znázorněno na obrázku Obr. 5.).
- V  $E_n$  na pozici  $(x_1, x_2, \dots, x_n) = (0, 0, \dots, 0)$  a nabývá hodnoty  $y = 0 \times n = 0$



Obr. 5. Třetí de Jongova funkce ve 2D s vyznačeným extrémem



Obr. 6. Třetí de Jongova funkce s vyznačeným globálním minimem

## 4.2 Multimodální funkce

Multimodální testovací funkce mohou nabývat několika extrémů, které mohou mít různou hodnotu. Jedná se tedy o složitější testování, kdy algoritmus musí překonat problémy, které tvoří právě tyto různé lokální extrémy, k nimž může za jistých okolností konvergovat.

### 4.2.1 Rossenbrockovo sedlo (druhá de Jongova funkce)

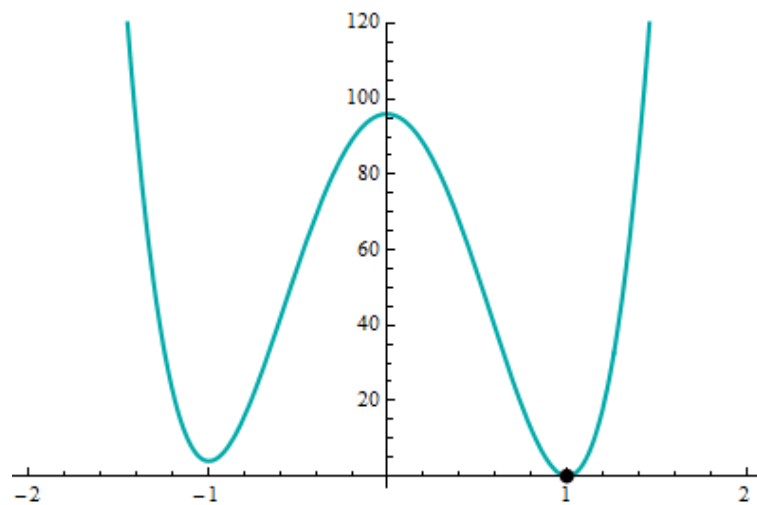
Rossenbrockovo sedlo, také známé jako *banánová funkce*, je druhou de Jongovou funkcí, která se nejčastěji používá k hodnocení výkonnosti algoritmu, jelikož globální extrém se nachází v dlouhém, úzkém sedle parabolického tvaru a jej obtížné ho najít.

Je dána rovnicí:

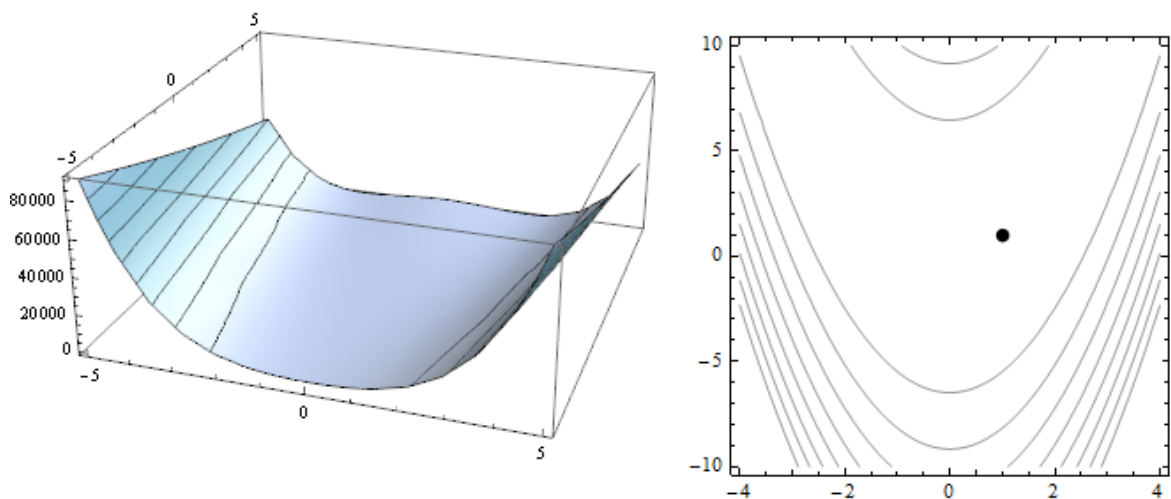
$$f(x) = \sum_{i=1}^{D-1} 100(x_i^2 - x_{i+1}) + (1 - x_i)^2 \quad (5)$$

Globální minimum se nachází:

- V  $E_2$  na pozici  $(x_1, x_2) = (1, 1)$  s hodnotou  $y = 0$ .
- V  $E_n$  na pozici  $(x_1, x_2, \dots, x_n) = (1, 1, \dots, 1)$  s hodnotou  $y = 0 \times n = 0$ .



Obr. 7. Rosenbrockovo sedlo ve 2D s vyznačeným extrémem



Obr. 8. Rosenbrockovo sedlo s vyznačeným globálním minimem

#### 4.2.2 Schwefelova funkce

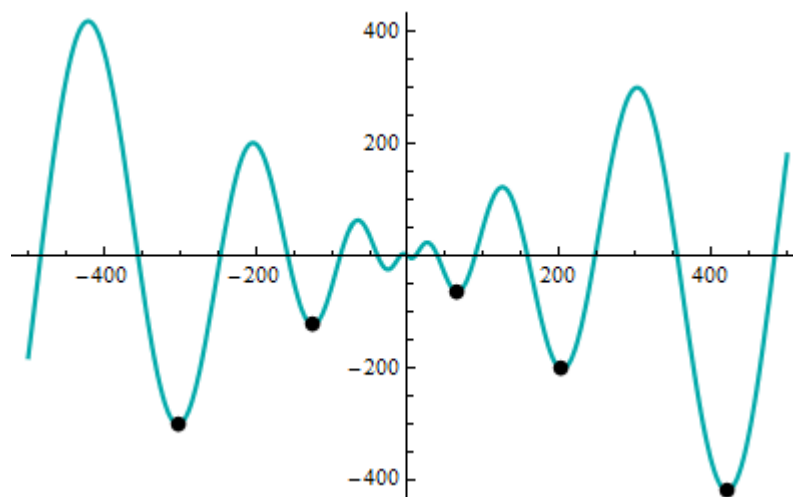
Schwefelova funkce je složitá testovací funkce. Její složitost je způsobena velkým množstvím lokálních extrémů, které se v případě, že se interval symetricky od počátku rozšiřuje, cyklicky střídají kolem počátku a testované algoritmy jsou potenciálně náchylné ke konvergenci ve špatném směru.

Schwefelova funkce je vyjádřena vztahem:

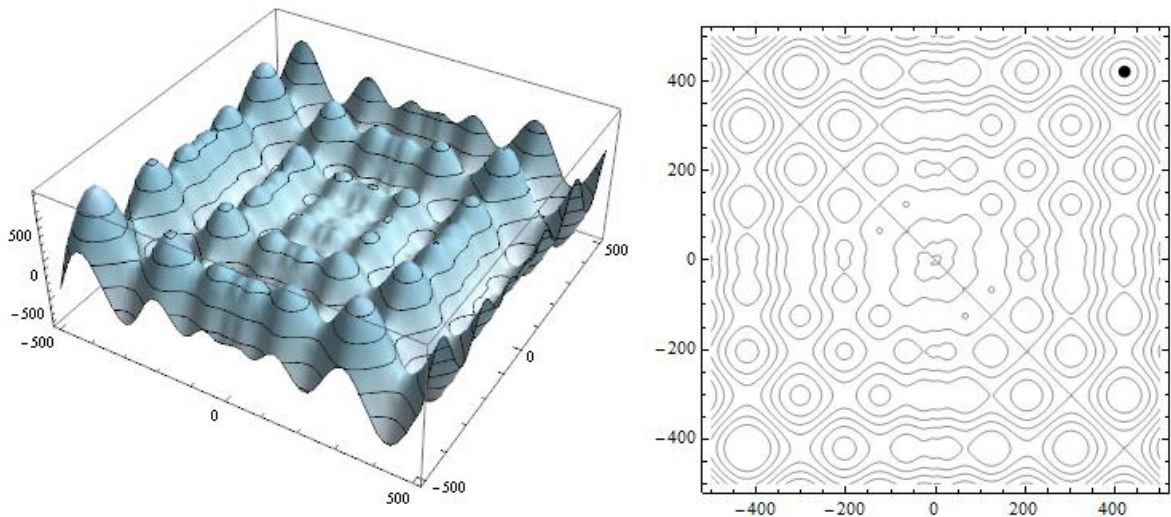
$$f(x) = \sum_{i=1}^D -x_i \cdot \sin(\sqrt{|x_i|}) \quad (6)$$

Globální minimum se nachází:

- V  $E_2$  na pozici  $(x_1, x_2) = (420,969, 420,969)$  s hodnotou  $y = -837,966$ .
- V  $E_n$  na pozici  $(x_1, x_2, \dots, x_n) = (420,969; 420,969; \dots; 420,969)$  s hodnotou  $y = -418,983 \times n$



Obr. 9. Schwefelova funkce ve 2D s vyznačenými cyklicky se střídajícími extrémů okolo počátku



Obr. 10. Schwefelova funkce s vyznačeným minimem

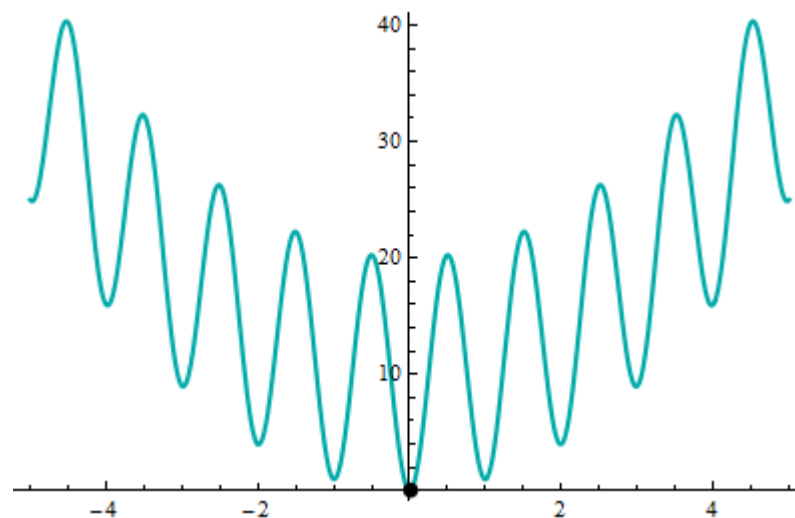
#### 4.2.3 Rastriginova funkce

Rastriginova funkce je vysoce multimodální členitá funkce s velkým množstvím extrémů. Je založena na první de Jongově funkci s přidáním kosínovou modulací. Je popsána vztahem:

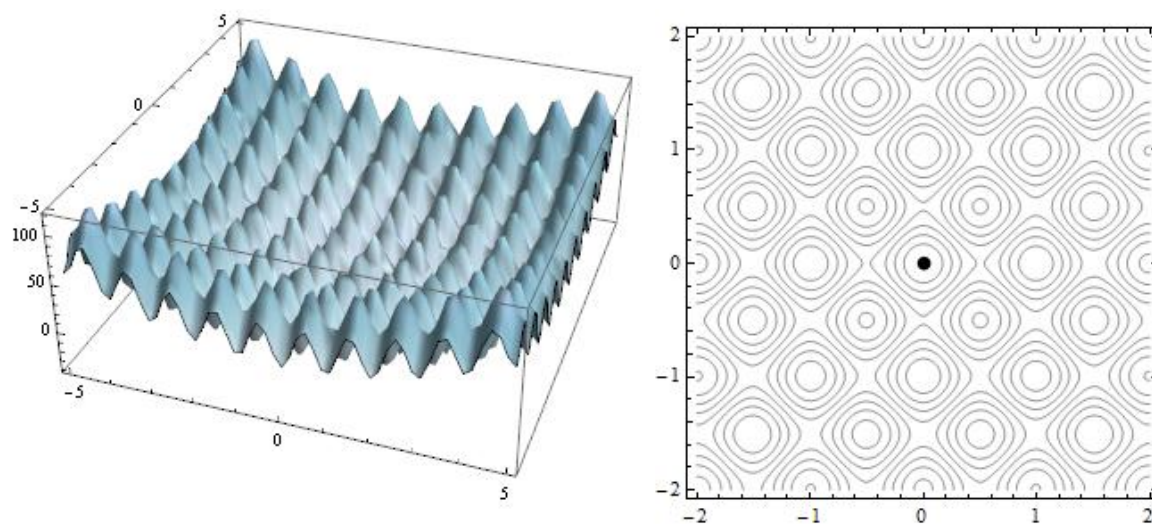
$$2D \sum_{i=1}^D x_i^2 - 10 \cos(2\pi x_i) \quad (7)$$

Globální minimum se nachází:

- V  $E_2$  na pozici  $(x_1, x_2) = (0, 0)$  s hodnotou  $y = -400$ .
- V  $E_n$  na pozici  $(x_1, x_2, \dots, x_n) = (0, 0, \dots, 0)$  s hodnotou  $y = -200 \times n$



Obr. 11. Rastriginova funkce ve 2D s vyznačeným extrémem



Obr. 12. Rastriginova funkce s vyznačeným globálním minimem

## 5 WOLFRAM MATHEMATICA

Software Mathematica je jedním z prostředí, jenž integruje nástroje pro numerickou a symbolickou matematiku, grafický a dokumentační systém a zajišťuje pokročilé propojení s dalšími aplikacemi. [9]

Mathematica je světově známá jako výkonná aplikace pro výpočty. Ale zvládá toho mnohem více - je jedinou vývojovou platformou, v níž se výpočty provádí integrované uvnitř kompletního pracovního postupu, který se pohybuje plynule od počátečních myšlenek až k individuálnímu použití nebo ke kompletním podnikovým řešením. [10]

### 5.1 Vznik

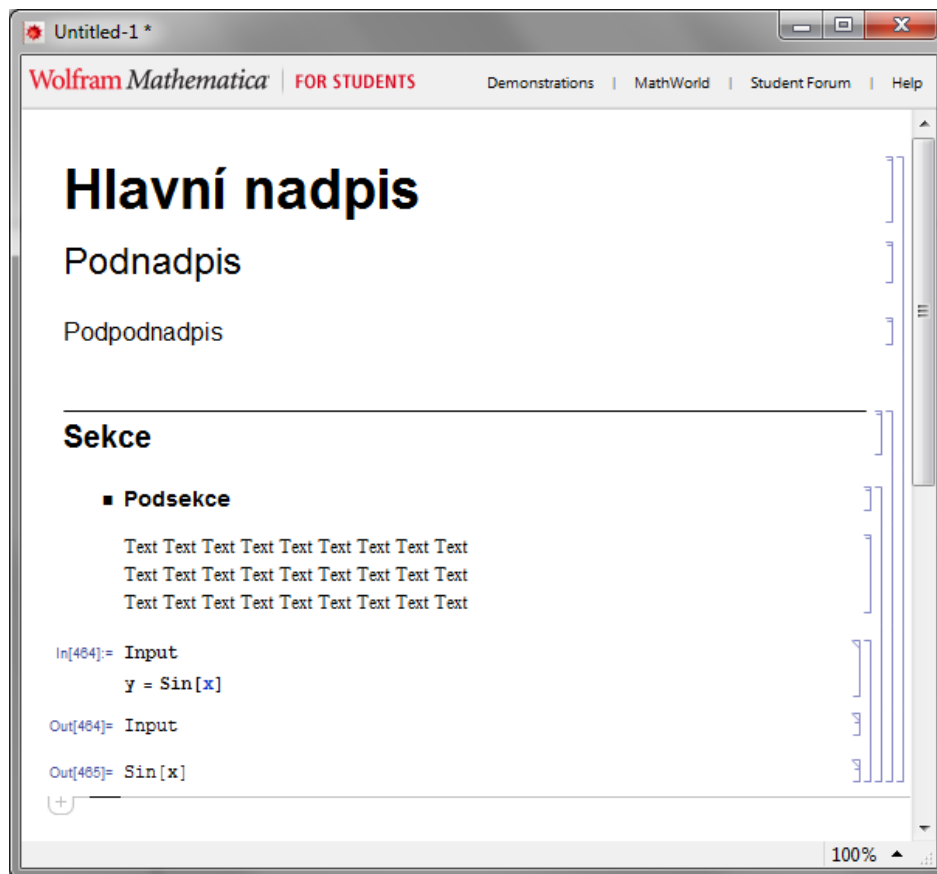
V roce 1987 byla založena Stephenem Wolframem společnost Wolfram Research, Inc., ve které byl a je tento produkt vytvářen. Již první vydání software Mathematica v roce 1988 mělo zásadní význam na způsob využívání počítačů v řadě technických i jiných odvětví. Říká se, že právě vydáním software Mathematica začal nový věk tzv. technical computing. Od 60tých let minulého století existovaly samostatné sady, ale byly vhodné vždy jen pro určité specifické úlohy. Např. sady pro úlohy numerické, algebraické, grafické a jiné. A právě software Mathematica je převratný v tom, že integruje všechny potřebné sady v jednom produktu. [9]

V současné době je nejnovější verze Mathematica 9, která vyšla v listopadu 2012 a je dostupná pro platformy Microsoft Windows, MacOS X a Linux. Poslední verze s sebou přinesla opět mnohá vylepšení oproti předchozím verzím, kterými jsou například kontextový asistent (tzv. našeptávač), který v předchozích verzích chyběl, plnou podporu pro webový přístup, nový interaktivní asistent pro práci s obrázky či nové ovládací a zobrazovací prvky.

### 5.2 Vývojové prostředí

Hlavní částí programu je tzv. notebook, do kterého se zapisují všechny výrazy. Notebook je složen z buněk, které lze libovolně formátovat (stylovat) co se funkce i vzhledu týče. Standardní formátování je typu Input, do kterého se zapisují vstupní výrazy. Každý výraz v buňce se spustí stiskem kláves Shift + Enter nebo klávesou Enter na numerické klávesnici. Poté se objeví nová buňka, tentokrát již typu Output, která obsahuje výsledek provedení výpočtu. Další styly mohou být různé nadpisy, podnápisy, sekce, texty, odrážky, kódy

apod. (viz Obr.13.). Výhodou tohoto rozdělení do sekcí je kromě přehlednosti především možnost rozdělit kód a spustit každou sekci zvlášť.



Obr. 13. Struktura notebooku v Mathematica 8

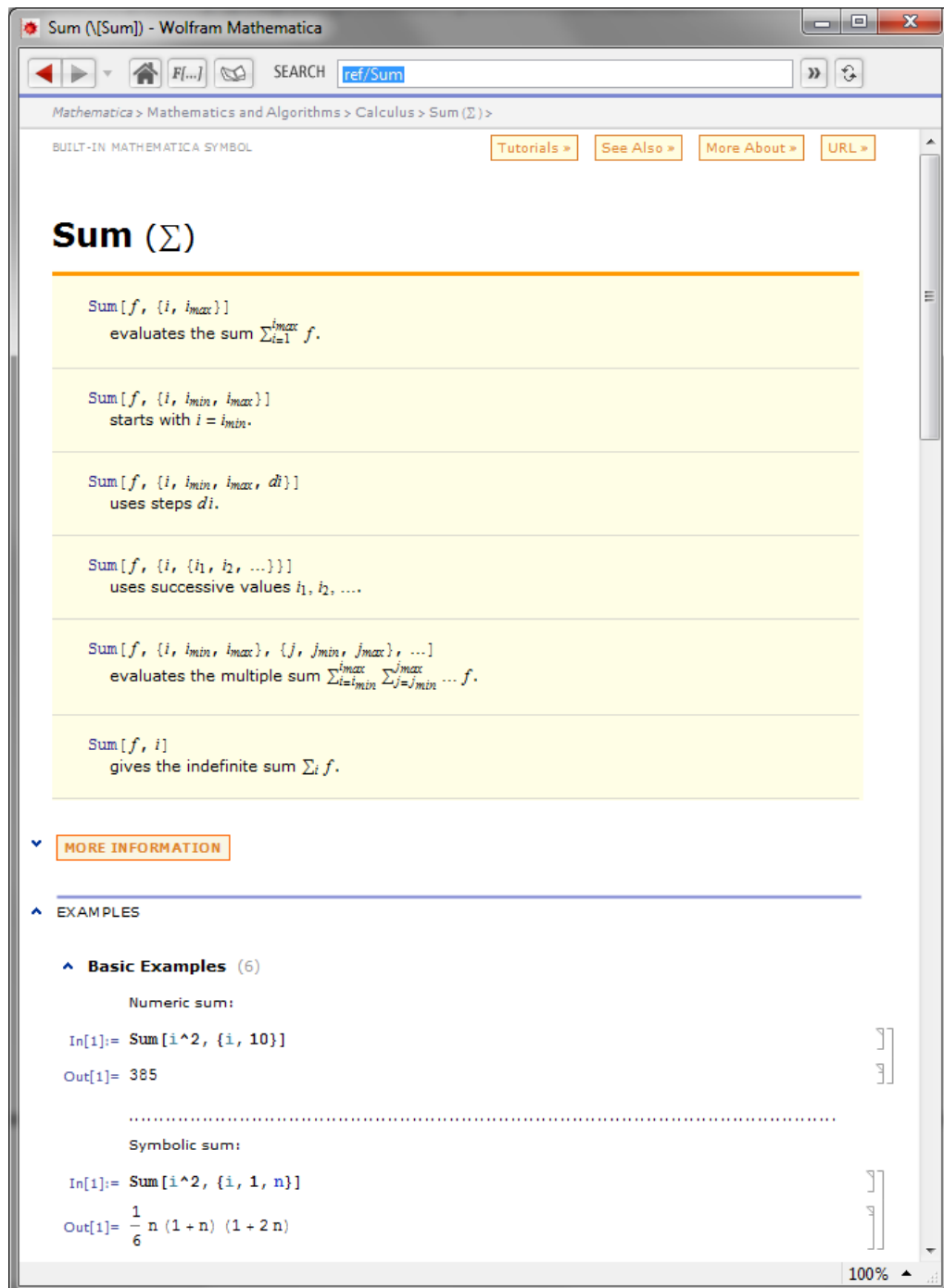
### 5.3 Interaktivní nápověda

Velmi důležitou součástí programu Mathematica je výborně propracovaná nápověda. Ta obsahuje velké množství příkladů a ukázek jednotlivých funkcí, jejich parametrů a různých nastavení. U každé funkce je možnost zobrazit si veškeré její atributy, včetně názorných ukázek, které lze libovolně upravovat a spouštět přímo v prostředí nápovědy.

Některé z funkcí mohou mít počet vstupních parametrů různý v závislosti na požadovaném použití (viz Obr.14.). Tím lze jednu funkci použít na různá řešení, kdy chování funkce a její výsledná hodnota bude ovlivněna množstvím vstupních parametrů.

Nápověda v software Mathematica zahrnuje kompletní dokumentaci pro všechny funkce v software Mathematica a úplný text The Mathematica Book jako plně indexovaný notebook s pokročilými vyhledávacími schopnostmi včetně hypertextových odkazů. Nápověda také obsahuje interaktivní příklady, které demonstrují použití funkcí, jejich

hlavní schopnosti a nejlepší způsob, jak jich využít. Na rozdíl od jiných software, Mathematica umožňuje upravovat a vyhodnocovat příklady uvedené v nápovědě. Uživatelské změny nejsou trvalé. V případě vymazání části textu či příkladu je pouze potřeba znovu zavést tuto stranu a tím obnovit originální informace. [9]



Obr. 14. Nápověda pro funkci Sum a její možné vstupní parametry

## **II. PRAKTICKÁ ČÁST**

## 6 ALGORITMUS DIFERENCIÁLNÍ EVOLUCE

Algoritmus diferenciální evoluce pro práci s diskrétními hodnotami byl vytvořen v prostředí Mathematica 8. Zvolená strategie byla použita DE/rand/1/bin. Pro výpočet hodnoty účelové funkce byla vytvořena speciální dvou rozměrná funkce, pracující s tabulkou souřadnic pro jednotlivé body. Tyto souřadnice jsou umístěny v textovém souboru a importovány do notebooku softwaru Mathematica.

Hodnoty mutační konstanty  $F$  a prahu křížení  $CR$  byly zvoleny konstantní pro celý průběh testování. Cílem této práce není otestovat vliv těchto dvou hodnot na výsledek diferenciální evoluce, nýbrž hlavním cílem je otestovat vliv hodnoty *Repairment* (viz kapitola 6.1) na průběh evolučního vývoje nejlepšího jedince.

Dalšími proměnnými parametry jsou velikost populace  $NP$  a počet generací  $Gen$ , jelikož je zřejmé, že při malém počtu generací, případně při malé populaci, nedojde diferenciální evoluce do přijatelného výsledku.

Každý průběh konečného počtu generací je také 30x opakován. Tím jsou získané výsledky statisticky platné a lze z nich určit průměrnou hodnotu, medián či směrodatnou odchylku všech opakování.

Všechny nastavené vstupní parametry lze vidět v následující tabulce:

Tab. 1. Nastavení parametrů pro testování

Parametr	Označení	Hodnota
Velikost dimenze	<i>Dim</i>	5 až 200
Počet generací	<i>Gen</i>	500 až 5000
Velikost populace	<i>NP</i>	30 až 75
Práh křížení	<i>CR</i>	0,5
Mutační konstanta	<i>F</i>	0,8
Oprava	<i>Repairment</i>	0,1 až 0,9
Počet opakování	<i>Repeat</i>	30

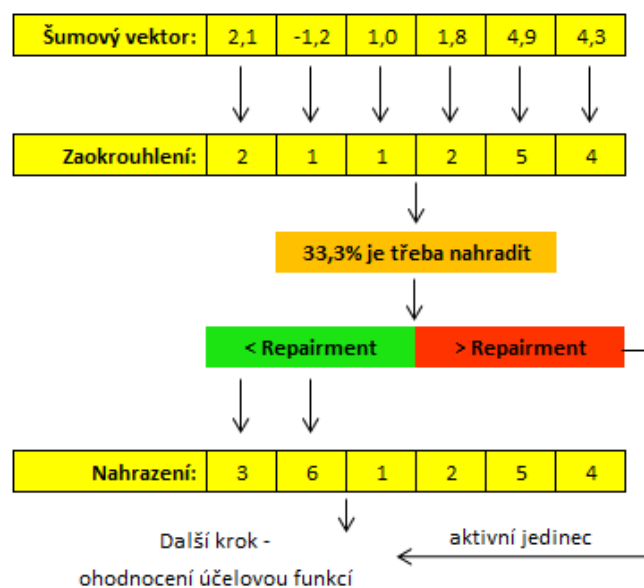
### 6.1 Repairment

Při použití algoritmu diferenciální evoluce na řešení problému obchodního cestujícího, je třeba upravit vliv mutace a křížení na výsledné parametry jedince. Jak jsem již popsal v teoretické části (kapitola 1.1.2), obchodní cestující musí projít všemi zadanými městy a

každým z nich právě jednou, přičemž na konci se musí vrátit zpět do výchozího bodu. Při násobení rozdílu dvou parametrů podle strategie DE/rand/1/bin, kdy hodnota mutační konstanty  $F$  nabývá hodnot v intervalu od 0 do 2, je zřejmé, že vynásobením byt celočíselných parametrů a následným přičtením k parametru třetího jedince, nezískáme vždy celočíselnou hodnotu, někdy i v záporných hodnotách. Je tedy třeba tuto hodnotu upravit, například absolutní hodnotou a zaokrouhlením na nejbližší celé číslo.

Zde ale nastává další problém. Zaokrouhlením můžeme získat číslo města, které už se v rámci dimenze jedince jednou (případně vícekrát) vyskytuje. Může se také vyskytnout nula nebo číslo větší než dimenze. Tento problém jsem vyřešil změnou záporných a nulových hodnot na nejnižší možnou hodnotu v dimenzi, teda na hodnotu 1. Hodnoty vyšší než rozsah dimenze jsou pak změněny na nejvyšší hodnotu rozsahu dimenze. Čísla, která se u jednoho jedince vyskytují vícekrát, jsou pak za pomoci funkcí *MemberQ*, *Union* a generátoru náhodných celých čísel změněna na požadované hodnoty.

Všechny tyto změny parametrů jedince jsou zaznamenány a je určen počet nahrazených prvků. Například jedinec na Obr.15. s dimenzí  $D = 6$  má dva z šesti prvků shodné, procentuálně se jedná o 33,3%. Pokud je tato hranice nižší než hodnota parametru *Repairment*, jsou tyto prvky nahrazeny, konkrétně číslo 2 a číslo 1 za hodnoty 3 a 6 a takto upravený jedinec je vpuštěn do dalšího kroku. Pokud je hranice *Repairment* překročena, jedinec není do dalších kroků vpuštěn a na jeho místo se dostává původní aktivní jedinec.



Obr. 15. Grafické znázornění průběhu oprav

## 7 OBCHODNÍ CESTUJÍCÍ ŘEŠENÝ HRUBOU SILOU A DIFERENCIÁLNÍ EVOLUCÍ

Problém obchodního cestujícího je nedeterministicky polynomiální těžký (NP-hard) problém, jeho řešení hrubou silou je tedy možné jen pro nízký počet měst. Pokud máme  $n$  měst, je třeba prohledat  $n!$  možných permutací a vybrat z nich nejkratší možnou cestu. V tabulce Tab. 2. lze vidět, jak stoupá počet všech permutací s nárůstem o 1 město.

Tab. 2. Počet permutací pro 1 až 15 měst

Počet měst $n$	Počet všech permutací
1	1
2	2
3	6
4	24
5	120
6	720
7	5 040
8	40 320
9	362 880
10	3 628 800
11	39 916 800
12	479 001 600
13	6 227 020 800
14	87 178 291 200
15	1 3087 674 368 000

### 7.1 Porovnání nalezených řešení

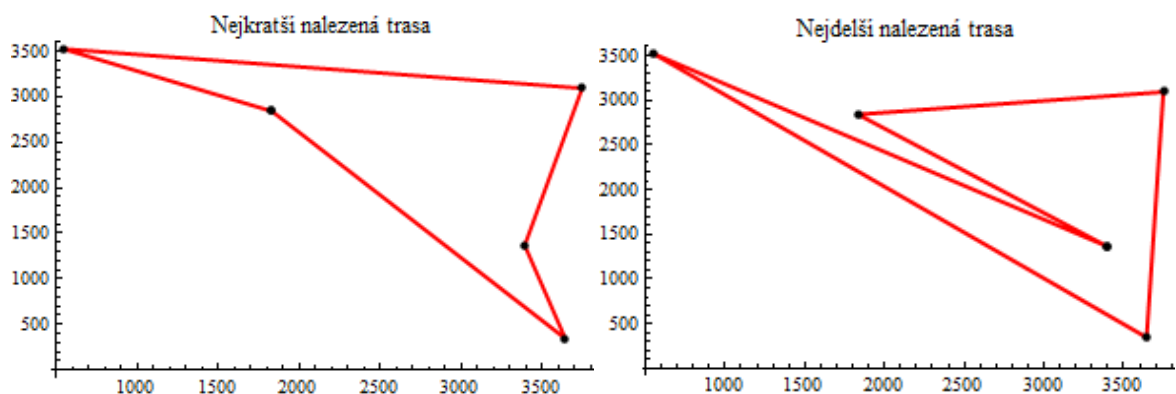
Počet generací a velikost populace u diferenciální evoluce byly pro testování takto malého počtu měst sníženy na  $Gen = 50$  a  $NP = 10$ . Tyto hodnoty jsou dostačující k nalezení minima (maxima) takto malé dimenze a vyšší počet by jen zvyšoval čas potřebný k výpočtu. Hodnota *Repairment* byla nastavena na 30%, aby byla z větší části zachována evoluce. Každé řešení proběhlo celkem 30 krát, kvůli statistické platnosti.

## 7.1.1 Pro 5 měst

Tab. 3. Porovnání nalezených výsledků

	Řešení hrubou silou	Diferenciální evoluce
Nejkratší trasa	10584	10584
Pořadí měst	1, 3, 2, 5, 4	4, 5, 2, 3, 1
Nejdelší trasa	14837	14837
Pořadí měst	1, 2, 4, 3, 5	3, 4, 2, 1, 5
Průměrný čas výpočtu	0,037s	0,185s

Z tabulky Tab. 3. lze vyčíst, že nejkratší i nejdelší nalezená trasa je totožná pro obě řešení a je zobrazena na Obr. 16. Pořadí měst nemusí být zachováno přesně. Obchodní cestující nemusí začínat ve stejném místě a nemusí také procházet stejným směrem, tedy čísla měst mohou jít pozpátku, jak je tomu v tomto případě. Čas výpočtu je v tomto případě rychlejší u řešení hrubou silou (čas u DE lze však ještě snížit počtem generací a velikostí populace).



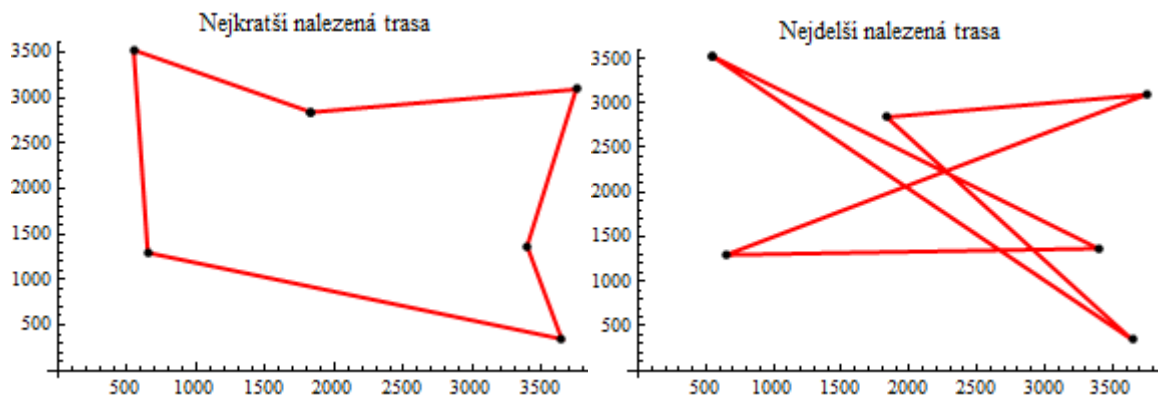
Obr. 16. Nejkratší a nejdelší nalezená trasa pro 5 měst

## 7.1.2 Pro 6 měst

Tab. 4. Porovnání nalezených výsledků

	Řešení hrubou silou	Diferenciální evoluce
Nejkratší trasa	11576	11576
Pořadí měst	1, 3, 2, 4, 5, 6	2, 3, 1, 6, 5, 4
Nejdelší trasa	19354	19354
Pořadí měst	1, 4, 2, 6, 3, 5	5, 3, 6, 2, 4, 1
Průměrný čas výpočtu	0,260s	0,186s

V tomto případě už byla diferenciální evoluce rychlejší. Nejkratší nalezená trasa i pořadí měst jsou shodné pro obě řešení.



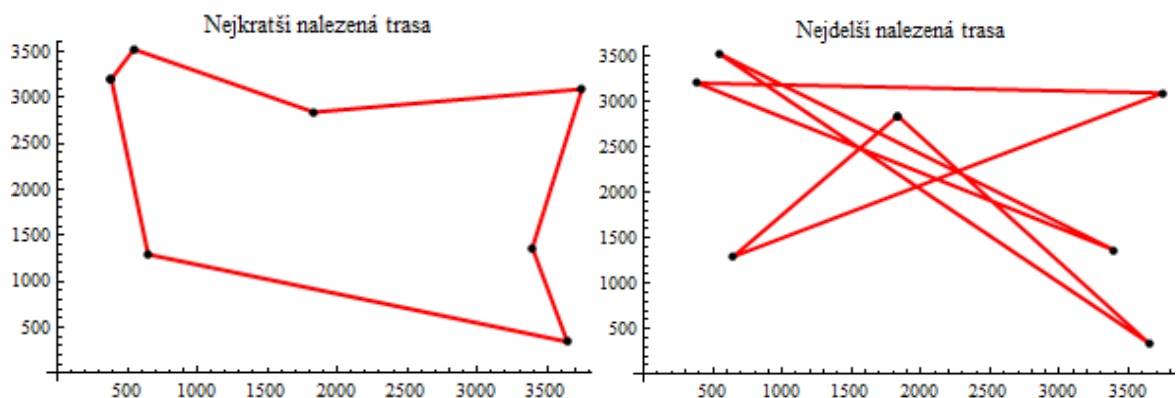
Obr. 17. Nejkratší a nejdelší nalezená trasa pro 6 měst

### 7.1.3 Pro 7 měst

Tab. 5. Porovnání nalezených výsledků

	Řešení hrubou silou	Diferenciální evoluce
Nejkratší trasa	11630	11630
Pořadí měst	2, 4, 5, 7, 6, 1, 3	4, 5, 7, 6, 1, 3, 2
Nejdelší trasa	23513	23513
Pořadí měst	1, 4, 6, 2, 7, 3, 5	4, 1, 5, 3, 7, 2, 6
Průměrný čas výpočtu	2,757s	0,230s

Diferenciální evoluce byla opět rychlejší, v tomto případě již několikanásobně. Nalezená trasa i pořadí měst je shodné pro obě řešení.



Obr. 18. Nejkratší a nejdelší nalezená trasa pro 7 měst

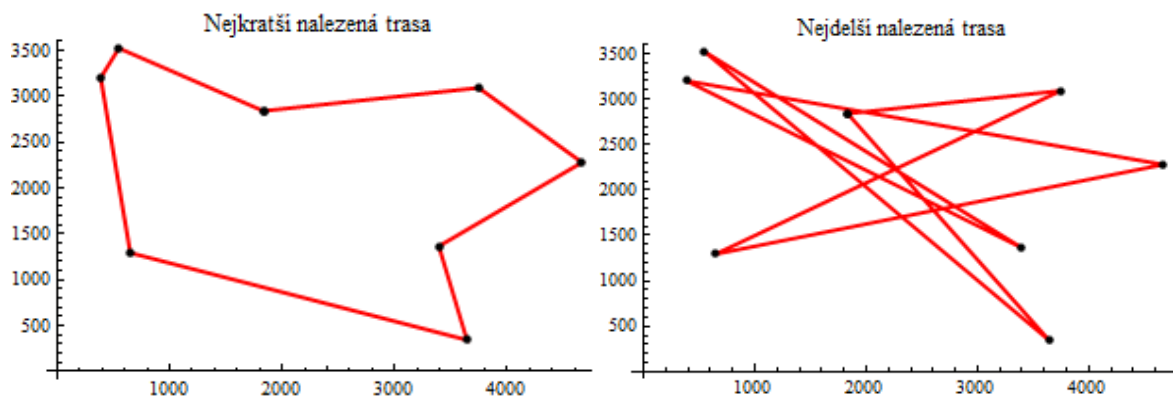
### 7.1.4 Pro 8 měst

Pro tento a následující případy zvýšen počet generací u diferenciální evoluce na  $Gen = 100$  (algoritmus nestíhal dojít do minima ve většině opakování). Tím pádem bude potřeba o něco více času pro nalezení optimálního řešení.

Tab. 6. Porovnání nalezených výsledků

	Řešení hrubou silou	Diferenciální evoluce
Nejkratší trasa	12622	12662
Pořadí měst	1, 3, 8, 2, 4, 5, 7, 6	2, 4, 5, 7, 6, 1, 3, 8
Nejdelší trasa	28650	28650
Pořadí měst	1, 4, 2, 6, 8, 7, 3, 5	4, 2, 6, 8, 7, 3, 5, 1
Průměrný čas výpočtu	128,518s	0,438s

I přes zvýšení počtu generací je diferenciální evoluce mnohem rychlejší a dosáhla shodného řešení jako útok hrubou silou.



Obr. 19. Nejkratší a nejdelší nalezená trasa pro 8 měst

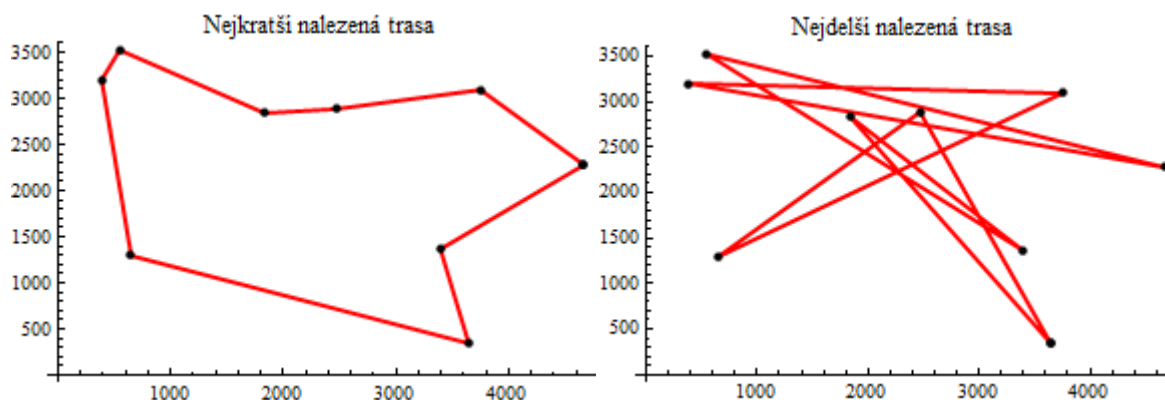
### 7.1.5 Pro 9 měst

Tab. 7. Porovnání nalezených výsledků

	Řešení hrubou silou	Diferenciální evoluce
Nejkratší trasa	-	12664
Pořadí měst	-	8, 2, 9, 4, 5, 7, 6, 1, 3
Nejdelší trasa	-	29650
Pořadí měst	-	1, 4, 3, 5, 8, 7, 2, 6, 9
Průměrný čas výpočtu	-	0,435s

Problém obchodního cestujícího pomocí řešení hrubou silou nebylo možné vyřešit, neboť na testované sestavě nebyl dostatek paměti k provedení  $9!$  všech možných permutací a jejich následnému porovnání.

Teoreticky by byl čas potřebný k nalezení řešení pomocí útoku hrubou silou pro 9 měst minimálně 9 krát vyšší než pro 8 měst a bylo by třeba porovnat celkem 360 800 všech možných devítimístných permutací.



Obr. 20. Nejkratší a nejdelší nalezená trasa pro 9 měst pomocí diferenciální evoluce

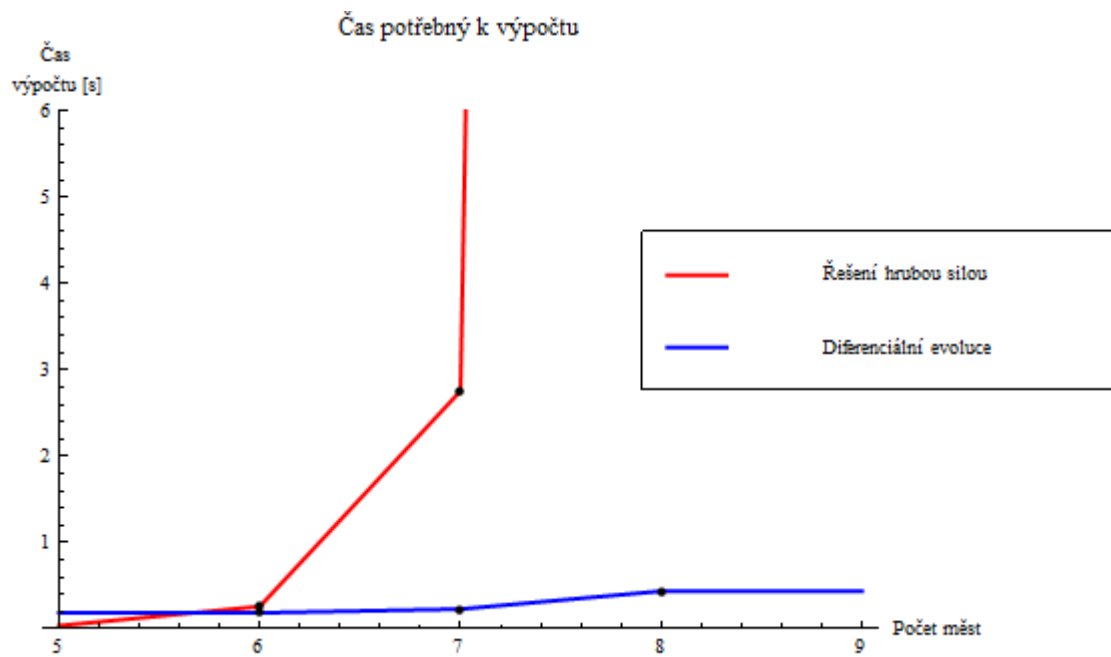
## 7.2 Srovnání

Řešením problému obchodního cestujícího hrubou silou i pomocí diferenciální evoluce pro 5 až 9 měst bylo dosaženo shodných výsledků, a to jak pro hledání nejkratší trasy (minima), tak pro hledání nejdelší možné trasy (maxima). Taktéž pořadí měst bylo shodné, případně jen se změnou směru nebo výchozího bodu.

Z tabulky Tab.8. a obrázku Obr. 21. je zřejmé, jak rychle narůstá čas potřebný k nalezení řešení problému obchodního cestujícího pomocí útoku hrubou silou. Potřebný čas u diferenciální evoluce narůstá mnohem pomaleji. Pomocí diferenciální evoluce tedy můžeme v reálném čase nalézt řešení pro mnohem větší počet měst.

Tab. 8. Doba trvání algoritmu

Počet měst	Útok hrubou silou	Diferenciální evoluce
5	0,037s	0,185s
6	0,260s	0,180s
7	2,757s	0,230s
8	128,518s	0,438s
9	N/A	0,435s



Obr. 21. Srovnání času výpočtu pro řešení hrubou silou a diferenciální evoluci

## 8 TESTOVÁNÍ ZÁVISLOSTI NA REPAIRMENT

Hranice *Repairment* v rozmezí 30% až 50% by měla zaručit spolehlivou funkčnost algoritmu. Nižší hranice způsobí to, že do dalších generací budou vpouštěni jedinci z předchozích generací, neprojeví se tedy evoluce. Naopak vysoká hranice bude pouštět do dalších generací náhodně vytvořené jedince a evoluce se zde také projeví jen v malé míře.

### 8.1 Dimenze 20, generace 500 a velikost populace 30

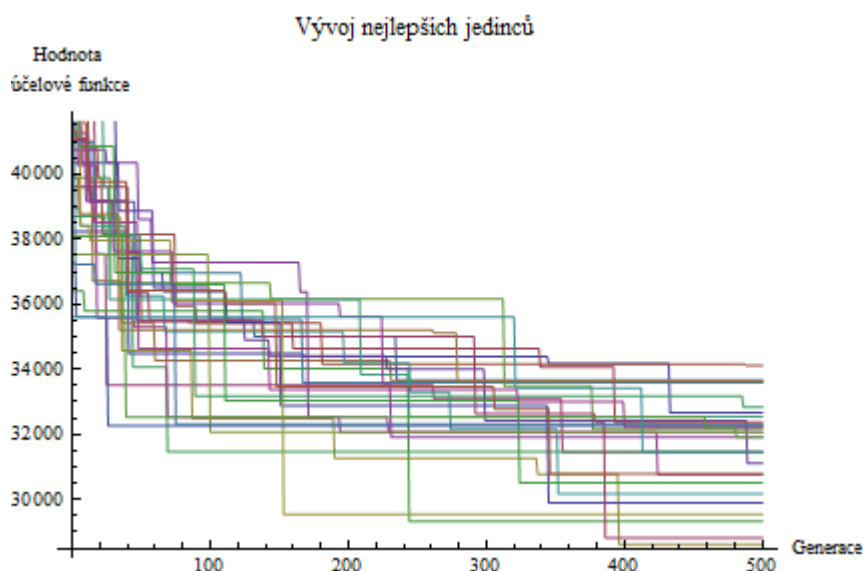
Počet měst je 20, řešení hrubou silou by vyžadovalo otestovat a následně mezi sebou porovnat  $20! = 2,432 \cdot 10^{18}$  permutací, což je prakticky neřešitelný úkol.

#### 8.1.1 Repairment 0,3

*Repairment* 0,3 znamená, že evoluce je zde zachování ze 70%, zbylá část je věci náhodného řešení.

Tab. 9. Nastavené parametry algoritmu pro malou dimenzi a nízký počet generací

Parametr	Hodnota	Parametr	Hodnota	Parametr	Hodnota	Parametr	Hodnota
<b><i>Repairment</i></b>	<b>0,3</b>	<i>Dim</i>	20	<i>F</i>	0,8	<i>Gen</i>	500
		<i>NP</i>	30	<i>CR</i>	0,5	<i>Repeat</i>	30



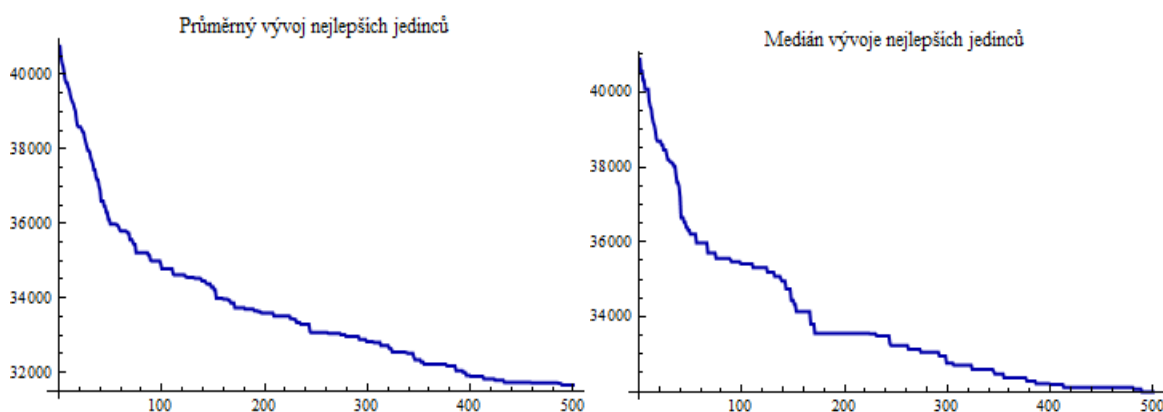
Obr. 22. Nejlepší jedinci

Z grafu na Obr. 22. je vidět, že hodnoty nejlepších jedinců po 500 generacích jsou pro jednotlivá opakování velmi odlišné. Lze usoudit, že algoritmus neměl k dispozici potřebný

počet generací nebo dostatečně velkou populaci k nalezení lepšího řešení. Hodnoty získané z měření jsou uvedeny v tabulce Tab. 10.

Tab. 10. Hodnoty získané z řešení

Nejlepší jedinec (hodnota účelové funkce)	<b>28616</b>
Parametry nejlepšího jedince	13, 6, 19, 7, 5, 15, 4, 9, 8, 17, 18, 14, 16, 11, 2, 10, 12, 20, 3, 1
Průměr nejlepších nalezených jedinců pro 30 opakování	31662
Medián nejlepších nalezených jedinců pro 30 opakování	31988
Směrodatná odchylka nejlepších jedinců	1471
Průměrný čas 1 opakování	10,032s
Celkový čas testu	300,954s



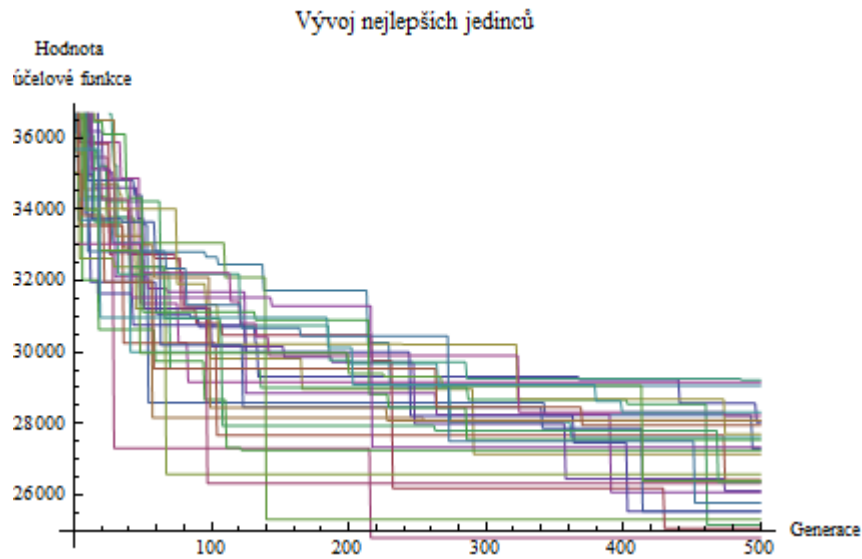
Obr. 23. Průměr a medián nejlepších nalezených jedinců

### 8.1.2 Repairment 0,5

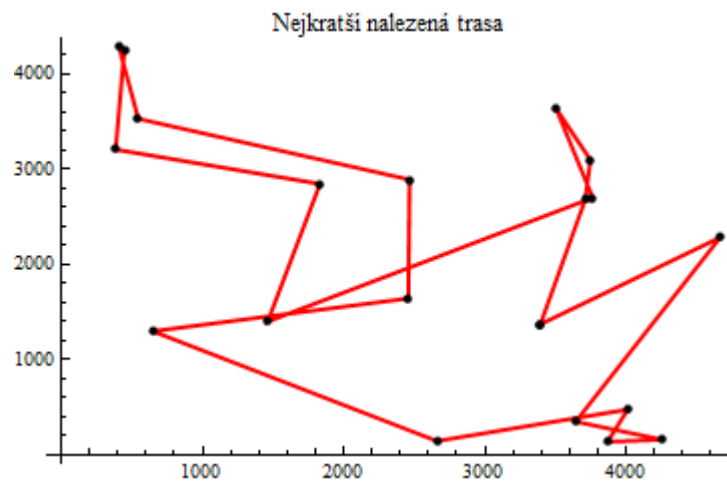
Pro *Repairment* 0,5 je z 50% zachována evoluce a zbylých 50% je věcí náhody.

Tab. 11. Nastavené parametry algoritmu pro malou dimenzi a nízký počet generací

Parametr	Hodnota	Parametr	Hodnota	Parametr	Hodnota	Parametr	Hodnota
<b><i>Repairment</i></b>	<b>0,5</b>	<i>Dim</i>	20	<i>F</i>	0,8	<i>Gen</i>	500
		<i>NP</i>	30	<i>CR</i>	0,5	<i>Repeat</i>	30



Obr. 24. Nejlepší jedinci

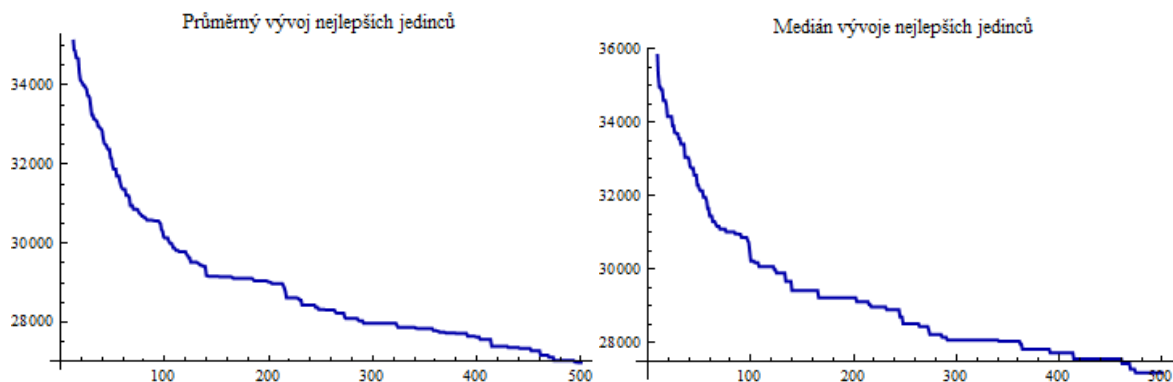


Obr. 25. Nejkratší nalezená trasa

Tab. 12. Hodnoty získané z řešení

Nejlepší jedinec (hodnota účelové funkce)	<b>24794</b>
Parametry nejlepšího jedince	3, 14, 2, 11, 16, 18, 4, 7, 19, 15, 5, 9, 17, 6, 10, 20, 13, 12, 1, 8
Průměr nejlepších nalezených jedinců pro 30 opakování	26978
Medián nejlepších nalezených jedinců pro 30 opakování	27192
Směrodatná odchylka nejlepších jedinců	1271
Průměrný čas 1 opakování	17,666s
Celkový čas testu	529,968s

Z obrázku Obr. 25. je zřejmé, že nejkratší nalezená trasa je nepřesná a nejspíše ještě daleko od skutečnosti (což však není možné žádným deterministickým způsobem ověřit). Nejkratší trasa ale většinou bývá po "obvodu" všech souřadnic. Špatné nalezené řešení je způsobeno především malým počtem generací, ale také malou populací. Pro nalezení optimálnějšího řešení je třeba tyto hodnoty zvětšit.



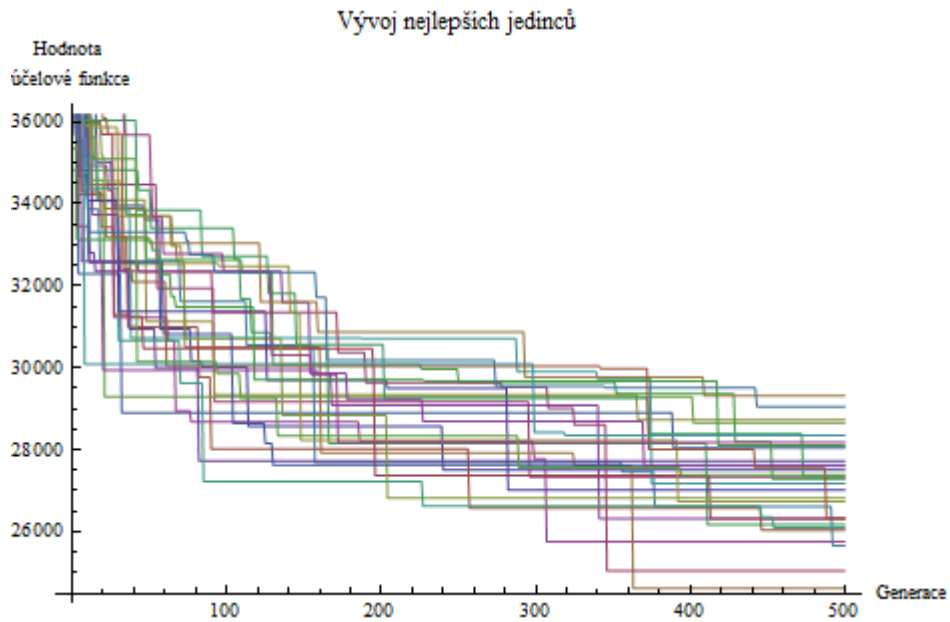
Obr. 26. Průměr a medián nejlepších nalezených jedinců

### 8.1.3 Repairment 0,7

Hodnota *Repairment* nastavená na 0,7 má za příčinu to, že 70% řešení bude ovlivněno náhodným generováním parametrů jedince. Pouze z 30% bude zachován evoluční vývoj jedinců.

Tab. 13. Nastavené parametry algoritmu pro malou dimenzi a nízký počet generací

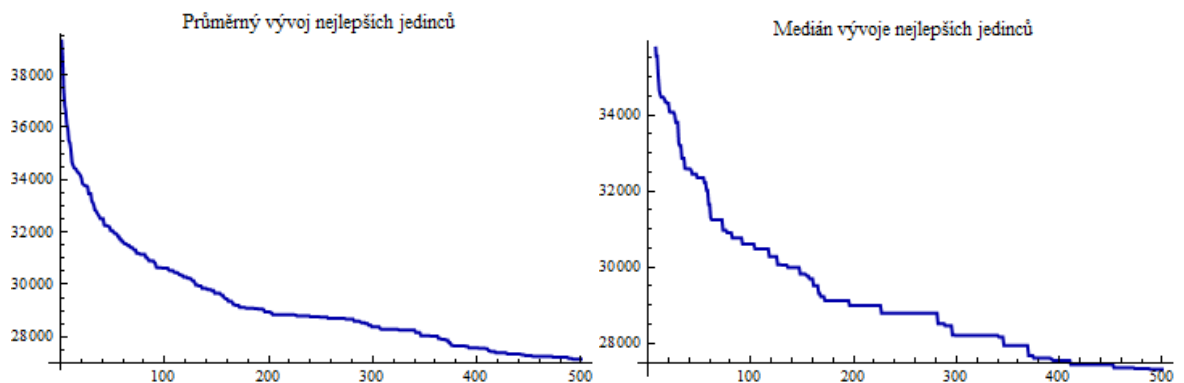
Parametr	Hodnota	Parametr	Hodnota	Parametr	Hodnota	Parametr	Hodnota
<b><i>Repairment</i></b>	<b>0,7</b>	<i>Dim</i>	20	<i>F</i>	0,8	<i>Gen</i>	500
		<i>NP</i>	30	<i>CR</i>	0,5	<i>Repeat</i>	30



Obr. 27. Nejlepší jedinci

Tab. 14. Hodnoty získané z řešení

Nejlepší jedinec (hodnota účelové funkce)	<b>24618</b>
Parametry nejlepšího jedince	14, 17, 9, 11, 2, 8, 16, 4, 19, 15, 7, 5, 18, 6, 3, 10, 13, 20, 12, 1
Průměr nejlepších nalezených jedinců pro 30 opakování	27144
Medián nejlepších nalezených jedinců pro 30 opakování	27312
Směrodatná odchylka nejlepších jedinců	1159
Průměrný čas 1 opakování	18,360s
Celkový čas testu	550,813s

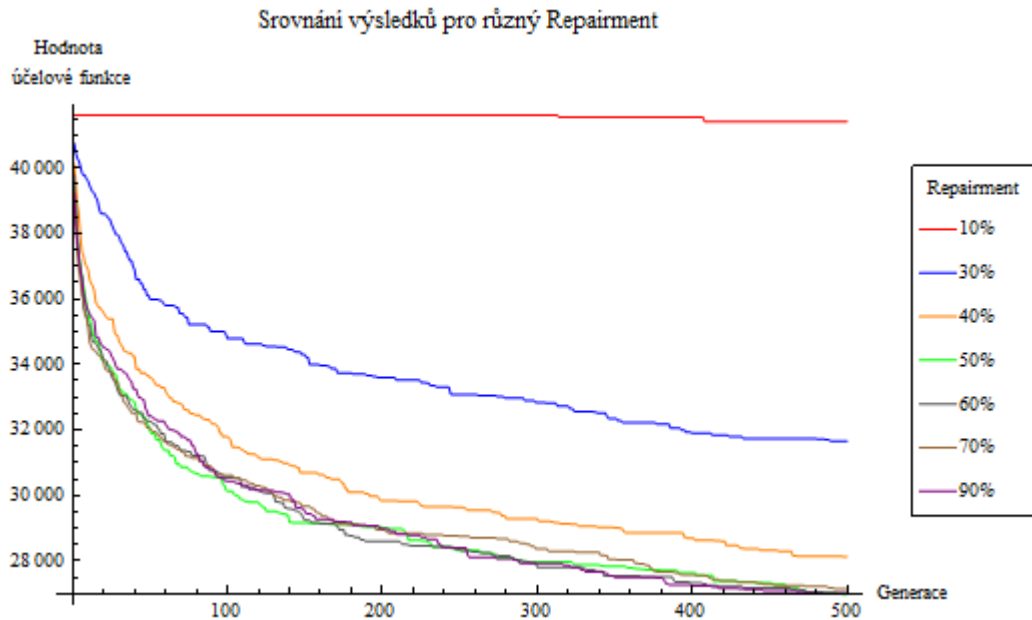


Obr. 28. Průměr a medián nejlepších nalezených jedinců

### 8.1.4 Porovnání průměrů a mediánů pro různou hranici oprav

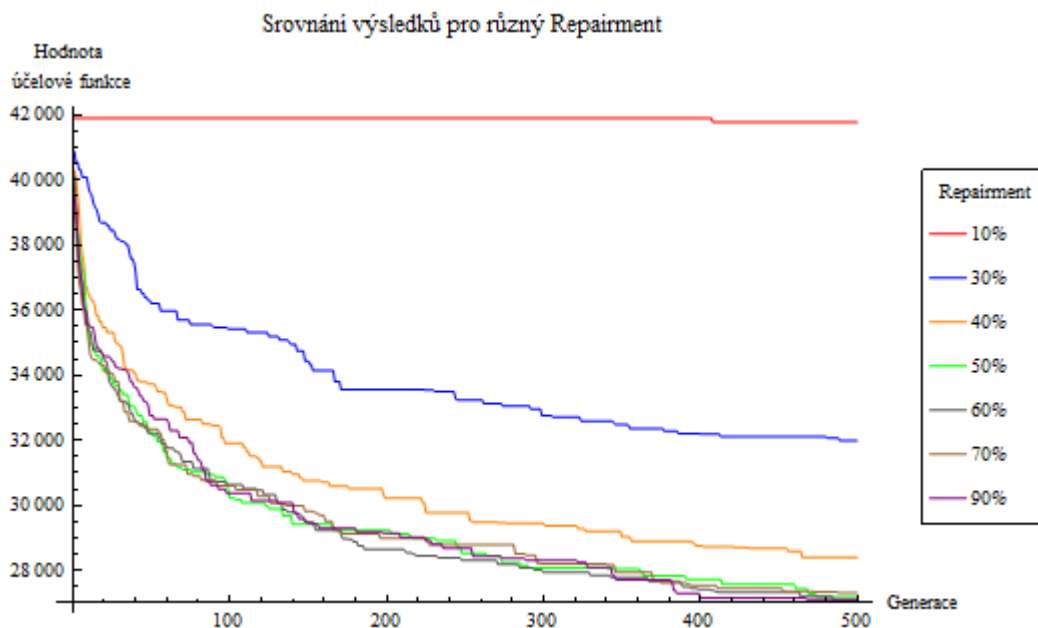
Na obrázcích Obr. 29. a Obr. 30 je srovnání průměrných hodnot a mediánů pro zvyšující se hranici hodnoty *Repairment*.

**Průměr:**



Obr. 29. Srovnání průměrů pro různé hodnoty hranice oprav

**Medián:**



Obr. 30. Srovnání mediánů pro různé hodnoty hranice oprav

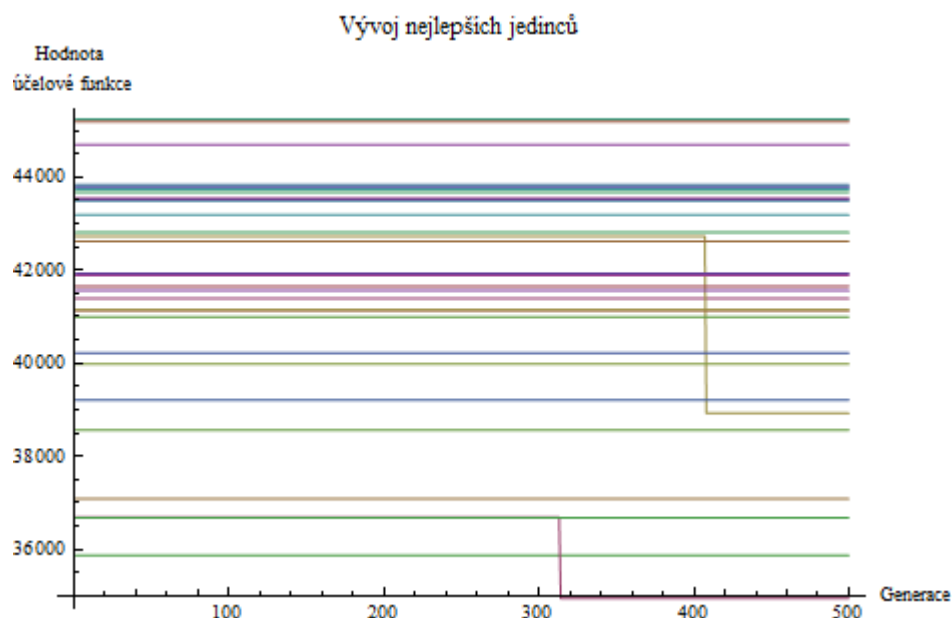
Nejnižší průměrná hodnota účelové funkce nejlepších jedinců po 30 opakováních byla 26978 při hranici 50% (na grafu vykreslen zelenou barvou).

Nejnižší medián hodnot účelové funkce nejlepších jedinců po 30 opakováních byl při hranici 60% a to 27074.

Podobných výsledků bylo dosaženo i při hranici 90% (průměr 26993, medián 27108), ale nutno podotknout, že při této hranici už se prakticky nejedná o evoluci, ale o náhodný jev, jelikož pouze 10% jedinců je tvořeno evolučním procesem. Relativně dobré výsledky jsou v tomto případě způsobeny náhodným vygenerováním hodně dobrého jedince oproti ostatním jedincům a jeho následným křížením se s dalšími jedinci - tím výrazně stoupne kvalita celé populace.

### 8.1.5 Nízká hranice oprav

Pokud je nastavena příliš nízká hranice oprav, může dojít ke stagnaci, jak je tomu na obrázku Obr. 31. Do dalších generací jsou vpouštěni jedinci jen pod 10% a méně shodných prvků, které je u nich třeba opravit, takže drtivou část nové generace tvoří jedinci z předchozí generace. Tím nedochází k evoluci, algoritmus tzv. stagnuje a nedospěje k vhodnému řešení.



Obr. 31. Stagnace při nízké hodnotě *Repairment*

## 8.2 Dimenze 20, generace 5000 a velikost populace 50

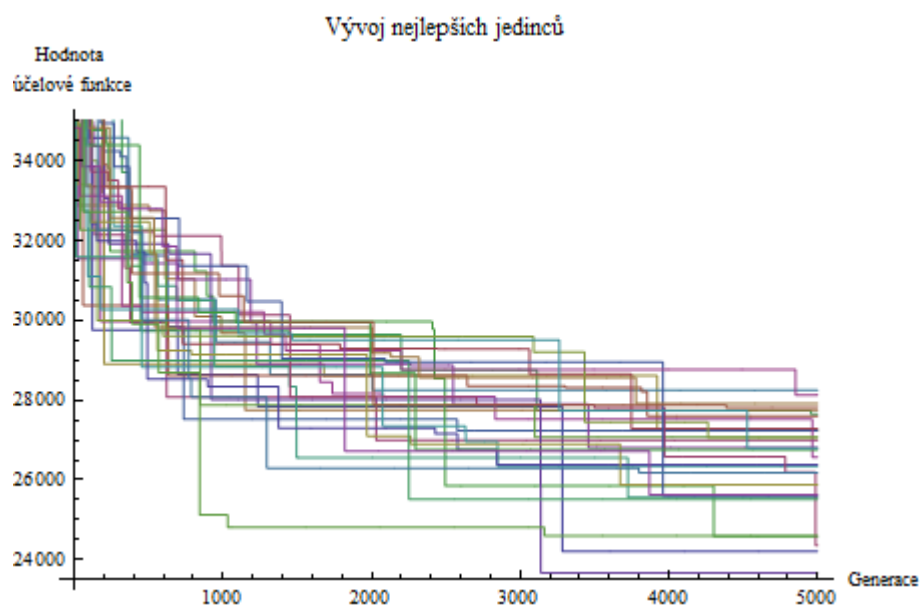
Oproti předchozímu testování, je nyní zvýšen počet generací z 500 na 5000 a velikost populace z 30 na 50, což by mělo přinést optimálnější řešení.

### 8.2.1 Repairment 0,3

Větší podíl na řešení má evoluce (70%), ve zbylých případech jsou to náhodně vygenerovaní jedinci.

Tab. 15. Nastavené parametry algoritmu pro malou dimenzi a vysoký počet generací

Parametr	Hodnota	Parametr	Hodnota	Parametr	Hodnota	Parametr	Hodnota
<b>Repairment</b>	<b>0,3</b>	<i>Dim</i>	20	<i>F</i>	0,8	<i>Gen</i>	5000
		<i>NP</i>	50	<i>CR</i>	0,5	<i>Repeat</i>	30

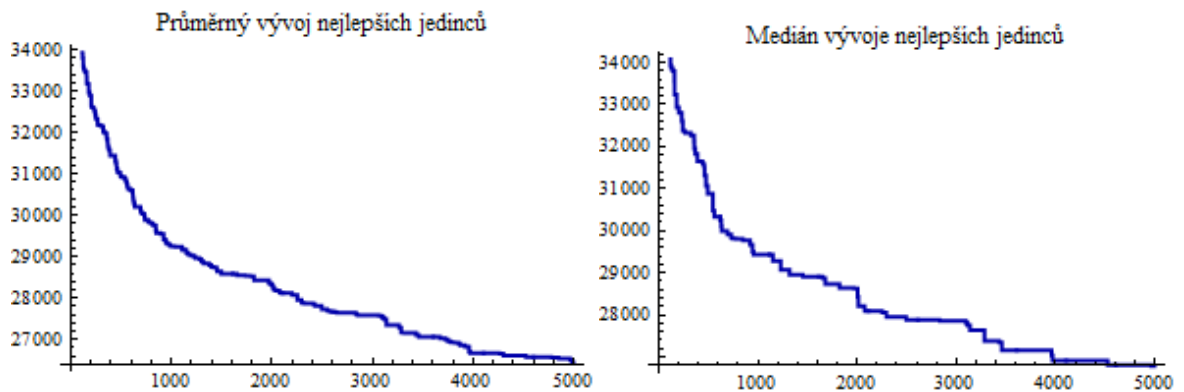


Obr. 32. Nejlepší jedinci

Z testování na Obr. 32. lze dle směrodatné odchylky nejlepších řešení usoudit, že rozptyl nejlepších jedinců je sice lepší, než pro nižší počet generací, ale stále je ještě dost vysoký, což může být způsobeno nízkou hodnotou *Repairment*, kdy se do nové populace dostává příliš málo nových jedinců a vstupují do ní pouze původní jedinci ze starších populací, ale také stále nedostatečně velkou populací. Hodnoty získané z měření jsou uvedeny v tabulce Tab. 16.

Tab. 16. Hodnoty získané z řešení

Nejlepší jedinec (hodnota účelové funkce)	<b>23669</b>
Parametry nejlepšího jedince	16, 14, 3, 2, 11, 9, 4, 6, 5, 7, 19, 15, 18, 17, 20, 13, 12, 1, 10, 8
Průměr nejlepších nalezených jedinců pro 30 opakování	26448
Medián nejlepších nalezených jedinců pro 30 opakování	26778
Směrodatná odchylka nejlepších jedinců	1251
Průměrný čas 1 opakování	165,782s
Celkový čas testu	4973,450s

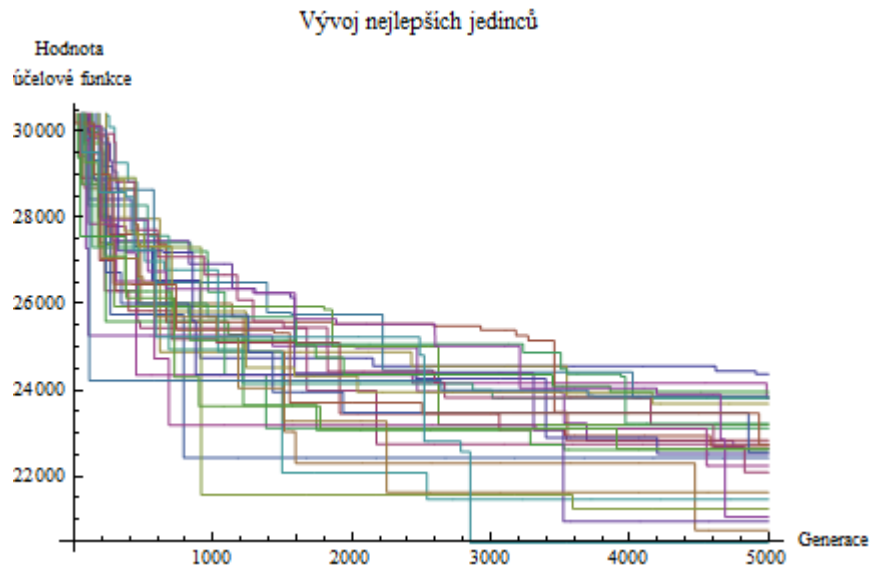


Obr. 33. Průměr a medián nejlepších nalezených jedinců

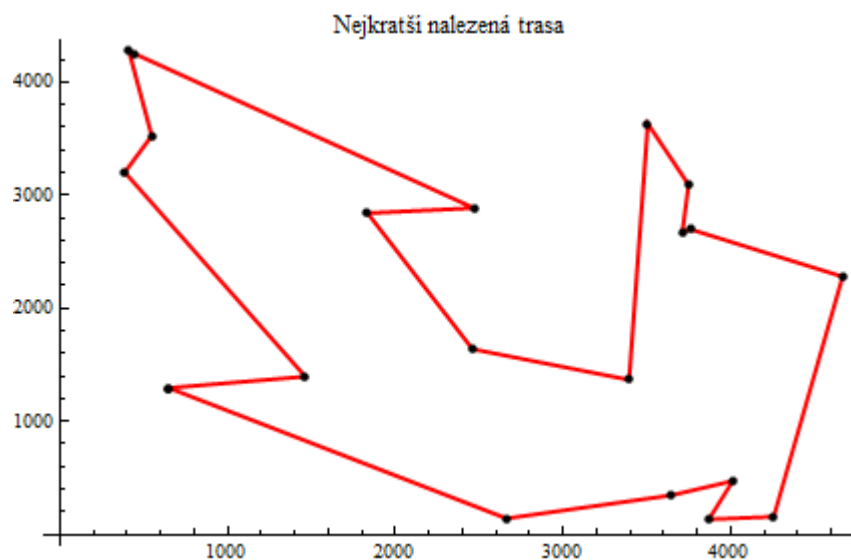
### 8.2.2 Repairment 0,5

Tab. 17. Nastavené parametry algoritmu pro malou dimenzi a vysoký počet generací

Parametr	Hodnota	Parametr	Hodnota	Parametr	Hodnota	Parametr	Hodnota
<i>Repairment</i>	<b>0,5</b>	<i>Dim</i>	20	<i>F</i>	0,8	<i>Gen</i>	5000
		<i>NP</i>	50	<i>CR</i>	0,5	<i>Repeat</i>	30



Obr. 34. Nejlepší jedinci



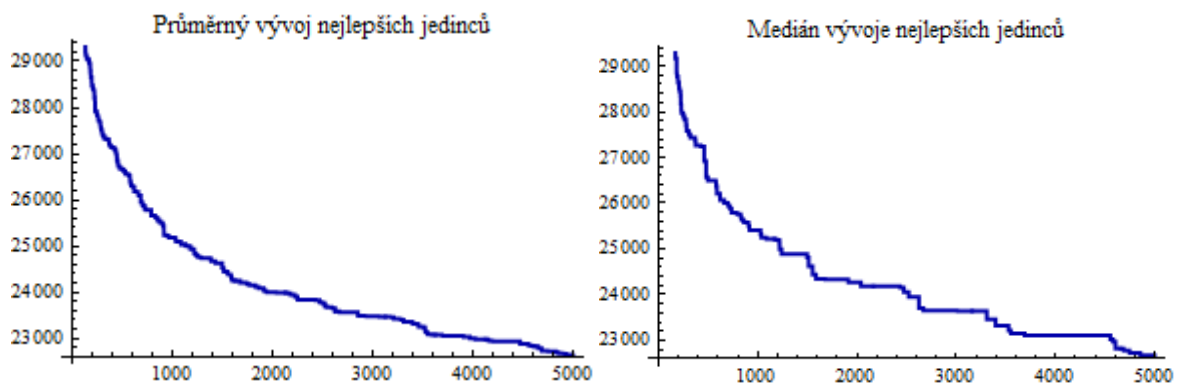
Obr. 35. Nejkratší nalezená trasa pomocí algoritmu DE

Z obrázku Obr. 35. je patrné, že nalezené řešení už je lepší a algoritmus směřuje do globálního minima. Chyba zde nastala nejspíše pouze u bodů uprostřed grafu a bodů na pravé straně.

Jak však plyne z definice optimalizačních algoritmů, jejich cílem není nalézt nejlepší řešení daného problému, ale optimální řešení v přijatelném čase, což lze v tomto případě označit jako optimální řešení.

Tab. 18. Hodnoty získané z řešení

Nejlepší jedinec (hodnota účelové funkce)	<b>20455</b>
Parametry nejlepšího jedince	6, 18, 7, 5, 15, 19, 9, 4, 17, 3, 11, 2, 14, 16, 8, 12, 13, 20, 1, 10
Průměr nejlepších nalezených jedinců pro 30 opakování	22631
Medián nejlepších nalezených jedinců pro 30 opakování	22673
Směrodatná odchylka nejlepších jedinců	1058
Průměrný čas 1 opakování	304,626s
Celkový čas testu	9138,770s

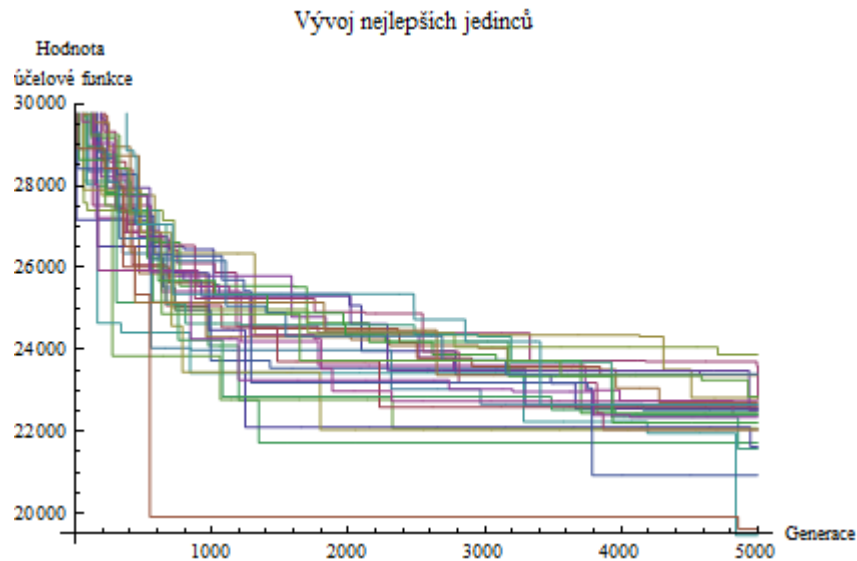


Obr. 36. Průměr a medián nejlepších nalezených jedinců

### 8.2.3 Repairment 0,7

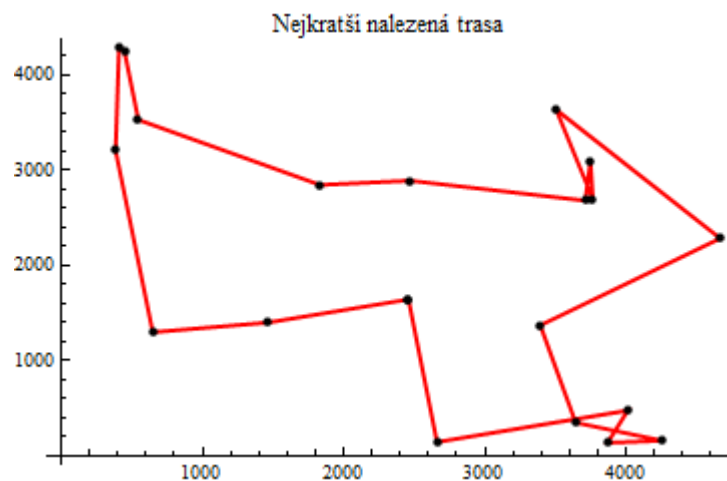
Tab. 19. Nastavené parametry algoritmu pro malou dimenzi a vysoký počet generací

Parametr	Hodnota	Parametr	Hodnota	Parametr	Hodnota	Parametr	Hodnota
<b>Repairment</b>	<b>0,7</b>	<i>Dim</i>	20	<i>F</i>	0,8	<i>Gen</i>	5000
		<i>NP</i>	50	<i>CR</i>	0,5	<i>Repeat</i>	30



Obr. 37. Nejlepší jedinci

Z obrázku Obr. 37. lze vidět velký rozdíl mezi nejhorším a nejlepším nalezeným jedincem (rozdíl hodnot účelové funkce více než 4000), i přesto, že směrodatná odchylka je menší než v předchozích případech. Tento fakt je pravděpodobně způsoben hodnotou *Repairment* na 0,7, která udává, že 70% řešení je věcí náhody, pouze 30% důsledkem evoluce. Tím pádem se v populaci mohl vyskytnout náhodně vygenerovaný jedinec, který byl mnohem kvalitnější než zbylí jedinci v populaci.

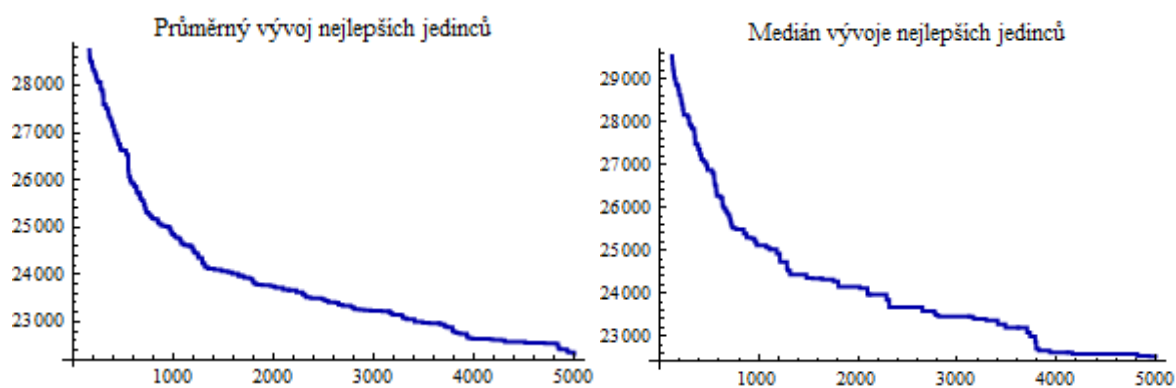


Obr. 38. Nejkratší nalezená trasa

Řešení opět není ideální, ale lze jej označit za optimální. Správné řešení však nelze dokázat deterministickým způsobem.

Tab. 20. Hodnoty získané z řešení

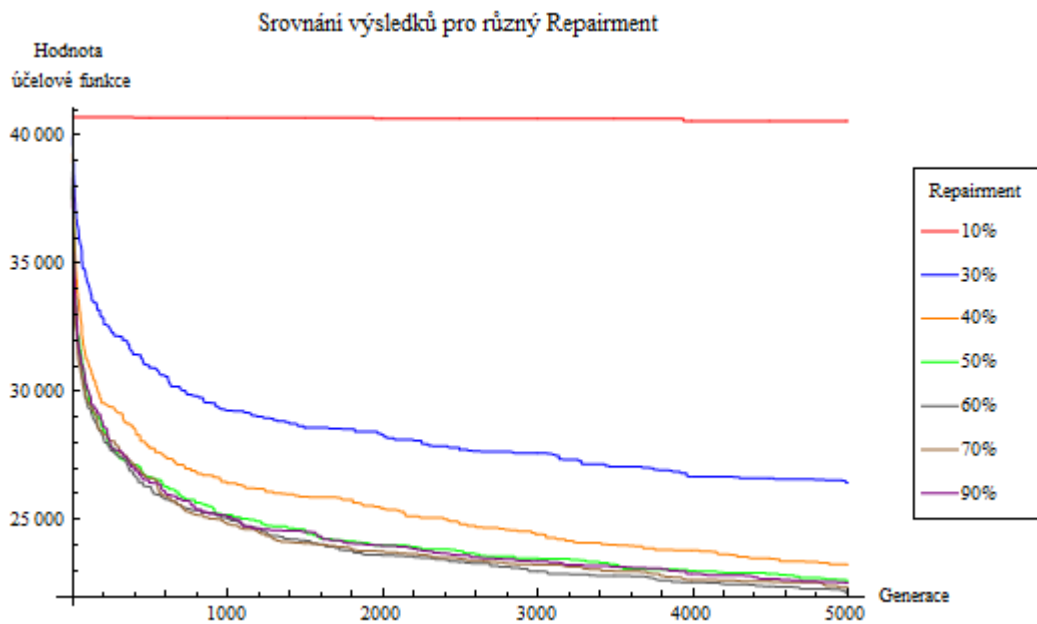
Nejlepší jedinec (hodnota účelové funkce)	<b>19466</b>
Parametry nejlepšího jedince	17, 18, 6, 7, 15, 19, 5, 4, 9, 14, 2, 16, 11, 8, 3, 1, 12, 13, 20, 10
Průměr nejlepších nalezených jedinců pro 30 opakování	22325
Medián nejlepších nalezených jedinců pro 30 opakování	22520
Směrodatná odchylka nejlepších jedinců	973
Průměrný čas 1 opakování	309,180s
Celkový čas testu	9275,390s



Obr. 39. Průměr a medián nejlepších nalezených jedinců

## 8.2.4 Porovnání průměrů a mediánů pro různou hranici oprav

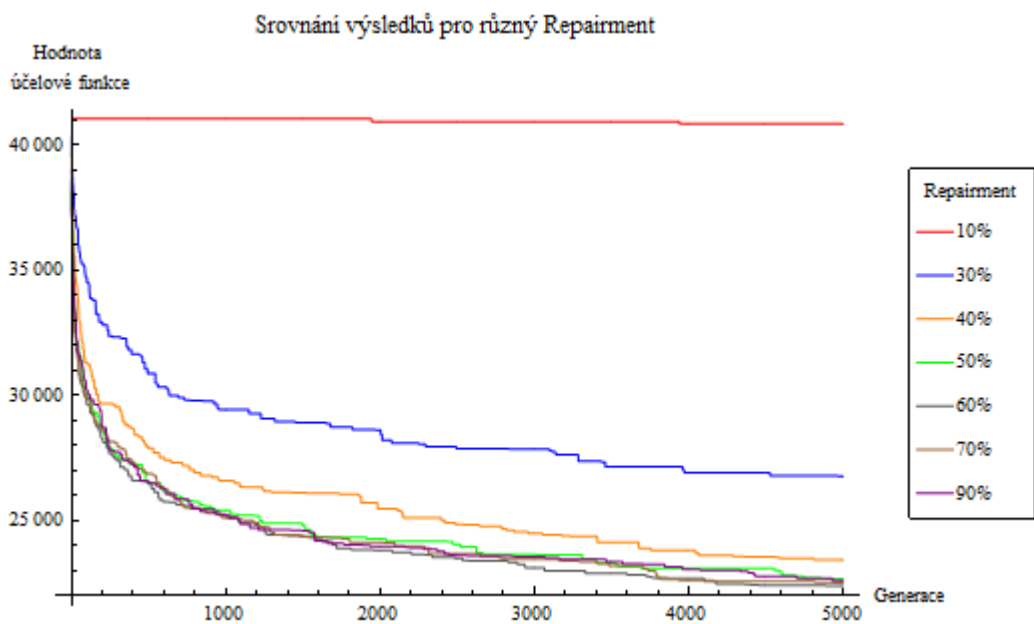
### Průměr:



Obr. 40. Srovnání průměrů pro různé hodnoty hranice oprav

Nejnižší průměrná hodnota účelové funkce nejlepších jedinců po 30 opakováních byla 22167 při hranici 60% (na grafu vykreslen šedou barvou).

### Medián:



Obr. 41. Srovnání mediánů pro různé hodnoty hranice oprav

Nejnižší medián hodnot účelové funkce nejlepších jedinců po 30 opakováních byl 22404 při hranici 60% (na grafu vykreslen šedou barvou).

### 8.3 Dimenze 50, generace 500 a velikost populace 30

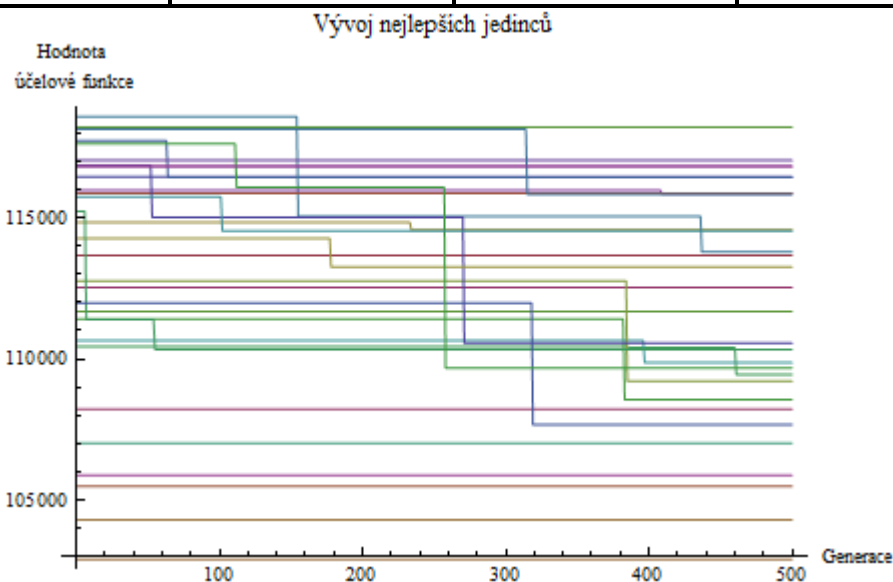
Velikost populace je v tomto případě poměrně malá vzhledem k velikosti dimenze. Je celkem pravděpodobné, že nejlepší jedinec se vyskytne už v první populaci a jeho hodnota se vzhledem k malému počtu generací, už se příliš nebo vůbec nezmění.

#### 8.3.1 Repairment 0,3

Nízká hranice oprav značí, že mnoho nových jedinců bude z populace vyřazeno a zůstanou zachováni původní jedinci. Tím pádem může algoritmus stagnovat a nemusí dojít k nijak výraznému vývoji.

Tab. 21. Nastavené parametry algoritmu pro střední dimenzi a nízký počet generací

Parametr	Hodnota	Parametr	Hodnota	Parametr	Hodnota	Parametr	Hodnota
<b>Repairment</b>	<b>0,3</b>	<i>Dim</i>	50	<i>F</i>	0,8	<i>Gen</i>	500
		<i>NP</i>	30	<i>CR</i>	0,5	<i>Repeat</i>	30



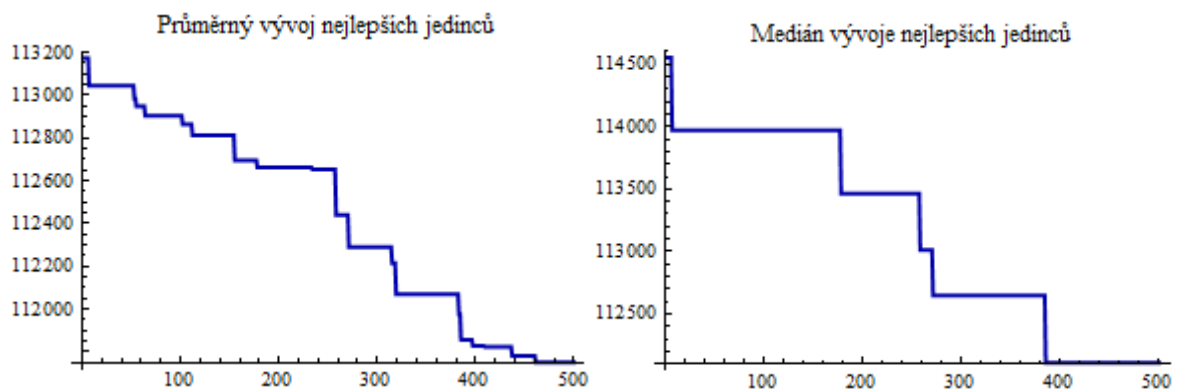
Obr. 42. Nejlepší jedinci

Z obrázku Obr. 42. je patrné, že při hranici oprav 30% algoritmus v mnoha případech stagnoval. To je způsobeno jak malým počtem generací (při větším počtu by jedinci postupně mutovali a křížili se a tím by se získávali stále lepší a lepší jedinci a algoritmus by, ač pomalu, pokračoval dál) a malou populací vzhledem k velikosti dimenze.

V případech, kdy nějaký vývoj jedinců probíhal, byl velmi pomalý, jelikož jen malá část jedinců prošla skrz opravu a zbytek byly původní jedinci z předchozí populace.

Tab. 22. Hodnoty získané z řešení

Nejlepší jedinec (hodnota účelové funkce)	<b>102900</b>
Parametry nejlepšího jedince	28, 35, 38, 50, 7, 29, 44, 3, 39, 9, 46, 8, 49, 10, 33, 47, 1, 14, 42, 41, 18, 17, 4, 11, 25, 31, 30, 2, 26, 22, 43, 40, 16, 6, 37, 24, 20, 23, 34, 19, 21, 12, 32, 13, 48, 5, 45, 15, 36, 27
Průměr nejlepších nalezených jedinců pro 30 opakování	111750
Medián nejlepších nalezených jedinců pro 30 opakování	112108
Směrodatná odchylka nejlepších jedinců	4294
Průměrný čas 1 opakování	19,721s
Celkový čas testu	591,641s

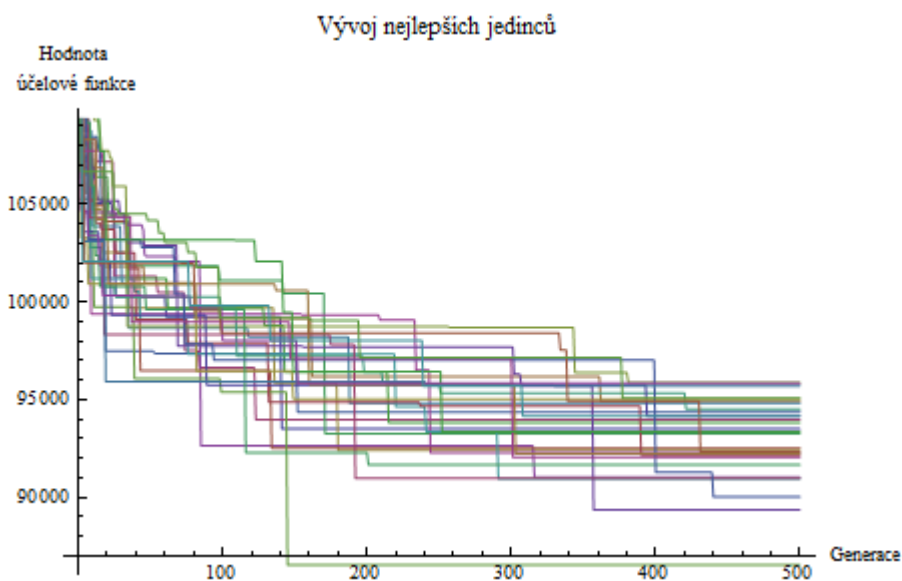


Obr. 43. Průměr a medián nejlepších nalezených jedinců

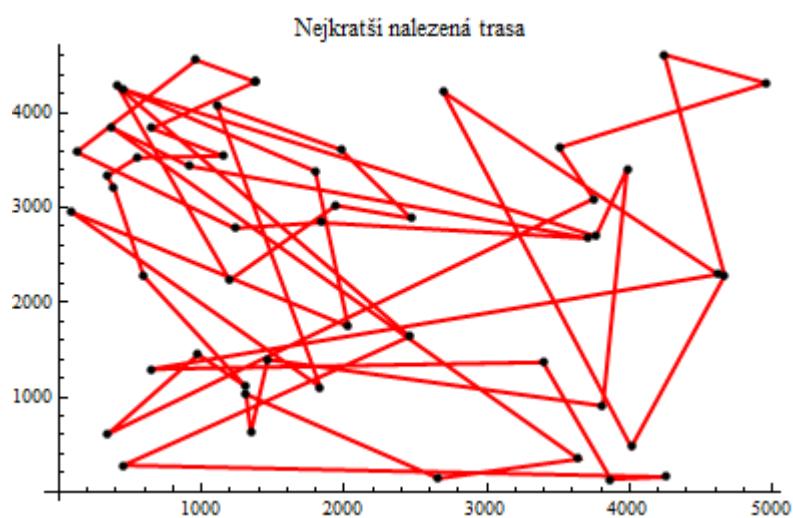
### 8.3.2 Repairment 0,5

Tab. 23. Nastavené parametry algoritmu pro střední dimenzi a nízký počet generací

Parametr	Hodnota	Parametr	Hodnota	Parametr	Hodnota	Parametr	Hodnota
<i>Repairment</i>	<b>0,5</b>	<i>Dim</i>	50	<i>F</i>	0,8	<i>Gen</i>	500
		<i>NP</i>	30	<i>CR</i>	0,5	<i>Repeat</i>	30



Obr. 44. Nejlepší jedinci

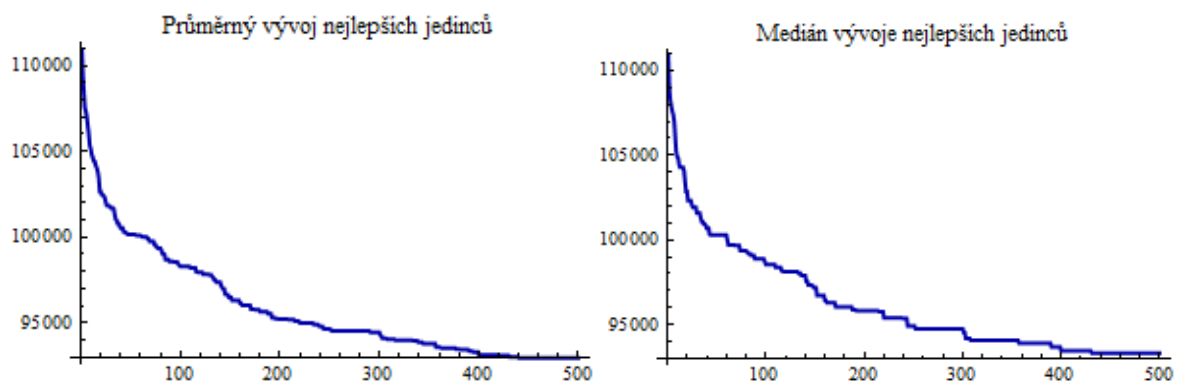


Obr. 45. Nejkratší nalezená trasa

Z obrázku Obr. 45. je zřejmé, že řešení je velmi špatné. Pro lepší řešení je třeba zvýšit především počet generací, ale i velikost populace.

Tab. 24. Hodnoty získané z řešení

Nejlepší jedinec (hodnota účelové funkce)	<b>86536</b>
Parametry nejlepšího jedince	43, 27, 39, 47, 4, 14, 40, 45, 1, 10, 41, 42, 18, 26, 30, 16, 19, 21, 44, 35, 49, 33, 38, 9, 48, 28, 15, 17, 46, 12, 13, 3, 6, 24, 37, 20, 8, 25, 31, 11, 2, 23, 29, 32, 22, 7, 34, 5, 50, 36
Průměr nejlepších nalezených jedinců pro 30 opakování	92968
Medián nejlepších nalezených jedinců pro 30 opakování	93324
Směrodatná odchylka nejlepších jedinců	2115
Průměrný čas 1 opakování	50,607s
Celkový čas testu	1518,20s

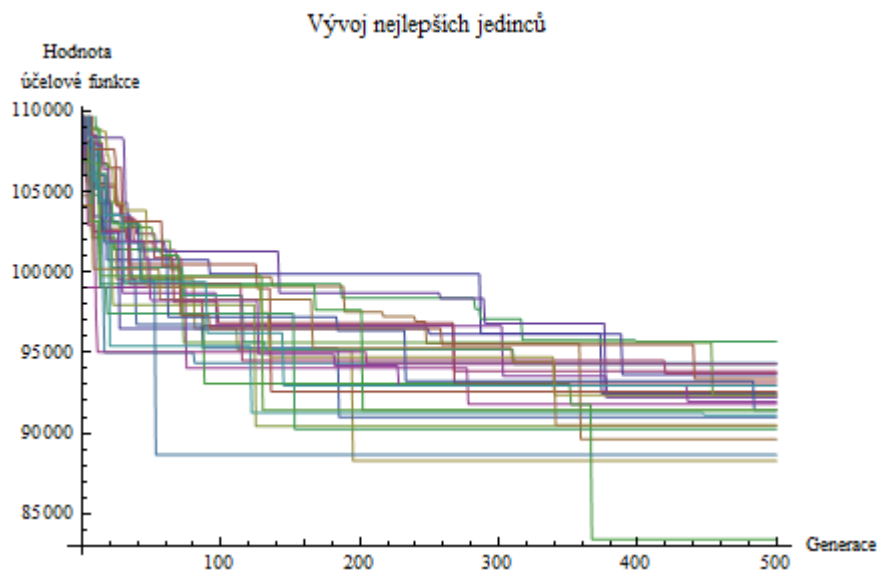


Obr. 46. Průměr a medián nejlepších nalezených jedinců

### 8.3.3 Repairment 0,7

Tab. 25. Nastavené parametry algoritmu pro střední dimenzi a nízký počet generací

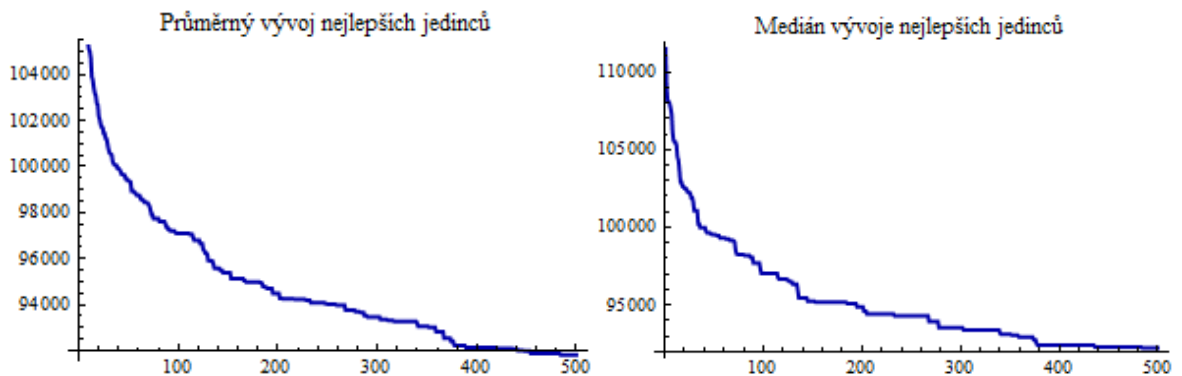
Parametr	Hodnota	Parametr	Hodnota	Parametr	Hodnota	Parametr	Hodnota
<b>Repairment</b>	<b>0,7</b>	<i>Dim</i>	50	<i>F</i>	0,8	<i>Gen</i>	500
		<i>NP</i>	30	<i>CR</i>	0,5	<i>Repeat</i>	30



Obr. 47. Nejlepší jedinci

Tab. 26. Hodnoty získané z řešení

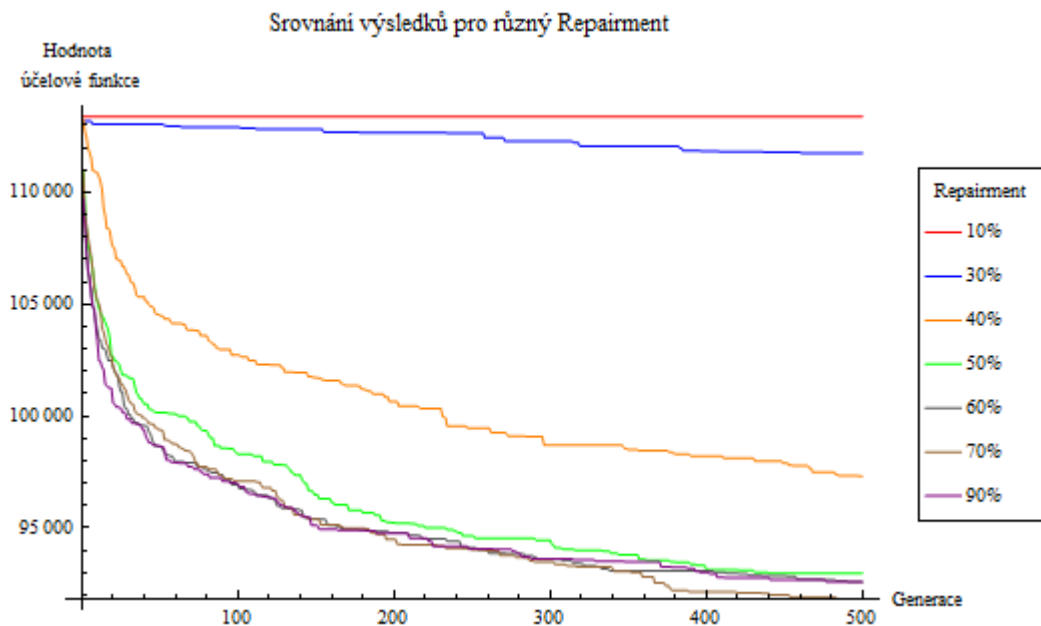
Nejlepší jedinec (hodnota účelové funkce)	<b>83394</b>
Parametry nejlepšího jedince	22, 28, 24, 3, 5, 27, 36, 48, 33, 32, 15, 7, 34, 35, 45, 43, 23, 46, 6, 40, 39, 47, 2, 11, 14, 8, 31, 20, 13, 10, 41, 29, 17, 25, 30, 44, 37, 9, 16, 38, 50, 12, 49, 18, 42, 1, 26, 4, 21, 19
Průměr nejlepších nalezených jedinců pro 30 opakování	91822
Medián nejlepších nalezených jedinců pro 30 opakování	92245
Směrodatná odchylka nejlepších jedinců	2335
Průměrný čas 1 opakování	51,580s
Celkový čas testu	1547,41s



Obr. 48. Průměr a medián nejlepších nalezených jedinců

### 8.3.4 Porovnání průměrů a mediánů pro různou hranici oprav

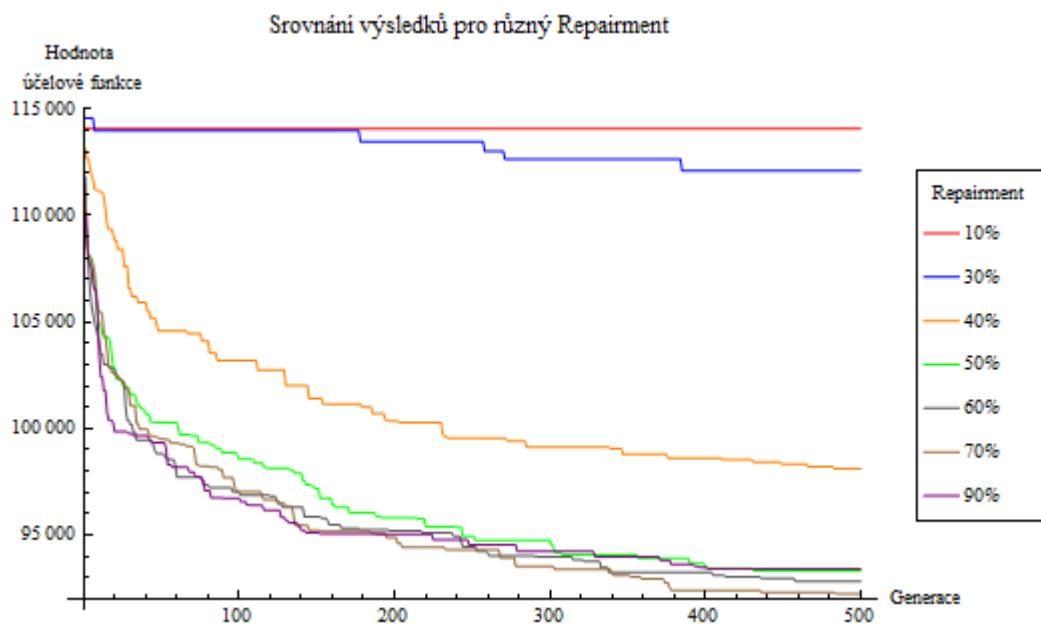
**Průměr:**



Obr. 49. Srovnání průměrů pro různé hodnoty hranice oprav

Nejnižší průměrná hodnota účelové funkce nejlepších jedinců po 30 opakováních byla 91822 při hranici 70% (na grafu vykreslen hnědou barvou).

**Medián:**



Obr. 50. Srovnání mediánů pro různé hodnoty hranice oprav

Nejnižší medián hodnot účelové funkce nejlepších jedinců po 30 opakováních byl 92245 při hranici 70% (na grafu vykreslen hnědou barvou).

Při hranici 10% a 30% byl vývoj velmi pomalý nebo nebyl vůbec žádný (stagnace), jelikož algoritmus neměl dostatečný počet generací a dostatečně velkou populaci.

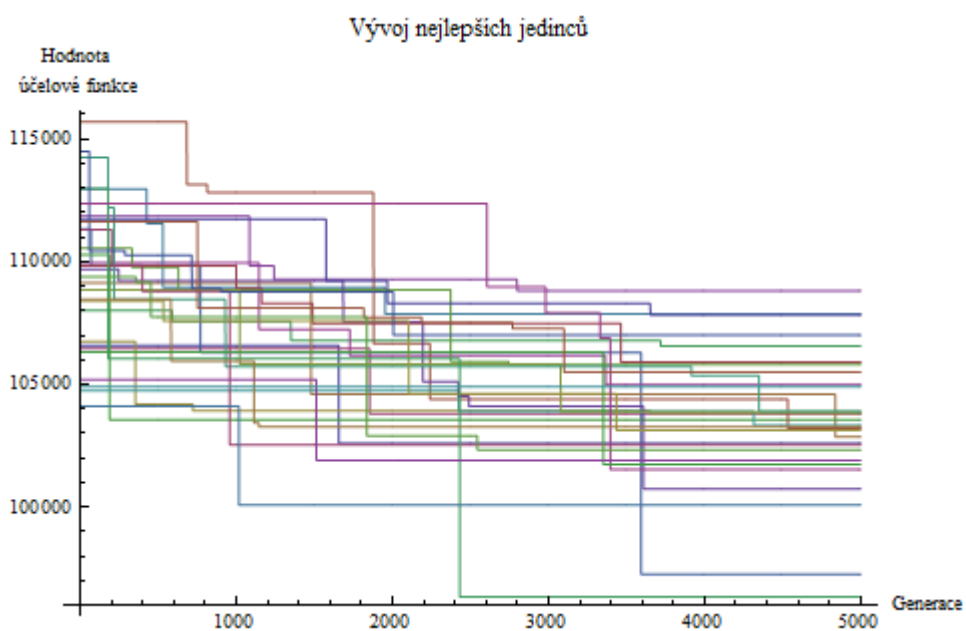
#### 8.4 Dimenze 50, generace 5000 a velikost populace 75

V tomto případě se vytvoří desetkrát více generací a populace je 2,5 krát větší než v případě předchozím. Lze tedy očekávat mnohem lepší vývoj jedinců.

##### 8.4.1 Repairment 0,3

Tab. 27. Nastavené parametry algoritmu pro střední dimenzi a vysoký počet generací

Parametr	Hodnota	Parametr	Hodnota	Parametr	Hodnota	Parametr	Hodnota
<i>Repairment</i>	0,3	<i>Dim</i>	50	<i>F</i>	0,8	<i>Gen</i>	5000
		<i>NP</i>	75	<i>CR</i>	0,5	<i>Repeat</i>	30

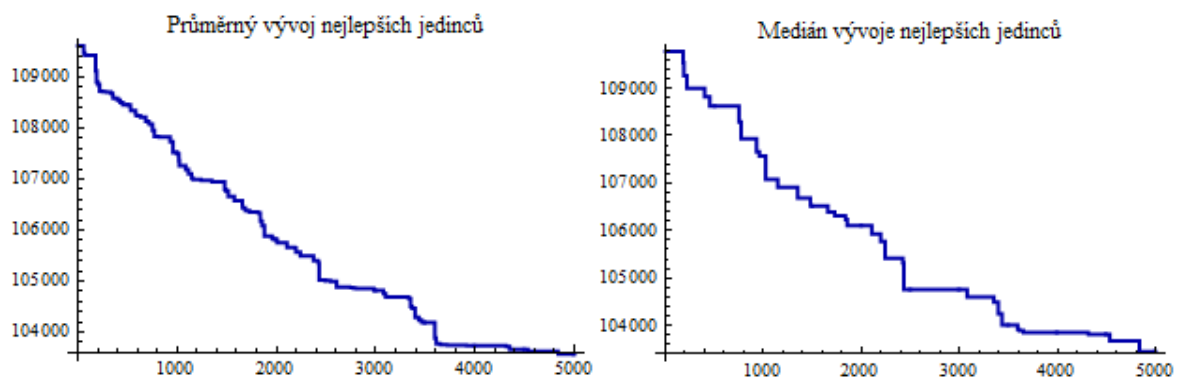


Obr. 51. Nejlepší jedinci

Z obrázku Obr. 51. lze vidět poměrně velký rozdíl mezi nejhorším a nejlepším nalezeným jedincem. Hodnota *Repairment* 0,3 je pro takto nastavené parametry příliš malá.

Tab. 28. Hodnoty získané z řešení

Nejlepší jedinec (hodnota účelové funkce)	<b>96334</b>
Parametry nejlepšího jedince	36, 7, 17, 44, 21, 45, 27, 39, 22, 41, 12, 6, 23, 35, 47, 33, 43, 14, 20, 32, 13, 1, 37, 30, 8, 5, 49, 18, 48, 34, 28, 4, 15, 40, 29, 19, 38, 3, 10, 11, 42, 26, 24, 16, 2, 31, 25, 50, 46, 9
Průměr nejlepších nalezených jedinců pro 30 opakování	103566
Medián nejlepších nalezených jedinců pro 30 opakování	103446
Směrodatná odchylka nejlepších jedinců	2819
Průměrný čas 1 opakování	494,72s
Celkový čas testu	14841,7s

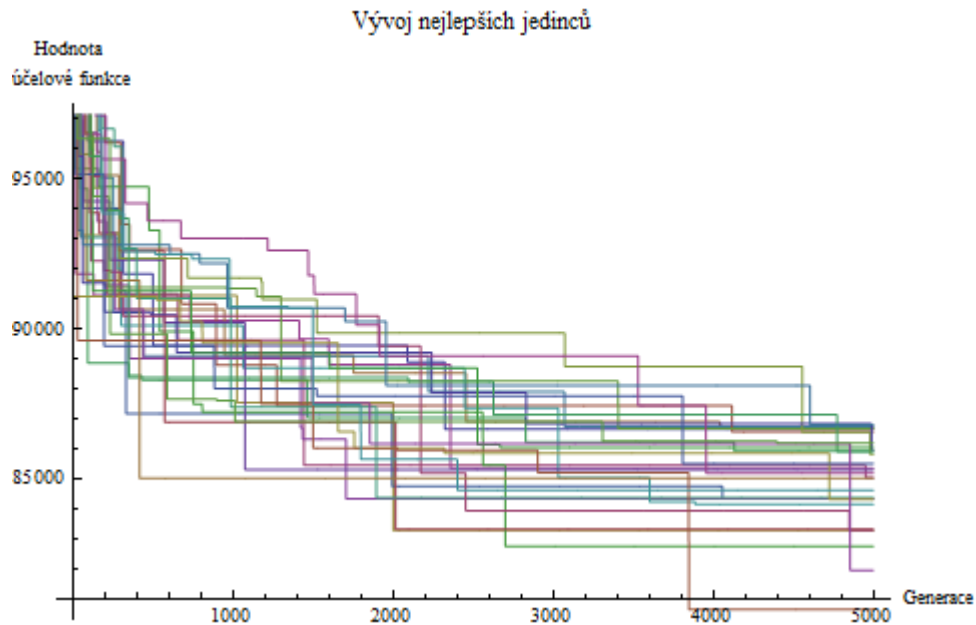


Obr. 52. Průměr a medián nejlepších nalezených jedinců

### 8.4.2 Repairment 0,5

Tab. 29. Nastavené parametry algoritmu pro střední dimenzi a vysoký počet generací

Parametr	Hodnota	Parametr	Hodnota	Parametr	Hodnota	Parametr	Hodnota
<b>Repairment</b>	<b>0,5</b>	<i>Dim</i>	50	<i>F</i>	0,8	<i>Gen</i>	5000
		<i>NP</i>	75	<i>CR</i>	0,5	<i>Repeat</i>	30

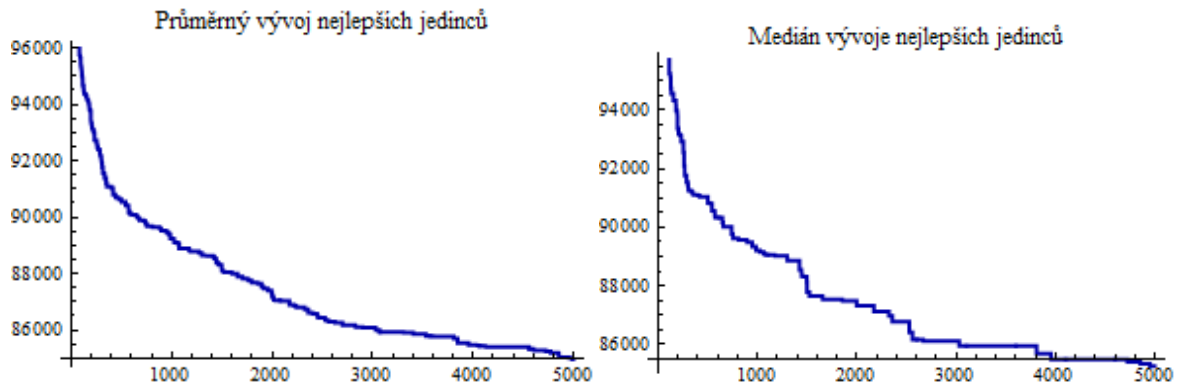


Obr. 53. Nejlepší jedinci

Rozdíl mezi nejhorším a nejlepším nalezením jedincem je při hranici oprav 50% výrazně lepší než při 30%.

Tab. 30. Hodnoty získané z řešení

Nejlepší jedinec (hodnota účelové funkce)	<b>80647</b>
Parametry nejlepšího jedince	11, 27, 14, 2, 31, 30, 12, 10, 42, 18, 28, 21, 5, 35, 22, 6, 48, 16, 24, 8, 20, 4, 23, 46, 32, 49, 39, 34, 44, 41, 15, 7, 43, 38, 19, 45, 33, 50, 17, 26, 3, 29, 37, 40, 36, 47, 25, 1, 13, 9
Průměr nejlepších nalezených jedinců pro 30 opakování	84960
Medián nejlepších nalezených jedinců pro 30 opakování	85252
Směrodatná odchylka nejlepších jedinců	1531
Průměrný čas 1 opakování	1224,93s
Celkový čas testu	36748,0s



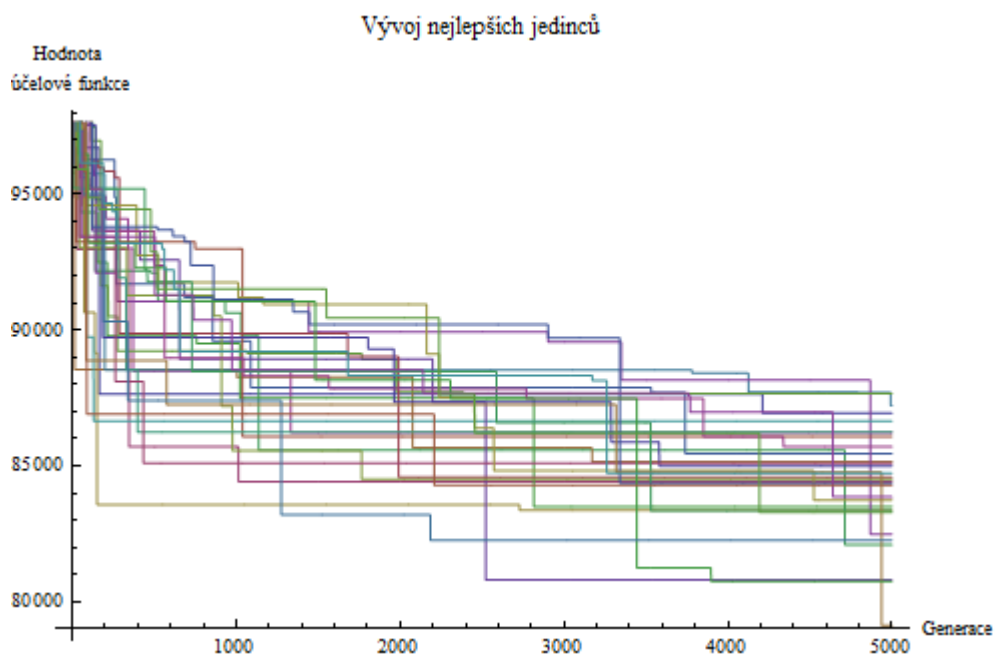
Obr. 54. Průměr a medián nejlepších nalezených jedinců

### 8.4.3 Repairment 0,7

Evoluce je zachována pouze z 30%, zbytek je věci náhodného generování parametrů jedinců.

Tab. 31. Nastavené parametry algoritmu pro střední dimenzi a vysoký počet generací

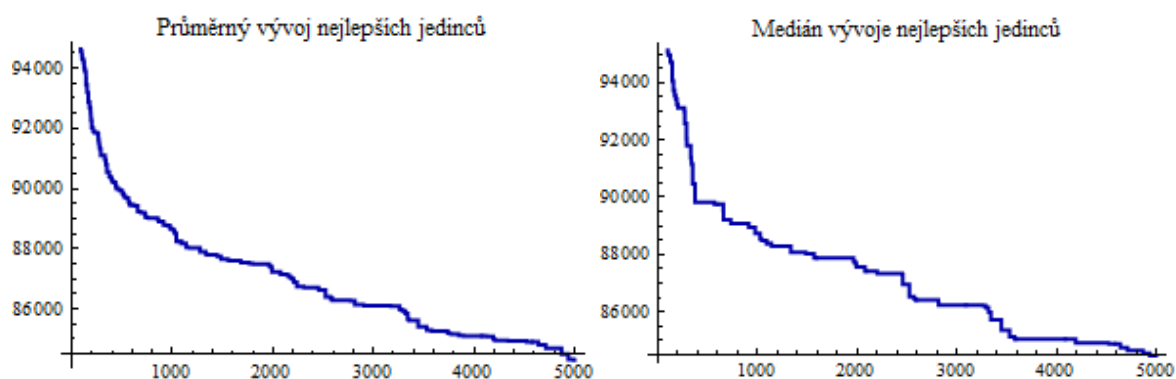
Parametr	Hodnota	Parametr	Hodnota	Parametr	Hodnota	Parametr	Hodnota
<b>Repairment</b>	<b>0,7</b>	<i>Dim</i>	50	<i>F</i>	0,8	<i>Gen</i>	5000
		<i>NP</i>	75	<i>CR</i>	0,5	<i>Repeat</i>	30



Obr. 55. Nejlepší jedinci

Tab. 32. Hodnoty získané z řešení

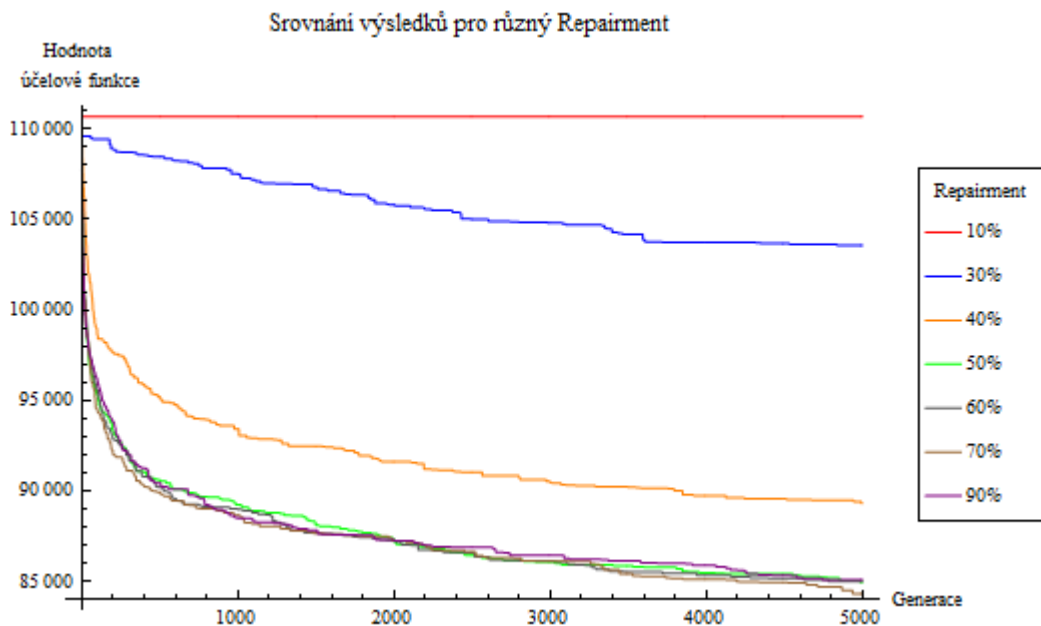
Nejlepší jedinec (hodnota účelové funkce)	<b>79125</b>
Parametry nejlepšího jedince	48, 46, 36, 40, 12, 20, 26, 31, 25, 11, 37, 9, 43, 21, 50, 5, 45, 15, 7, 47, 49, 1, 41, 17, 18, 10, 4, 38, 44, 32, 6, 28, 3, 24, 16, 2, 33, 35, 34, 19, 27, 22, 23, 29, 39, 42, 14, 30, 8, 13
Průměr nejlepších nalezených jedinců pro 30 opakování	84297
Medián nejlepších nalezených jedinců pro 30 opakování	84464
Směrodatná odchylka nejlepších jedinců	2004
Průměrný čas 1 opakování	1274,60s
Celkový čas testu	38237,9s



Obr. 56. Průměr a medián nejlepších nalezených jedinců

#### 8.4.4 Porovnání průměrů a mediánů pro různou hranici oprav

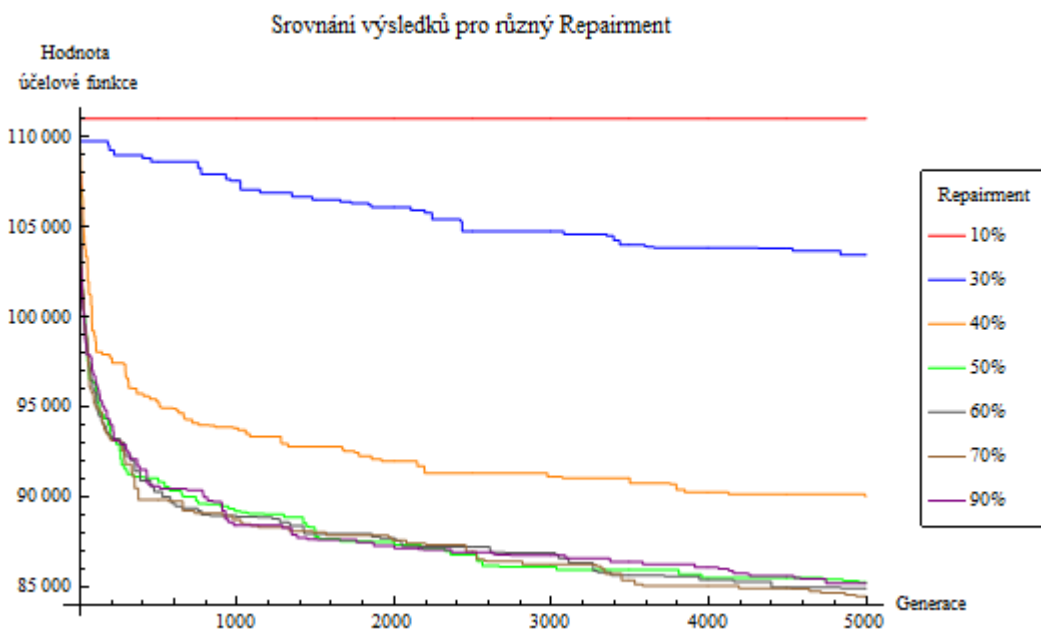
**Průměr:**



Obr. 57. Srovnání průměrů pro různé hodnoty hranice oprav

Nejnižší průměrná hodnota účelové funkce nejlepších jedinců po 30 opakováních byla 84297 při hranici 70% (na grafu vykreslen hnědou barvou).

**Medián:**



Obr. 58. Srovnání mediánů pro různé hodnoty hranice oprav

Nejnižší medián hodnot účelové funkce nejlepších jedinců po 30 opakováních byla 84464 při hranici 70% (na grafu vykreslen hnědou barvou).

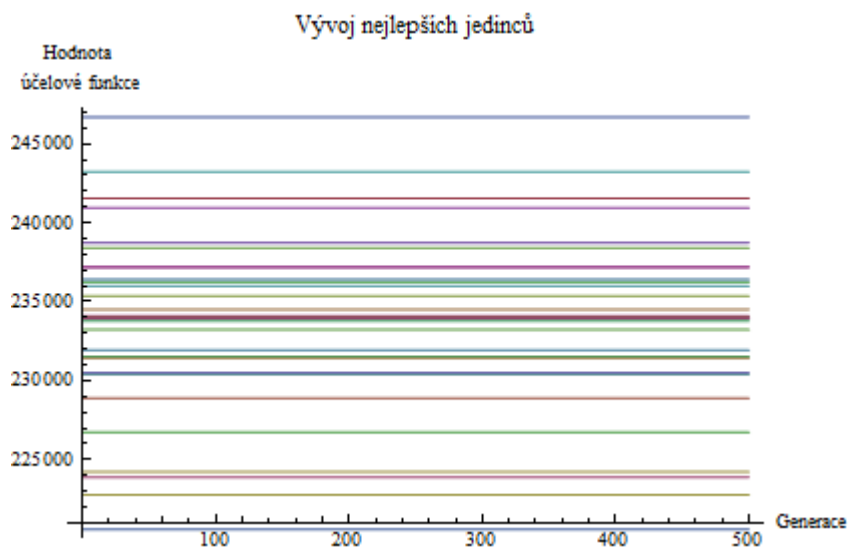
## 8.5 Dimenze 100, generace 500 a velikost populace 30

Velikost populace i počet generací jsou vzhledem k dimenzi velmi malé. Lze očekávat stagnaci algoritmu nebo velmi pomalý vývoj.

### 8.5.1 Repairment 0,3

Tab. 33. Nastavené parametry algoritmu pro velkou dimenzi a malý počet generací

Parametr	Hodnota	Parametr	Hodnota	Parametr	Hodnota	Parametr	Hodnota
<i>Repairment</i>	<b>0,3</b>	<i>Dim</i>	100	<i>F</i>	0,8	<i>Gen</i>	500
		<i>NP</i>	30	<i>CR</i>	0,5	<i>Repeat</i>	30

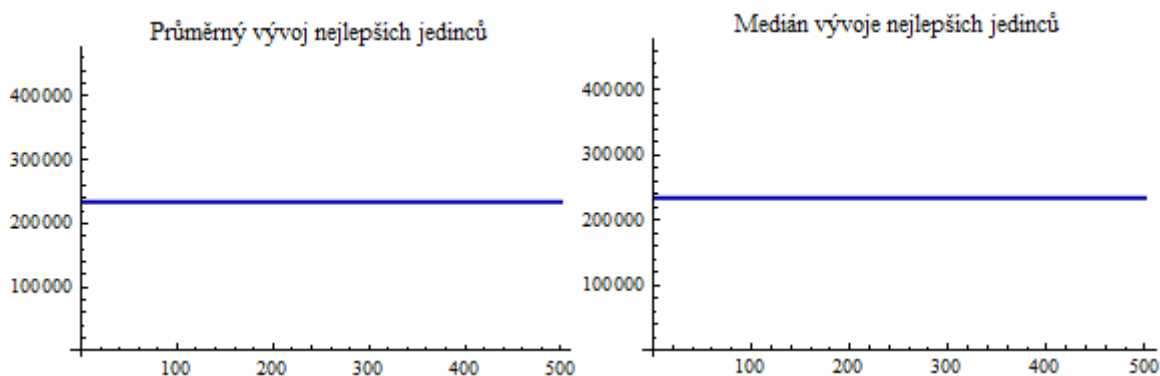


Obr. 59. Nejlepší jedinci

Algoritmus stagnuje, populace je velmi malá a počet generací nedostačuje k tomu, aby nějaký vývoj vůbec začal. Kvůli malé hranici oprav jsou do dalších populací vpouštěni jedinci z předchozí populace.

Tab. 34. Hodnoty získané z řešení

Nejlepší jedinec (hodnota účelové funkce)	<b>220573</b>
Průměr nejlepších nalezených jedinců pro 30 opakování	233613
Medián nejlepších nalezených jedinců pro 30 opakování	234018
Směrodatná odchylka nejlepších jedinců	6028
Průměrný čas 1 opakování	38,104s
Celkový čas testu	1143,110s



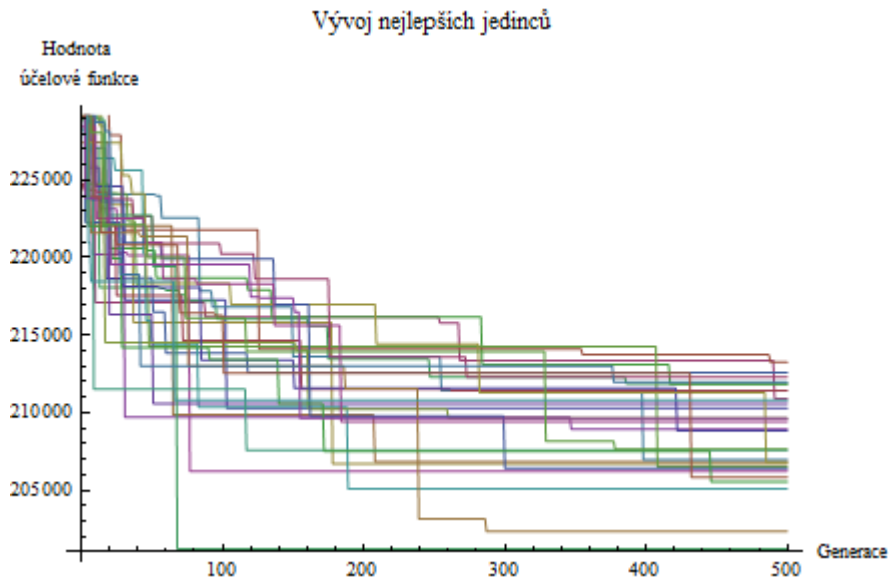
Obr. 60. Průměr a medián nejlepších nalezených jedinců

Z Obr. 60. je zřejmé, že nedošlo k žádnému vývoji původně vygenerované populace.

### 8.5.2 Repairment 0,5

Tab. 35. Nastavené parametry algoritmu pro velkou dimenzi a malý počet generací

Parametr	Hodnota	Parametr	Hodnota	Parametr	Hodnota	Parametr	Hodnota
<b>Repairment</b>	<b>0,5</b>	<i>Dim</i>	100	<i>F</i>	0,8	<i>Gen</i>	500
		<i>NP</i>	30	<i>CR</i>	0,5	<i>Repeat</i>	30

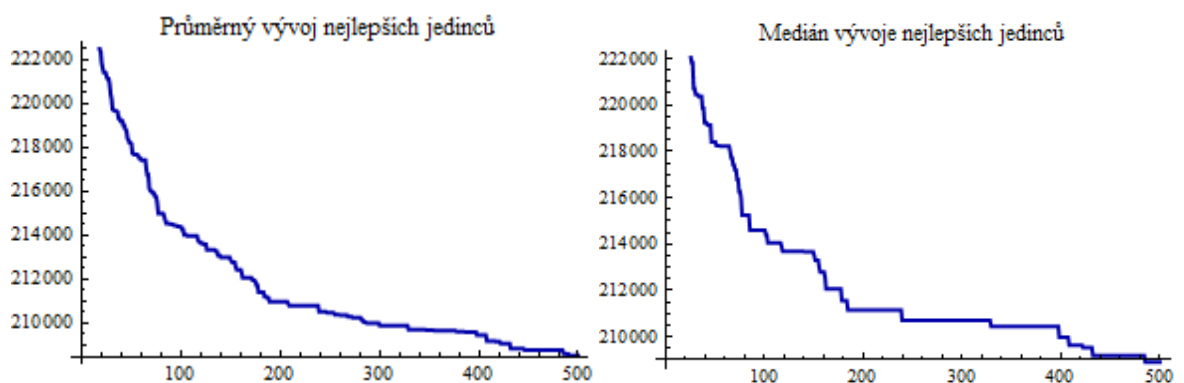


Obr. 61. Nejlepší jedinci

Oproti předchozímu případu už lze vidět, že probíhá vývoj jedinců. Do nových populací je nyní vpouštěno větší množství nově vytvořených a kvalitnějších jedinců.

Tab. 36. Hodnoty získané z řešení

Nejlepší jedinec (hodnota účelové funkce)	<b>201217</b>
Průměr nejlepších nalezených jedinců pro 30 opakování	208532
Medián nejlepších nalezených jedinců pro 30 opakování	208891
Směrodatná odchylka nejlepších jedinců	3016
Průměrný čas 1 opakování	105,463s
Celkový čas testu	3163,890s

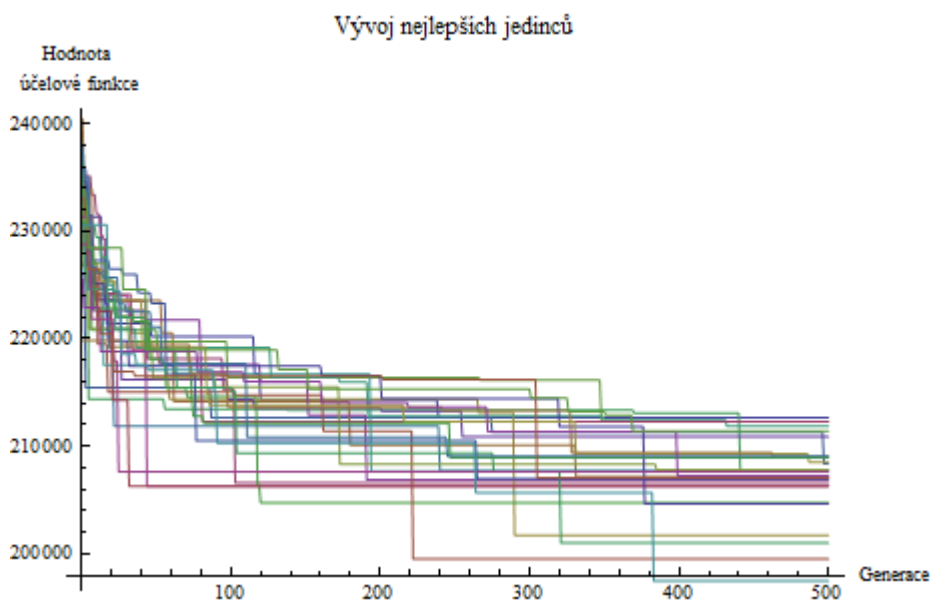


Obr. 62. Průměr a medián nejlepších nalezených jedinců

## 8.5.3 Repairment 0,7

Tab. 37. Nastavené parametry algoritmu pro velkou dimenzi a malý počet generací

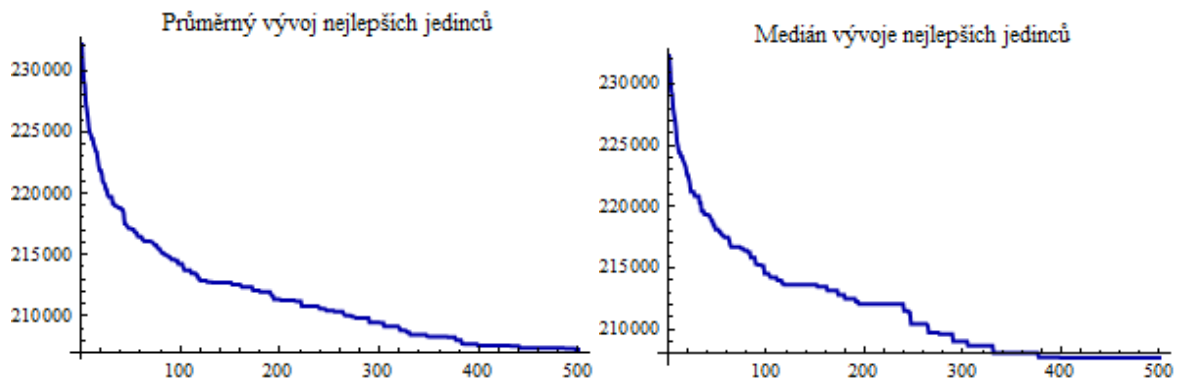
Parametr	Hodnota	Parametr	Hodnota	Parametr	Hodnota	Parametr	Hodnota
<i>Repairment</i>	<b>0,7</b>	<i>Dim</i>	100	<i>F</i>	0,8	<i>Gen</i>	500
		<i>NP</i>	30	<i>CR</i>	0,5	<i>Repeat</i>	30



Obr. 63. Nejlepší jedinci

Tab. 38. Hodnoty získané z řešení

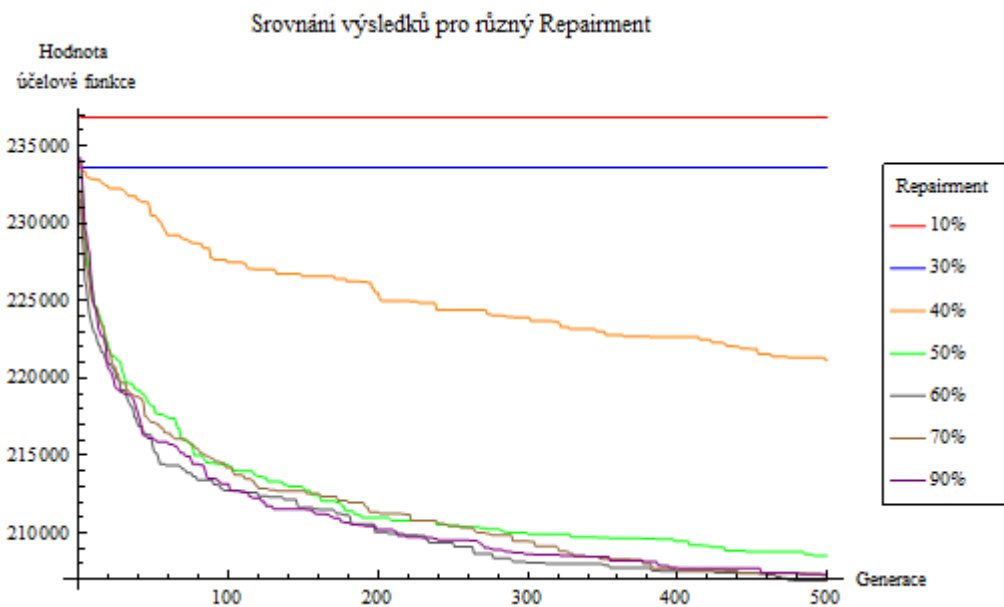
Nejlepší jedinec (hodnota účelové funkce)	<b>197458</b>
Průměr nejlepších nalezených jedinců pro 30 opakování	207252
Medián nejlepších nalezených jedinců pro 30 opakování	207646
Směrodatná odchylka nejlepších jedinců	2646
Průměrný čas 1 opakování	121,357s
Celkový čas testu	3640,700s



Obr. 64. Průměr a medián nejlepších nalezených jedinců

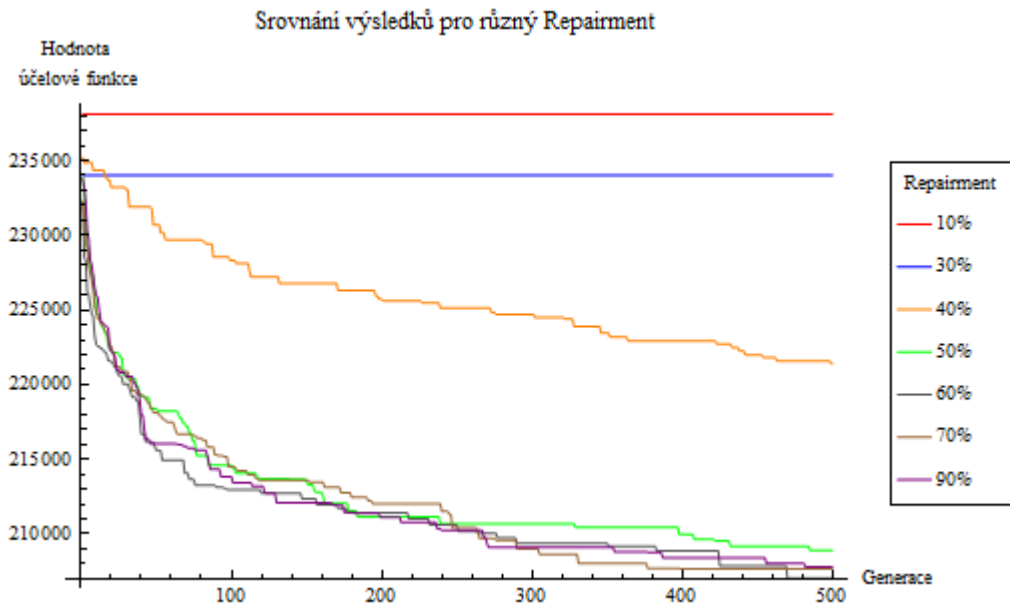
#### 8.5.4 Porovnání průměrů a mediánů pro různou hranici oprav

**Průměr:**



Obr. 65. Srovnání průměrů pro různé hodnoty hranice oprav

Nejnižší průměrná hodnota účelové funkce nejlepších jedinců po 30 opakováních byla 206908 při hranici 60% (na grafu vykreslen šedou barvou).

**Medián:**

Obr. 66. Srovnání mediánů pro různé hodnoty hranice oprav

Nejnižší medián hodnot účelové funkce nejlepších jedinců po 30 opakováních byla 207036 při hranici 60% (na grafu vykreslen šedou barvou).

Z obrázků Obr. 65. a Obr. 66. je zřejmé, že pro hranici oprav 10% a 30% algoritmus stagnoval a neprobíhal téměř žádný vývoj. Pro hranici oprav 40% už vývoj probíhal, ale byl velmi pomalý a pro nalezení optimálního řešení by bylo třeba velkého počtu generací. Pro vyšší hranice oprav už vývoj probíhal v pořádku a algoritmus směřoval do minima.

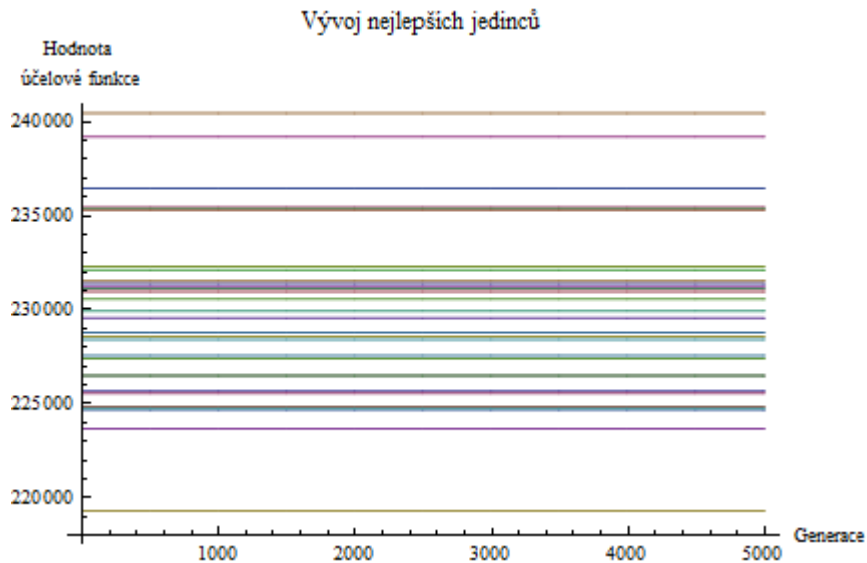
## 8.6 Dimenze 100, generace 5000 a velikost populace 75

Velikost populace i počet generací je zvýšen, lze očekávat mnohem lepších výsledků než v předchozím případě.

### 8.6.1 Repairment 0,3

Tab. 39. Nastavené parametry algoritmu pro velkou dimenzi a velký počet generací

Parametr	Hodnota	Parametr	Hodnota	Parametr	Hodnota	Parametr	Hodnota
<b>Repairment</b>	<b>0,3</b>	<i>Dim</i>	100	<i>F</i>	0,8	<i>Gen</i>	5000
		<i>NP</i>	75	<i>CR</i>	0,5	<i>Repeat</i>	30

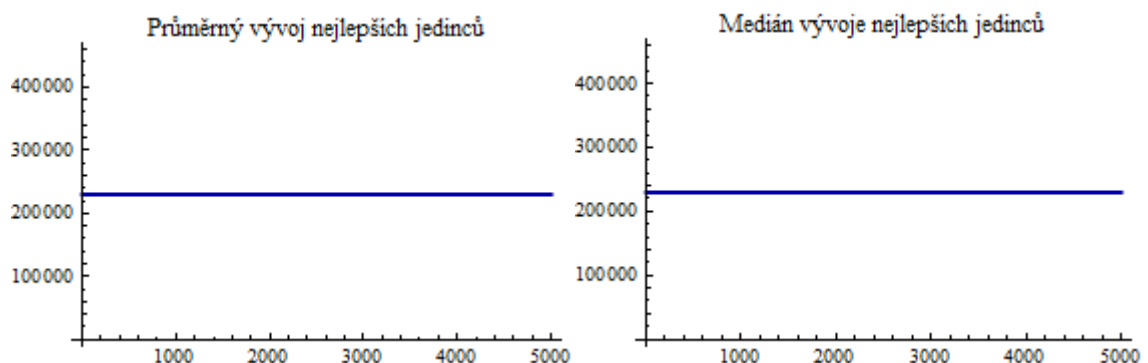


Obr. 67. Nejlepší jedinci

Algoritmus i v tomto případě stagnuje. Kvůli malé hranici oprav jsou do dalších populací vpouštěni jedinci z předchozí populace.

Tab. 40. Hodnoty získané z řešení

Nejlepší jedinec (hodnota účelové funkce)	<b>219300</b>
Průměr nejlepších nalezených jedinců pro 30 opakování	229835
Medián nejlepších nalezených jedinců pro 30 opakování	229740
Směrodatná odchylka nejlepších jedinců	4765
Průměrný čas 1 opakování	1570,6s
Celkový čas testu	47119,2s



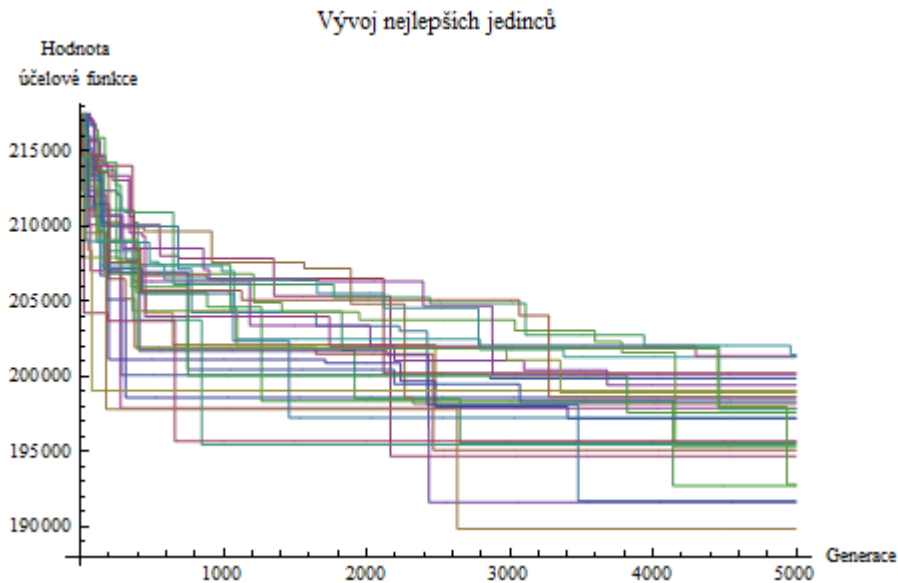
Obr. 68. Průměr a medián nejlepších nalezených jedinců

Z Obr. 68. je zřejmé, že nedošlo k téměř žádnému vývoji původně vygenerované populace. Je třeba zvýšit hodnotu *Repairment*.

### 8.6.2 Repairment 0,5

Tab. 41. Nastavené parametry algoritmu pro velkou dimenzi a velký počet generací

Parametr	Hodnota	Parametr	Hodnota	Parametr	Hodnota	Parametr	Hodnota
<i>Repairment</i>	<b>0,5</b>	<i>Dim</i>	100	<i>F</i>	0,8	<i>Gen</i>	5000
		<i>NP</i>	75	<i>CR</i>	0,5	<i>Repeat</i>	30

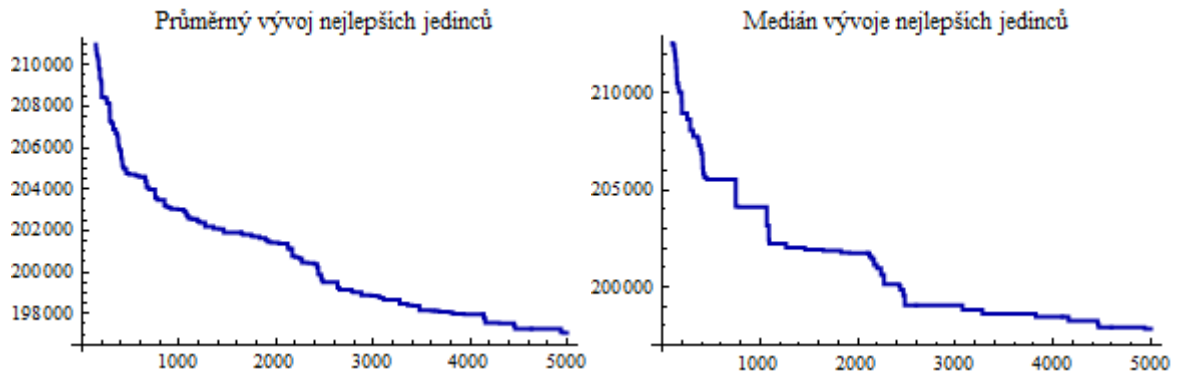


Obr. 69. Nejlepší jedinci

Naproti předchozímu případu už je zřejmé, že vývoj jedinců probíhá. Do nových populací je vpouštěno větší množství nově vytvořených a kvalitnějších jedinců.

Tab. 42. Hodnoty získané z řešení

Nejlepší jedinec (hodnota účelové funkce)	<b>189841</b>
Průměr nejlepších nalezených jedinců pro 30 opakování	197083
Medián nejlepších nalezených jedinců pro 30 opakování	197843
Směrodatná odchylka nejlepších jedinců	3084
Průměrný čas 1 opakování	4313,4s
Celkový čas testu	129402,0s

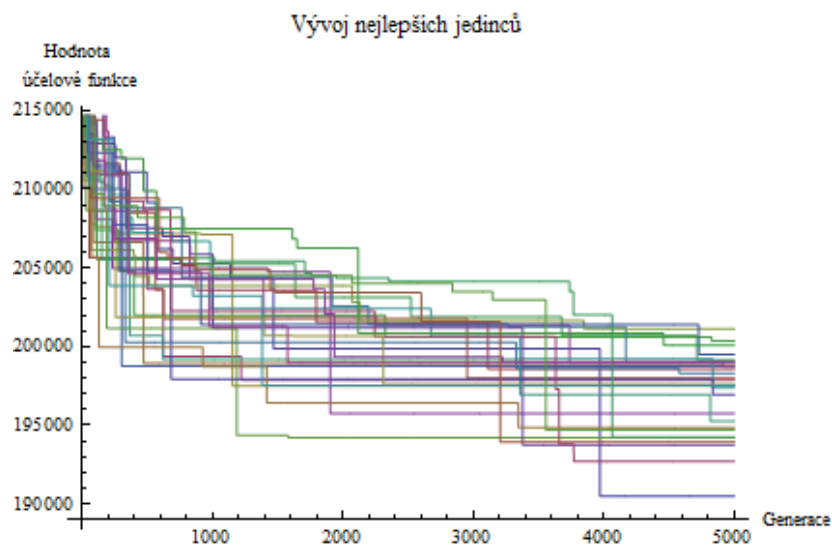


Obr. 70. Průměr a medián nejlepších nalezených jedinců

### 8.6.3 Repairment 0,7

Tab. 43. Nastavené parametry algoritmu pro velkou dimenzi a velký počet generací

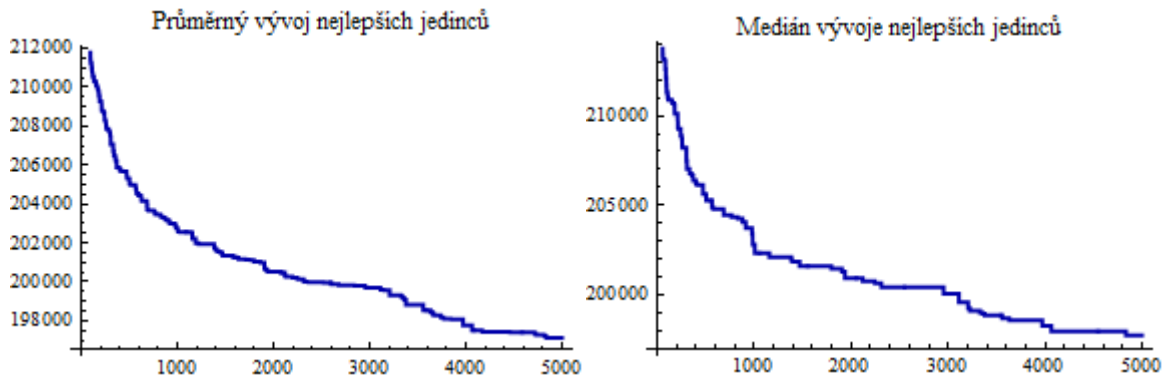
Parametr	Hodnota	Parametr	Hodnota	Parametr	Hodnota	Parametr	Hodnota
<b>Repairment</b>	<b>0,7</b>	<i>Dim</i>	100	<i>F</i>	0,8	<i>Gen</i>	5000
		<i>NP</i>	75	<i>CR</i>	0,5	<i>Repeat</i>	30



Obr. 71. Nejlepší jedinci

Tab. 44. Hodnoty získané z řešení

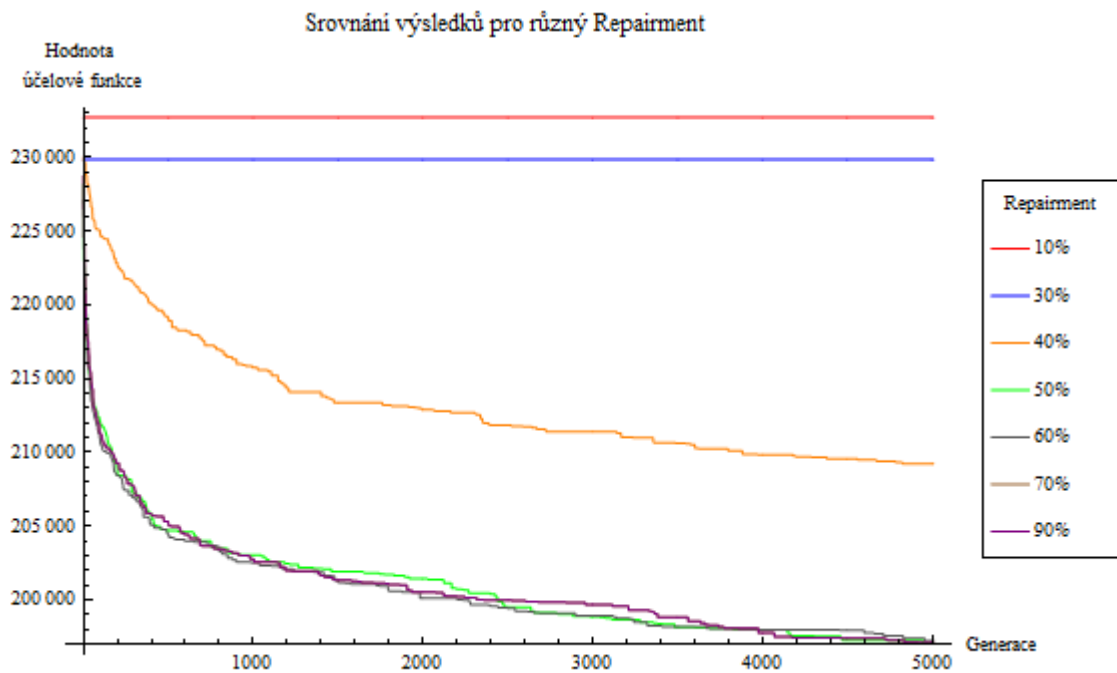
Nejlepší jedinec (hodnota účelové funkce)	<b>190497</b>
Průměr nejlepších nalezených jedinců pro 30 opakování	197109
Medián nejlepších nalezených jedinců pro 30 opakování	197742
Směrodatná odchylka nejlepších jedinců	2545
Průměrný čas 1 opakování	4742,9s
Celkový čas testu	142287,0s



Obr. 72. Průměr a medián nejlepších nalezených jedinců

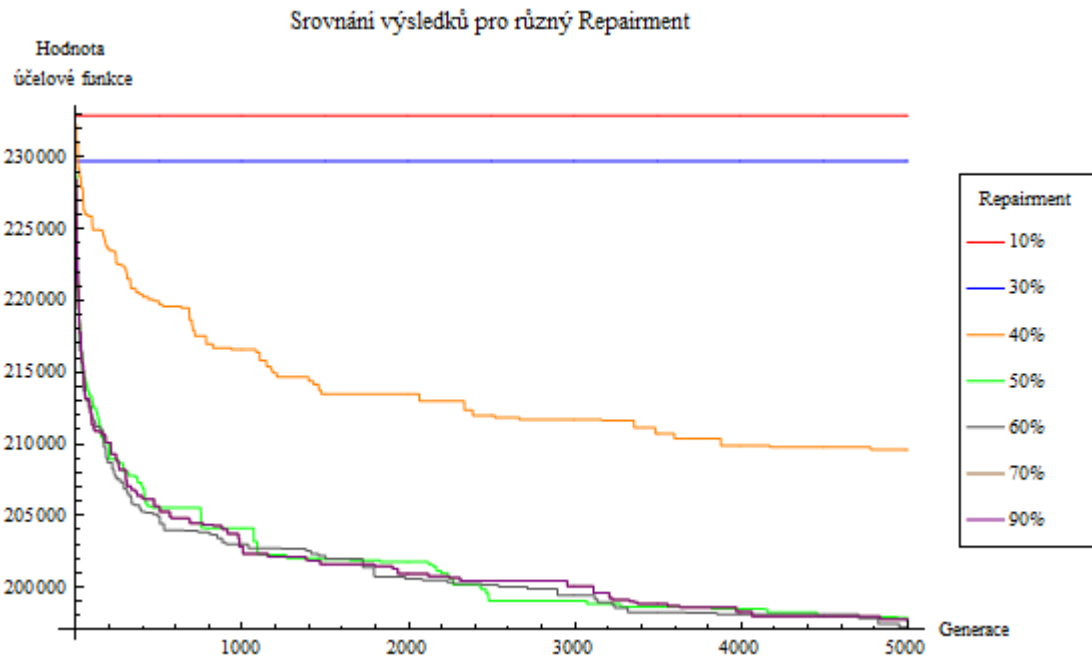
### 8.6.4 Porovnání průměrů a mediánů pro různou hranici oprav

**Průměr:**



Obr. 73. Srovnání průměrů pro různé hodnoty hranice oprav

Nejnižší průměrná hodnota účelové funkce nejlepších jedinců po 30 opakováních byla 197083 při hranici 60% (na grafu vykreslen šedou barvou).

**Medián:**

Obr. 74. Srovnání mediánů pro různé hodnoty hranice oprav

Nejnižší medián hodnot účelové funkce nejlepších jedinců po 30 opakováních byla 197174 při hranici 60% (na grafu vykreslen šedou barvou).

Z obrázků Obr. 73. a Obr. 74. je zřejmé, že pro hranici oprav 10% a 30% algoritmus stagnoval a vývoj téměř neprobíhal. Pomalý vývoj byl i při hranici oprav 40%, s mnohem vyšším počtem generací by však teoreticky do vhodného řešení dospěl. Pro vyšší hranice oprav už vývoj probíhal v pořádku a algoritmus směřoval do minima. Nejvhodnější řešení byla nalezena pro hranici oprav 60%.

## 9 KOMPLETNÍ SROVNÁNÍ VÝSLEDKŮ

V této kapitole jsou výsledky nalezených řešení (nejlepších jedinců, průměrů, mediánů a směrodatných odchylek nejlepších jedinců) všech důležitých testování pro různé nastavení parametrů. V každém řádku je vyznačeno nejlepší (zeleně) a nejhorší (červeně) nalezené řešení. Červeným rámečkem je pak vyznačena nejlepší hodnota pro konkrétní velikost dimenze. Tyto hodnoty jsou také vyjádřeny graficky.

### 9.1 Nejlepší jedinci

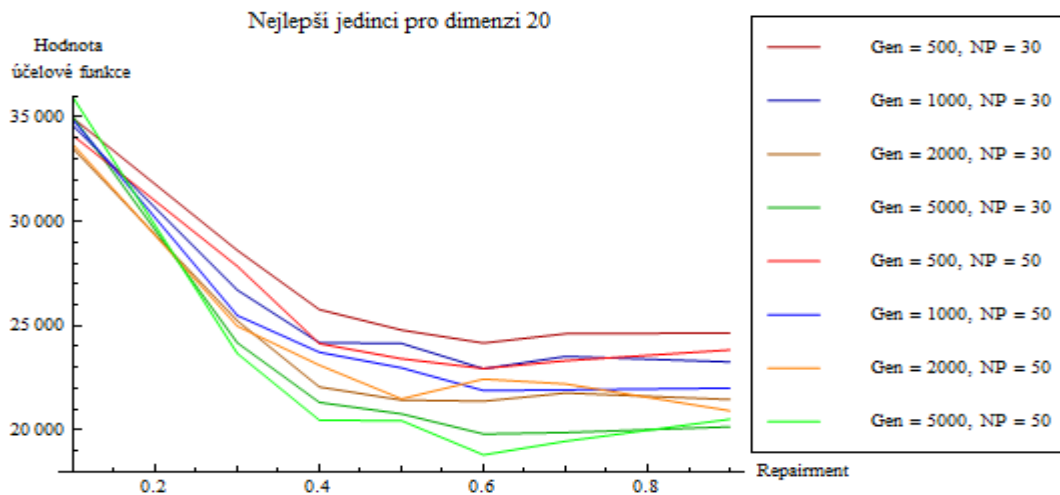
Tab. 45. Srovnání nejlepších jedinců pro všechny dimenze

Nejlepší nalezený jedinec									
Dimenze	Populace	Generace	Repairment						
			0,1	0,3	0,4	0,5	0,6	0,7	0,9
20	30	500	34942	28616	25774	24794	24173	24618	24655
		1000	34579	26713	24185	24154	22942	23532	23264
		2000	33481	25238	22067	21438	21387	21782	21464
		5000	34997	24184	21325	20777	19817	19882	20158
	50	500	34108	27853	24147	23423	22950	23325	23839
		1000	34850	25490	23730	22985	21898	21930	22016
		2000	33672	24982	23109	21498	22446	22210	20930
		5000	35925	23669	20477	20455	18812	19466	20525
50	30	500	101706	102900	88493	86536	86754	83394	83925
		1000	104612	103712	88396	82156	85725	82846	82619
		2000	103574	98581	88465	85348	84881	79750	82878
		5000	101331	99130	84330	80989	81097	80696	81914
	75	500	97420	98657	93777	87059	85161	84633	85467
		1000	104498	102215	89686	84733	82085	85284	76484
		2000	102587	98003	83471	77394	82743	83295	76178
		5000	102809	96334	83112	80647	79968	79125	82831
100	30	500	224896	220573	213604	201217	191001	197458	197997
		1000	208566	219553	211594	199162	199001	193181	195859
		2000	218479	215381	208705	196413	196423	197011	196954
		5000	225628	222298	206787	196232	191833	192108	188729
	75	500	217236	225016	208443	198248	196130	199856	196814
		5000	224929	219300	201180	189841	190224	190497	192348
Nejlepší Repairment - průměr:			0,668						
Nejlepší Repairment - medián			0,600						

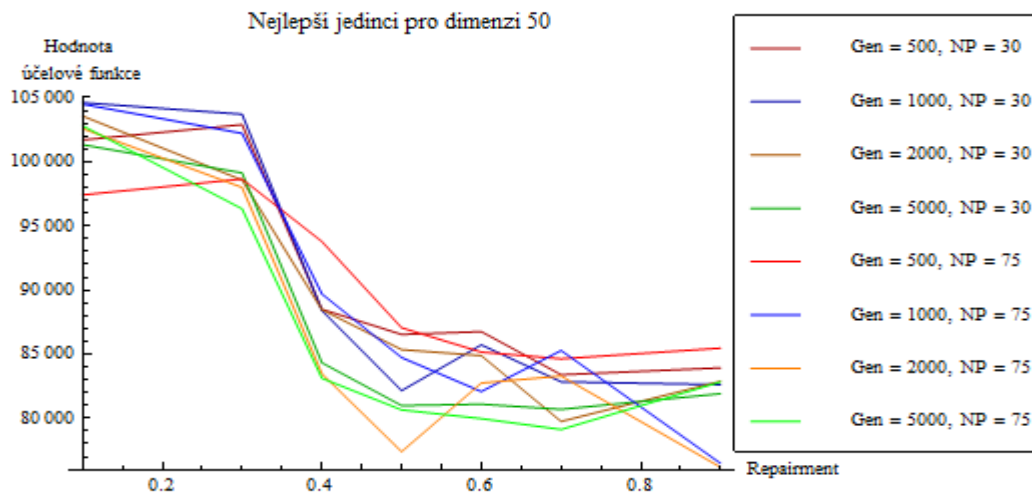
nejhorší řešení

nejlepší řešení

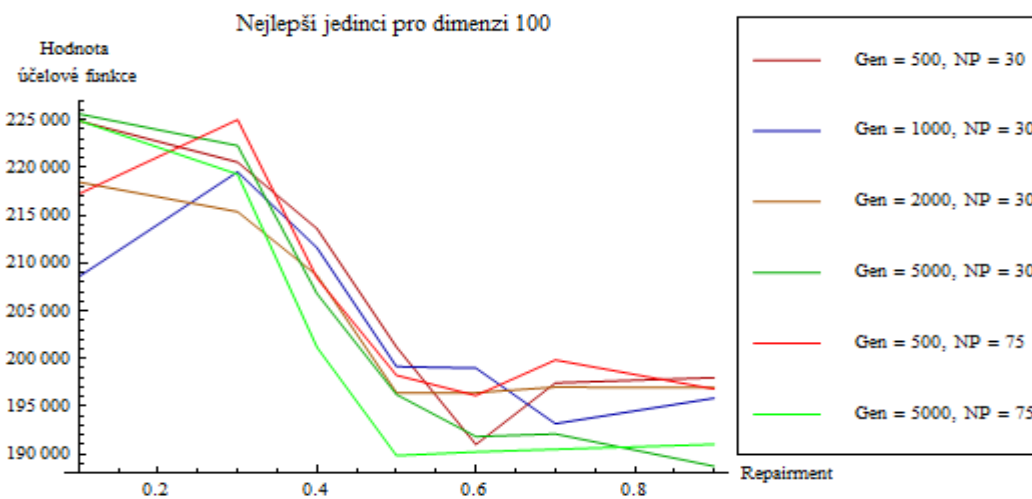
nejlepší řešení z celé dimenze



Obr. 75. Nejlepší nalezení jedinci pro dimenzi 20



Obr. 76. Nejlepší nalezení jedinci pro dimenzi 50



Obr. 77. Nejlepší nalezení jedinci pro dimenzi 100

Z obrázků Obr. 75. až Obr. 77. je zřejmé, že nejlepšího řešení bylo dosaženo při velkém počtu generací (zelené křivky). Jen mírně lepších výsledků bylo dosaženo při vyšší populaci, ovšem jednoznačně největší vliv na kvalitu řešení má počet generací, po které probíhá vývoj jedinců. Nejlepších výsledků bylo dosaženo mezi hodnotami *Repairment* 0,6 až 0,7.

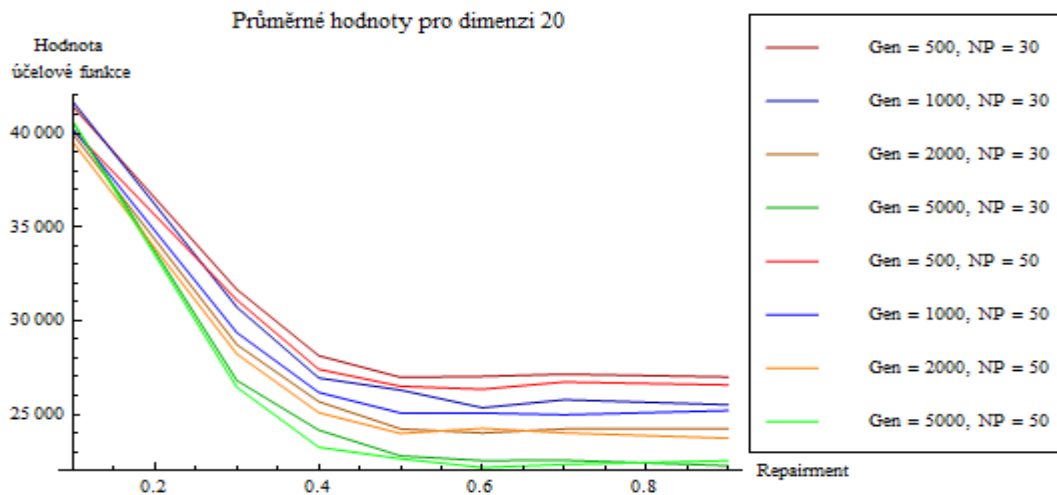
## 9.2 Průměrné hodnoty nejlepších jedinců

			Průměr						
Dimenze	Populace	Generace	Repairment						
			0,1	0,3	0,4	0,5	0,6	0,7	0,9
20	30	500	41417	31662	28128	26978	27027	27144	26993
		1000	41674	30720	26941	26298	25350	25782	25512
		2000	39943	28720	25677	24226	24004	24227	24229
		5000	40600	26814	24158	22784	22536	22550	22259
	50	500	40152	31109	27405	26496	26344	26728	26569
		1000	40221	29363	26170	25073	25076	24980	25206
		2000	39518	28221	25093	23982	24254	24009	23726
		5000	40560	26448	23246	22631	22167	22325	22529
50	30	500	113387	111750	97306	92968	92578	91822	92574
		1000	112139	110877	95630	90476	90882	89997	89777
		2000	113383	106322	93259	89372	88849	87834	88499
		5000	112718	105028	91248	86582	86066	86152	86500
	75	500	109894	107216	96465	90850	90764	90249	90479
		1000	109989	107448	93428	88978	88294	89018	88077
		2000	110335	104680	91796	86957	86744	86807	86695
		5000	110682	103566	89309	84960	84986	84297	85081
100	30	500	236852	233613	221165	208532	206908	207252	207334
		1000	234294	234150	217620	205817	204901	204386	205053
		2000	232793	234498	215890	202881	203002	202981	202927
		5000	237222	235950	212326	200466	199893	199291	198611
	75	500	232049	232323	219094	206064	205211	206337	205312
		1000	232683	229835	209224	197083	197239	197109	198026
		2000							
		5000							
Nejlepší Repairment - průměr:			0,695						
Nejlepší Repairment - medián			0,700						

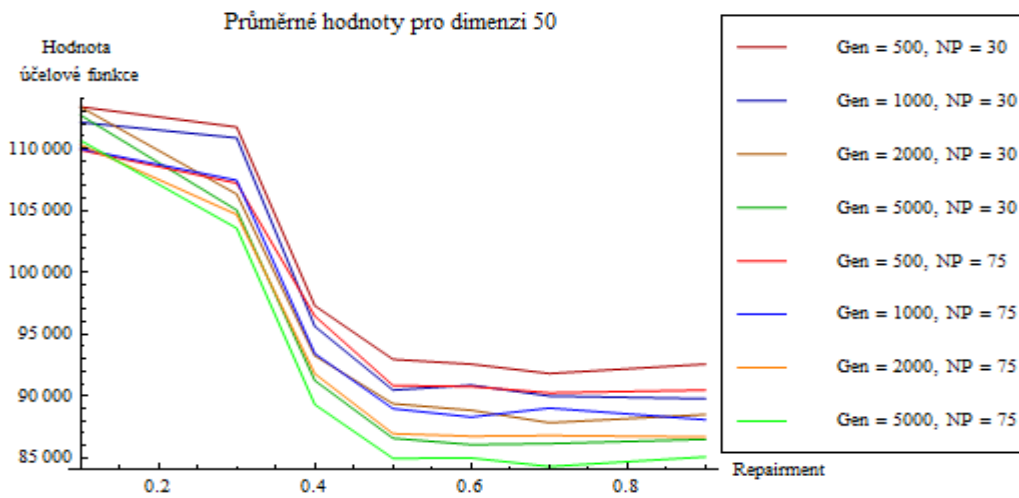
nejhorší průměr

nejlepší průměr

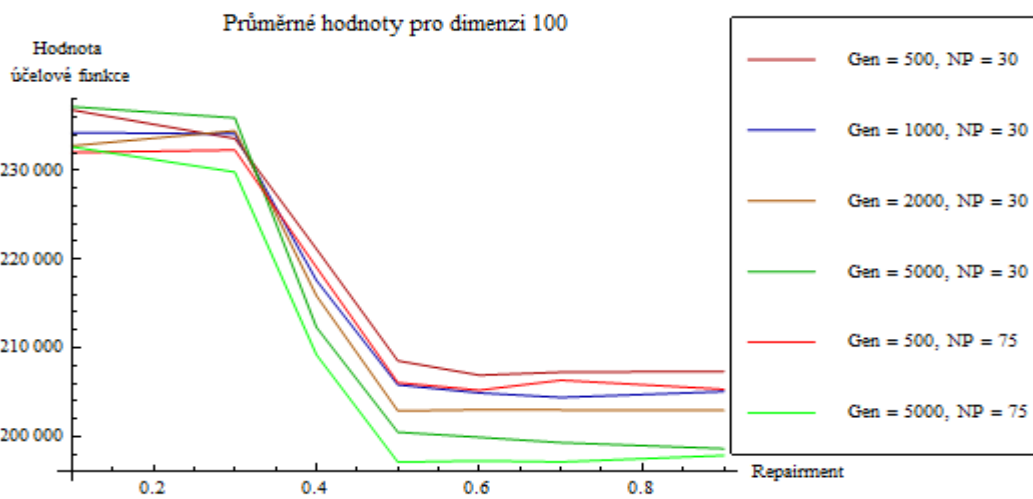
nejlepší průměr z celé dimenze



Obr. 78. Průměrné hodnoty nejlepších jedinců pro dimenzi 20



Obr. 79. Průměrné hodnoty nejlepších jedinců pro dimenzi 50



Obr. 80. Průměrné hodnoty nejlepších jedinců pro dimenzi 100

Z obrázků Obr. 78. až Obr. 80 je opět je jasné vidět, že nejlepších průměrů bylo dosaženo při velkém počtu generací. Pro malou dimenzi nehrála velikost populace téměř žádnou roli, se zvyšující se dimenzí bylo dosaženo mírně lepších výsledků. Opět lze vyvodit tvrzení, že kvalita diferenciální evoluce je dána především počtem generací. Nejlepších výsledků bylo dosaženo mezi hodnotami *Repairment* 0,6 až 0,7.

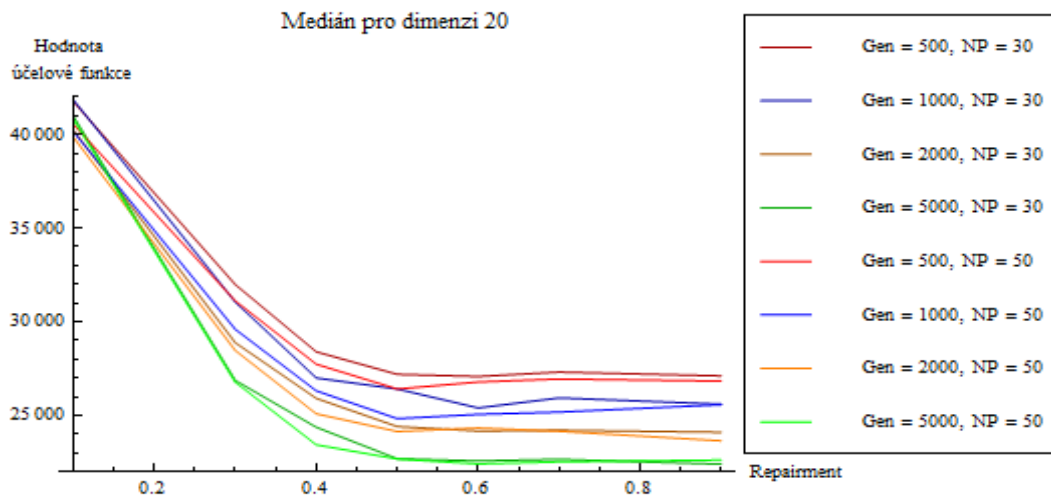
### 9.3 Medián nejlepších jedinců

			Medián						
Dimenze	Populace	Generace	Repairment						
			0,1	0,3	0,4	0,5	0,6	0,7	0,9
20	30	500	41772	31988	28398	27192	27074	27312	27108
		1000	41866	31054	26996	26410	25400	25932	25599
		2000	40245	28884	25911	24403	24161	24205	24096
		5000	40994	26864	24366	22683	22570	22650	22396
	50	500	40558	31114	27730	26411	26782	26946	26833
		1000	40169	29596	26310	24826	25052	25172	25572
		2000	39874	28468	25084	24142	24324	24150	23638
		5000	40836	26778	23432	22673	22404	22520	22616
50	30	500	114065	112108	98107	93324	92826	92245	93406
		1000	112214	111006	96031	90647	91110	90122	89396
		2000	113983	107112	93674	89418	89046	88664	89030
		5000	112780	105022	91209	87288	86054	86744	86841
	75	500	110704	106698	96524	91006	90981	90535	90506
		1000	110688	107794	93879	88614	88476	88882	88364
		2000	110654	105218	92390	87438	86974	86762	87351
		5000	111029	103446	90034	85252	84912	84464	85184
100	30	500	238110	234018	221412	208891	207036	207646	207780
		1000	234520	234424	217713	205896	204682	205573	204941
		2000	233598	235848	215744	203527	203622	203753	203846
		5000	237246	235976	212375	200680	200384	199531	199118
	75	500	232072	233180	219848	206022	205976	206484	205109
		1000	232914	229740	209592	197843	197174	197742	197680
		2000							
		5000							
Nejlepší Repairment - průměr:			0,705						
Nejlepší Repairment - medián			0,700						

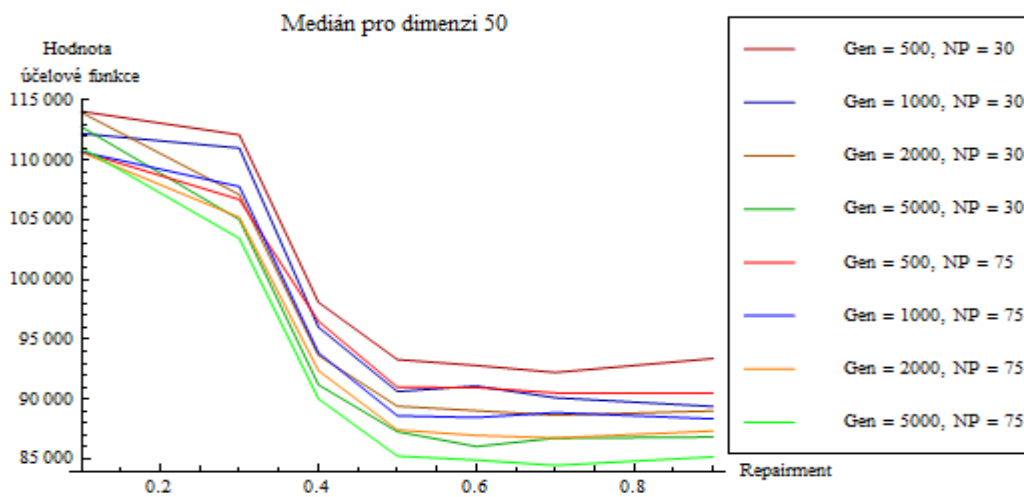
nejhorší medián

nejlepší medián

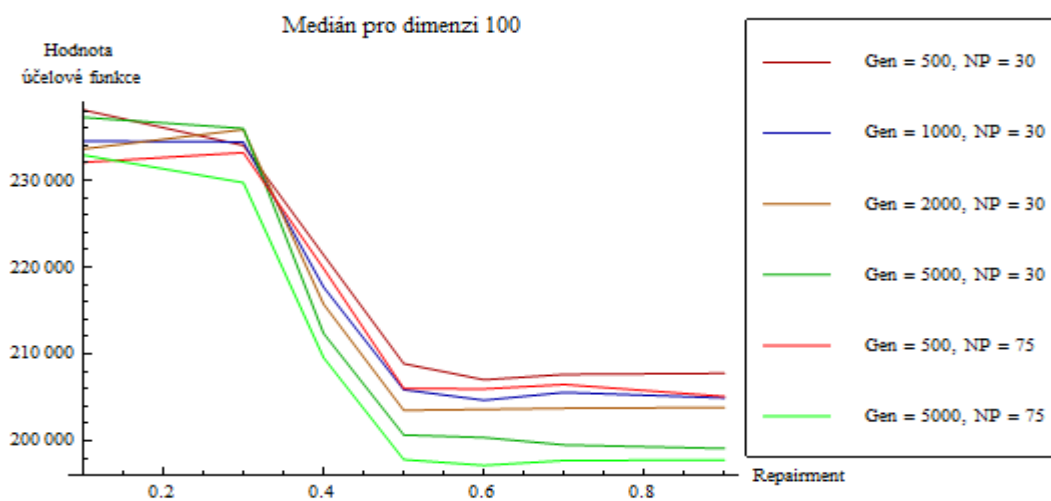
nejlepší medián z celé dimenze



Obr. 81. Mediány nejlepších jedinců pro dimenzi 20



Obr. 82. Mediány nejlepších jedinců pro dimenzi 50



Obr. 83. Mediány nejlepších jedinců pro dimenzi 100

Stejně jako v grafickém vyjádření u průměrů, i u mediánů je zřejmé, že nejlepších řešení bylo dosaženo při velkém počtu generací. Velikost populace nehrála příliš velký vliv u nízkých dimenzí, malou měrou byly ovlivněny větší dimenze. Nejlepších výsledků bylo dosaženo v okolí hodnoty *Repairment* 0,7.

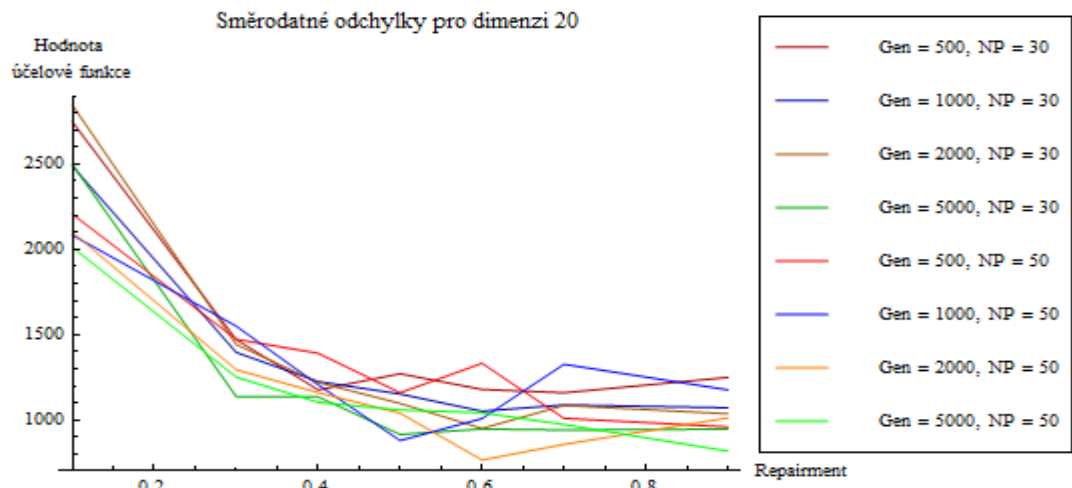
#### 9.4 Směrodatné odchytky nejlepších jedinců

Směrodatná odchytky									
Dimenze	Populace	Generace	Repairment						
			0,1	0,3	0,4	0,5	0,6	0,7	0,9
20	30	500	2749	1471	1176	1271	1179	1159	1249
		1000	2489	1396	1225	1152	1051	1088	1071
		2000	2847	1442	1220	1097	950	1085	1036
		5000	2499	1137	1135	915	947	941	946
	50	500	2207	1473	1391	1159	1332	1009	959
		1000	2084	1550	1206	879	1008	1326	1176
		2000	2101	1294	1160	1040	765	857	1012
		5000	2013	1251	1103	1058	1042	973	817
50	30	500	4157	4294	3094	2115	2227	2335	2601
		1000	3661	3362	2772	2509	1980	2725	2372
		2000	4003	3433	2726	1916	1969	2923	2236
		5000	3933	2647	2679	2022	1830	2110	1829
	75	500	3800	3894	1481	1813	2057	2286	2121
		1000	3088	2707	1678	1979	2387	1858	2897
		2000	3134	2916	2135	2525	2003	1817	2583
		5000	2959	2819	2607	1531	2058	2004	1272
100	30	500	4957	6028	3925	3016	4708	3646	3666
		1000	6033	5882	3460	2627	2865	3896	3090
		2000	5858	6749	3449	3474	2895	3041	3127
		5000	4653	5933	2554	2413	2867	3103	3182
	75	500	5130	4159	3981	2889	3686	2657	3687
		1000	4057	4765	2852	3084	2810	2545	2593
		2000							
		5000							
Nejlepší Repairment - průměr:			0,623						
Nejlepší Repairment - medián			0,600						

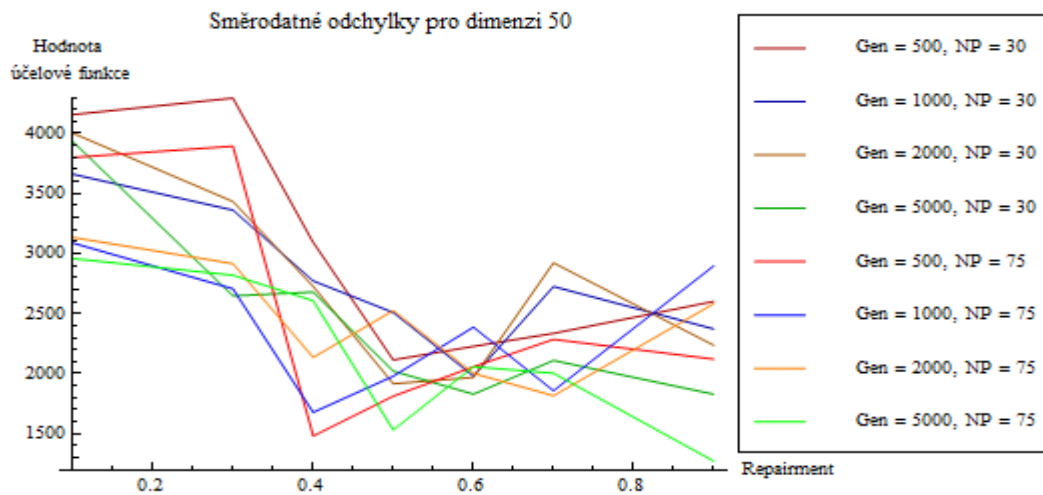
největší sm. odchylka

nejmenší sm. odchylka

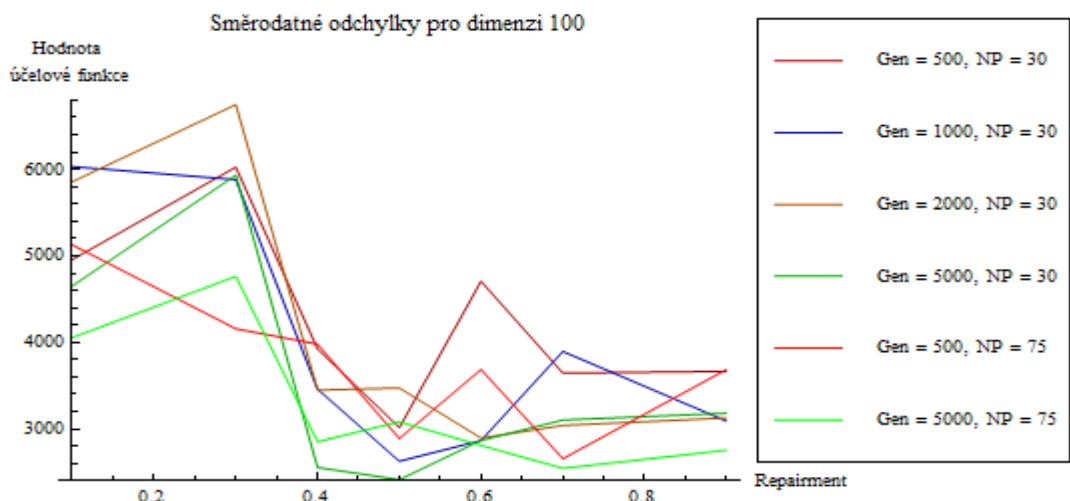
nejmenší sm. odch. z celé dimenze



Obr. 84. Směrodatné odchylky pro dimenzi 20



Obr. 85. Směrodatné odchylky pro dimenzi 50



Obr. 86. Směrodatné odchylky pro dimenzi 100

Z obrázků Obr. 84 až Obr. 86. lze usoudit, že počet generací ani velikost populace nemají příliš výrazný vliv na hodnotu směrodatné odchylky, tedy na vzdálenost hodnot účelové funkce nejlepších jedinců od sebe. Nejlepších výsledků bylo dosaženo v okolí hodnoty *Repairment* 0,6, ale z grafického vyjádření je zřejmé, že hodnota směrodatné odchylky "skákala" z nižších hodnot na vyšší a opačně. Velké směrodatné odchylky byly pouze u nízkých hodnot *Repairment*, jelikož v těchto hodnotách algoritmus často stagnoval nebo se vyvíjel velmi pomalu, tím pádem byla směrodatná odchylka hodně ovlivněna prvotní vygenerovanou populací a rozmanitostí jedinců v ní, jejichž vývoj dál probíhal jen velmi pomalu.

Řešení problému obchodního cestujícího nebylo možno z časových důvodů testovat pro ještě více generací nebo pro větší populace. Ze získaných hodnot lze říci, že řešení nejvíce ovlivňuje počet generací, ve kterých dochází k vývoji jedinců. Teoreticky by tedy pro ještě větší počet generací než 5000 bylo řešení ještě kvalitnější. Takové řešení je však špatně proveditelné v domácím prostředí a pro spolehlivé testování by bylo třeba spustit tento algoritmus na PC, které má záložní zdroj, pro případ výpadku elektrické energie, a může běžet i několik týdnů, případně měsíců bez omezení.

## ZÁVĚR

Cílem práce bylo porovnat možnosti různého nastavení parametru pro hranici oprav při průběhu diskrétní diferenciální evoluce a také porovnat výhodnost použití diferenciální evoluce při řešení optimalizačních úloh. Byla naprogramována diskrétní verze diferenciální evoluce a následně testována na optimalizační úloze problém obchodního cestujícího.

Celkem bylo provedeno 169 řešení pro různé nastavení vstupních parametrů, které zabralo přes 570 hodin čistého času na dvou PC.

Problém obchodního cestujícího byl vyřešen hrubou silou pro malé množství měst a porovnán s řešením pomocí diferenciální evoluce. Porovnání jasně ukázalo výhodnost použití diferenciální evoluce, kdy dosáhla naprosto shodných výsledků za mnohonásobně kratší čas.

Testování hranice oprav ukázalo, že jako nejvhodnější se jeví zvolit tuto hranici mezi 60% až 70%, kdy algoritmus dosahoval nejlepších výsledků. To znamená, že evoluce v tomto případě byla zachována ze 30% až 40%, zbytek byl věcí náhodného generování parametrů jedinců.

Z porovnání jednotlivých řešení je zřejmé, že algoritmus diferenciální evoluce není příliš závislý na velikosti populace, ale důležitou roli hraje počet generací, po které se populace jedinců postupně vyvíjí. Při dostatečném počtu generací by algoritmus teoreticky dosáhl až celkového minima. Pro větší počet dimenzí však nelze zjistit přesnou hodnotu globálního minima jakýmkoliv deterministickým způsobem.

Veškeré provedené testy jsou přiloženy v noteboocích softwaru Wolfram Mathematica na přiloženém CD.

## CONCLUSION

The goal of the thesis was to compare the different options for setting repairment boundary for discrete differential evolution and compare the advantages of usage of the differential evolution for solving optimization problems. The discrete version of the differential evolution was programmed and then tested on optimizing issue the traveling salesman problem.

Overall 169 solutions for various settings of the input parameters was tested, which took over 570 hours on two PCs.

The traveling salesman problem was solved by brute-force attack for a small number of cities and then compared with the differential evolution solution. This comparison clearly showed the advantages of using differential evolution. It achieved exactly the same results for much less time.

Repairment boundary testing was showed that the most suitable is to choose this level between 60% and 70%, when the algorithm achieved the best results. The evolution in this case was keeping from 30% to 40%, the rest was just a random generation of the parameters.

From the comparison of the solutions it's obvious that the differential evolution algorithm doesn't depend on population size, but more important is the the number of generations. When the number of generations is suffiscied the algorithm theoretically achieve the global minimum. But for more dimensions it's not possible to determine the exact value of the global minimum in any deterministic way.

There are all performed tests in the notebooks of the software Wolfram Mathematica on the attached CD.

**SEZNAM POUŽITÉ LITERATURY**

- [1] ZELINKA, Ivan. *Umělá inteligence v problémech globální optimalizace*. BEN, 2002, 190 s. ISBN 80-7300-069-5.
- [2] ZELINKA, Ivan. *Evoluční výpočetní techniky: principy a aplikace*. 1. vyd. Praha: BEN - technická literatura, 2009, 534 s. ISBN 978-80-7300-218-3.
- [3] DE JONG, Kenneth A. *Evolutionary computation: a unified approach*. Cambridge: MIT Press, 2006, ix, 256 s. ISBN 02-620-4194-4.
- [4] MAŘÍK, V., ŠTĚPÁNKOVÁ, O., LAŽANSKÝ, J.: *Umělá inteligence*, Academia, 1993, ISBN 80-200-0496-3.
- [5] MAŘÍK, V., ŠTĚPÁNKOVÁ, O., LAŽANSKÝ, J.: *Umělá inteligence 4.*, Academia, 2003, ISBN 80-200-1044-0.
- [6] ZELINKA, Ivan, Zuzana OPLATKOVÁ a Roman ŠENKEŘÍK. *Aplikace umělé inteligence*. Vyd. 1. Zlín: Univerzita Tomáše Bati ve Zlíně, 2010, 151 s. ISBN 978-80-7318-898-6.
- [7] PRICE, Kenneth V, Rainer M STORN a Jouni A LAMPINEN. *Differential evolution: a practical approach to global optimization* [online]. Berlin: Springer, 2005.
- [8] *Handbook of Optimization: From Classical to Modern Approach*. 1. vyd. Editor Ivan Zelinka, Václav Snášel, Ajith Abraham. Berlin: Springer, 2013, xii, 1100 s. Intelligent systems reference library, 38. ISBN 978-3-642-30503-0.
- [9] CHRAMCOV, Bronislav. *Základy práce v prostředí Mathematica*. Vyd. 2. Ve Zlíně: Univerzita Tomáše Bati, 2006. 122 s. ISBN 80-731-8510-5.
- [10] Wolfram Mathematica. WOLFRAM. *Wolfram Mathematica: Technical Computing Software*[online]. [cit. 2013-05-06]. Dostupné z: <http://www.wolfram.com/mathematica/>

**SEZNAM POUŽITÝCH SYMBOLŮ A ZKRATEK**

CR	Crossover - Práh křížení.
DE	Diferenciální evoluce.
Dim	Dimenze.
EA	Evoluční algoritmy.
EVT	Evoluční výpočetní techniky.
F	Mutační konstanta.
GA	Genetické algoritmy.
Gen	Generace.
NP	Velikost populace.
NFL	No Free Lunch Theorem.
TSP	Traveling salesman problem - Problém obchodního cestujícího.

## SEZNAM OBRÁZKŮ

Obr. 1. No Free Lunch Teorém v grafickém vyjádření .....	13
Obr. 2. Příklad jednoho generačního cyklu diferenciální evoluce.....	20
Obr. 3. První de Jongova funkce ve 2D s vyznačeným extrémem .....	23
Obr. 4. První de Jongova funkce s vyznačeným globálním minimem .....	23
Obr. 5. Třetí de Jongova funkce ve 2D s vyznačeným extrémem .....	24
Obr. 6. Třetí de Jongova funkce s vyznačeným globálním minimem .....	24
Obr. 7. Rossenbrockovo sedlo ve 2D s vyznačeným extrémem.....	25
Obr. 8. Rossenbrockovo sedlo s vyznačeným globálním minimem.....	25
Obr. 9. Schwefelova funkce ve 2D s vyznačenými cyklicky se střídajícími extrémy okolo počátku .....	26
Obr. 10. Schwefelova funkce s vyznačeným minimem.....	27
Obr. 11. Rastriginova funkce ve 2D s vyznačeným extrémem .....	27
Obr. 12. Rastriginova funkce s vyznačeným globálním minimem.....	28
Obr. 13. Struktura notebooku v Mathematica 8.....	30
Obr. 14. Náповěda pro funkci Sum a její možné vstupní parametry.....	31
Obr. 15. Grafické znázornění průběhu oprav .....	34
Obr. 16. Nejkratší a nejdelší nalezená trasa pro 5 měst.....	36
Obr. 17. Nejkratší a nejdelší nalezená trasa pro 6 měst.....	37
Obr. 18. Nejkratší a nejdelší nalezená trasa pro 7 měst.....	37
Obr. 19. Nejkratší a nejdelší nalezená trasa pro 8 měst.....	38
Obr. 20. Nejkratší a nejdelší nalezená trasa pro 9 měst pomocí diferenciální evoluce .....	39
Obr. 21. Srovnání času výpočtu pro řešení hrubou silou a diferenciální evoluci .....	40
Obr. 22. Nejlepší jedinci .....	41
Obr. 23. Průměr a medián nejlepších nalezených jedinců.....	42
Obr. 24. Nejlepší jedinci .....	43
Obr. 25. Nejkratší nalezená trasa .....	43
Obr. 26. Průměr a medián nejlepších nalezených jedinců.....	44
Obr. 27. Nejlepší jedinci .....	45
Obr. 28. Průměr a medián nejlepších nalezených jedinců.....	45
Obr. 29. Srovnání průměrů pro různé hodnoty hranice oprav .....	46
Obr. 30. Srovnání mediánů pro různé hodnoty hranice oprav .....	46
Obr. 31. Stagnace při nízké hodnotě <i>Repairment</i> .....	47

Obr. 32. Nejlepší jedinci .....	48
Obr. 33. Průměr a medián nejlepších nalezených jedinců .....	49
Obr. 34. Nejlepší jedinci .....	50
Obr. 35. Nejkratší nalezená trasa pomocí algoritmu DE .....	50
Obr. 36. Průměr a medián nejlepších nalezených jedinců .....	51
Obr. 37. Nejlepší jedinci .....	52
Obr. 38. Nejkratší nalezená trasa .....	52
Obr. 39. Průměr a medián nejlepších nalezených jedinců .....	53
Obr. 40. Srovnání průměrů pro různé hodnoty hranice oprav .....	54
Obr. 41. Srovnání mediánů pro různé hodnoty hranice oprav .....	54
Obr. 42. Nejlepší jedinci .....	55
Obr. 43. Průměr a medián nejlepších nalezených jedinců .....	56
Obr. 44. Nejlepší jedinci .....	57
Obr. 45. Nejkratší nalezená trasa .....	57
Obr. 46. Průměr a medián nejlepších nalezených jedinců .....	58
Obr. 47. Nejlepší jedinci .....	59
Obr. 48. Průměr a medián nejlepších nalezených jedinců .....	59
Obr. 49. Srovnání průměrů pro různé hodnoty hranice oprav .....	60
Obr. 50. Srovnání mediánů pro různé hodnoty hranice oprav .....	60
Obr. 51. Nejlepší jedinci .....	61
Obr. 52. Průměr a medián nejlepších nalezených jedinců .....	62
Obr. 53. Nejlepší jedinci .....	63
Obr. 54. Průměr a medián nejlepších nalezených jedinců .....	64
Obr. 55. Nejlepší jedinci .....	64
Obr. 56. Průměr a medián nejlepších nalezených jedinců .....	65
Obr. 57. Srovnání průměrů pro různé hodnoty hranice oprav .....	66
Obr. 58. Srovnání mediánů pro různé hodnoty hranice oprav .....	66
Obr. 59. Nejlepší jedinci .....	67
Obr. 60. Průměr a medián nejlepších nalezených jedinců .....	68
Obr. 61. Nejlepší jedinci .....	69
Obr. 62. Průměr a medián nejlepších nalezených jedinců .....	69
Obr. 63. Nejlepší jedinci .....	70
Obr. 64. Průměr a medián nejlepších nalezených jedinců .....	71
Obr. 65. Srovnání průměrů pro různé hodnoty hranice oprav .....	71

Obr. 66. Srovnání mediánů pro různé hodnoty hranice oprav .....	72
Obr. 67. Nejlepší jedinci .....	73
Obr. 68. Průměr a medián nejlepších nalezených jedinců .....	73
Obr. 69. Nejlepší jedinci .....	74
Obr. 70. Průměr a medián nejlepších nalezených jedinců .....	75
Obr. 71. Nejlepší jedinci .....	75
Obr. 72. Průměr a medián nejlepších nalezených jedinců .....	76
Obr. 73. Srovnání průměrů pro různé hodnoty hranice oprav .....	76
Obr. 74. Srovnání mediánů pro různé hodnoty hranice oprav .....	77
Obr. 75. Nejlepší nalezení jedinci pro dimenzi 20 .....	79
Obr. 76. Nejlepší nalezení jedinci pro dimenzi 50 .....	79
Obr. 77. Nejlepší nalezení jedinci pro dimenzi 100 .....	79
Obr. 78. Průměrné hodnoty nejlepších jedinců pro dimenzi 20 .....	81
Obr. 79. Průměrné hodnoty nejlepších jedinců pro dimenzi 50 .....	81
Obr. 80. Průměrné hodnoty nejlepších jedinců pro dimenzi 100 .....	81
Obr. 81. Mediány nejlepších jedinců pro dimenzi 20 .....	83
Obr. 82. Mediány nejlepších jedinců pro dimenzi 50 .....	83
Obr. 83. Mediány nejlepších jedinců pro dimenzi 100 .....	83
Obr. 84. Směrodatné odchylky pro dimenzi 20 .....	85
Obr. 85. Směrodatné odchylky pro dimenzi 50 .....	85
Obr. 86. Směrodatné odchylky pro dimenzi 100 .....	85

**SEZNAM TABULEK**

Tab. 1. Nastavení parametrů pro testování .....	33
Tab. 2. Počet permutací pro 1 až 15 měst.....	35
Tab. 3. Porovnání nalezených výsledků .....	36
Tab. 4. Porovnání nalezených výsledků .....	36
Tab. 5. Porovnání nalezených výsledků .....	37
Tab. 6. Porovnání nalezených výsledků .....	38
Tab. 7. Porovnání nalezených výsledků .....	38
Tab. 8. Doba trvání algoritmu.....	39
Tab. 9. Nastavené parametry algoritmu pro malou dimenzi a nízký počet generací.....	41
Tab. 10. Hodnoty získané z řešení .....	42
Tab. 11. Nastavené parametry algoritmu pro malou dimenzi a nízký počet generací.....	42
Tab. 12. Hodnoty získané z řešení .....	43
Tab. 13. Nastavené parametry algoritmu pro malou dimenzi a nízký počet generací.....	44
Tab. 14. Hodnoty získané z řešení .....	45
Tab. 15. Nastavené parametry algoritmu pro malou dimenzi a vysoký počet generací .....	48
Tab. 16. Hodnoty získané z řešení .....	49
Tab. 17. Nastavené parametry algoritmu pro malou dimenzi a vysoký počet generací .....	50
Tab. 18. Hodnoty získané z řešení .....	51
Tab. 19. Nastavené parametry algoritmu pro malou dimenzi a vysoký počet generací .....	51
Tab. 20. Hodnoty získané z řešení .....	53
Tab. 21. Nastavené parametry algoritmu pro střední dimenzi a nízký počet generací.....	55
Tab. 22. Hodnoty získané z řešení .....	56
Tab. 23. Nastavené parametry algoritmu pro střední dimenzi a nízký počet generací.....	57
Tab. 24. Hodnoty získané z řešení .....	58
Tab. 25. Nastavené parametry algoritmu pro střední dimenzi a nízký počet generací.....	58
Tab. 26. Hodnoty získané z řešení .....	59
Tab. 27. Nastavené parametry algoritmu pro střední dimenzi a vysoký počet generací .....	61
Tab. 28. Hodnoty získané z řešení .....	62
Tab. 29. Nastavené parametry algoritmu pro střední dimenzi a vysoký počet generací .....	63
Tab. 30. Hodnoty získané z řešení .....	63

Tab. 31. Nastavené parametry algoritmu pro střední dimenzi a vysoký počet generací .....	64
Tab. 32. Hodnoty získané z řešení .....	65
Tab. 33. Nastavené parametry algoritmu pro velkou dimenzi a malý počet generací.....	67
Tab. 34. Hodnoty získané z řešení .....	68
Tab. 35. Nastavené parametry algoritmu pro velkou dimenzi a malý počet generací.....	68
Tab. 36. Hodnoty získané z řešení .....	69
Tab. 37. Nastavené parametry algoritmu pro velkou dimenzi a malý počet generací.....	70
Tab. 38. Hodnoty získané z řešení .....	70
Tab. 39. Nastavené parametry algoritmu pro velkou dimenzi a velký počet generací.....	72
Tab. 40. Hodnoty získané z řešení .....	73
Tab. 41. Nastavené parametry algoritmu pro velkou dimenzi a velký počet generací.....	74
Tab. 42. Hodnoty získané z řešení .....	74
Tab. 43. Nastavené parametry algoritmu pro velkou dimenzi a velký počet generací.....	75
Tab. 44. Hodnoty získané z řešení .....	75
Tab. 45. Srovnání nejlepších jedinců pro všechny dimenze.....	78