FAKULTA INFORMAČNÍCH TECHNOLOGIÍ ČVUT V PRAZE

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

Název:	Technika ostrovů při generování molekul
Student:	Daniel Šulik
Vedoucí:	Mgr. Jan Starý, Ph.D.
Studijní program:	Informatika
Studijní obor:	Teoretická informatika
Katedra:	Katedra teoretické informatiky
Platnost zadání:	Do konce letního semestru 2018/19

Pokyny pro vypracování

1. Rozšiřte existující aplikaci (BP Jonatana Matějky) o další evoluční techniku ostrovů.
2. Rozšiřte dosavadní minimalistický přístup ke generovanání nových molekul o elementární chemickou inteligenci, totiž mocnost vazeb a možnost mutace i na jiných prvcích než vodících.
3. Novou funkcionalitu řádně otestuje na reálných datech, nejlépe nad stejnou databází molekul jako předchozí BP.
4. Novou funkcionalitu řádně zdokumentujte, tj. rozšiřte stavající manuálové stránky.
5. Změřte dopad provedených změn na rychlost výpočtu: o kolik se použitím nových postupů výpočet zpomalí?

Seznam odborné literatury

Sršeň, Štěpán. Distribuovaná infrastruktura pro virtuální screening molekul. Bakalářská práce. ČVUT v Praze, Fakulta informačních technologií, 2017.
Matějka, Jonatan. Genetické algoritmy pro generování molekul. Bakalářská práce. ČVUT v Praze, Fakulta informačních technologií, 2017.

doc. Ing. Jan Janoušek, Ph.D.
vedoucí katedry

doc. RNDr. Ing. Marcel Jiřina, Ph.D.
děkan

V Praze dne 5. ledna 2018

FAKULTA
INFORMAČNÍCH
TECHNOLOGIÍ
ČVUT V PRAZE

Bakalárska práca

Technika ostrovů při generování molekul

Daniel Šulik

Katedra teoretickej informatiky
Vedúci práce: Mgr. Ján Starý, Ph.D.

11. mája 2018

Poďakovanie

Za poctivý prístup, časté a užitočné rady, ale aj za venovaný čas chcem srdečne poďakovať Mgr. Jánovi Starému, Ph.D. Taktiež by som chcel poďakovať Matejovi Dzurovovi k jemnému úvodu do chémie, svojmu Petrovi Šulikovi za podporu a štýlistickú korekciu, a tiež správcom laboratória SAGElab za prístup ku výpočtovým technológiam, pomocou ktorých som mohol otestovať funkčnosť programu.

Prehlásenie

Prehlasujem, že som predloženú prácu vypracoval samostatne a že som uviedol všetky informačné zdroje v súlade s Metodickým pokynom o etickej príprave vysokoškolských záverečných prác.

Beriem na vedomie, že sa na moju prácu vzťahujú práva a povinnosti vyplývajúce zo zákona č. 121/2000 Sb., autorského zákona, v znení neskorších predpisov, a skutočnosť, že České vysoké učení technické v Praze má právo na uzavrenie licenčnej zmluvy o použití tejto práce ako školského diela podľa § 60 odst. 1 autorského zákona.

V Prahe 11. mája 2018 .....

Odkaz na túto prácu

Šulik, Daniel. *Technika ostrovů při generování molekul.* Bakalárska práca. Praha: České vysoké učení technické v Praze, Fakulta informačních technologií, 2018.

Abstrakt

Cieľom tejto práce je rozšíriť už existujúci program, ktorý slúži na generovanie molekúl podľa počiatočných parametrov a molekulárnej databázy poskytnutej užívateľom. Spomínaný program bude rozšírený o dalšie evolučné techniky z bežného života. Výsledky generované rozšíreným programom budú zanalýzované a porovnané s výsledkami pôvodného programu. Výsledkom je sada unixovských programov vykonávajúcich jednotlivé úkony genetických algoritmov. Prínosom tejto práce je poukázanie na využitie výpočtovej techniky a evolučných algoritmov na časovo náročné problémy, známe aj pod názvom NP problémy.

Klíčová slova Evolučný algoritmus, generovanie molekúl, virtuálny screening, graf molekuly, ORCA, OpenBabel

Abstract

The purpose of this work is to extend an existing program, which generates molecules defined by starting parameters and a molecular database provided by the user. The program will be extended by other evolution techniques, which may be spotted in daily life situations. The result is set of UNIX utilities performing different genetic techniques. The contribution of this work is to point out that usage of computing machines in co-operation with evolution algorithms can be used on time demanding problems, known as NP problems.

Keywords Evolution algorithms, molecul generation, virtual screening, molecular graph, ORCA, OpenBabel

Obsah

Zoznam obrázkov

Úvod

Už od polovice 20. storočia sa ľudia snažili zúžitkovať silu výpočtovej techniky a ako vidíme, dôsledky týchto snáh siahajú hlboko do každodenného života bežného smrteľníka.

So stále stúpajúcim výkonom, našťastie a bohužiaľ pre odporcov, pribúdajú aj ďalšie oblasti, kde sa môžu uplatniť. Jednou z týchto oblastí je aj výskum. Presnejšie oblasť farmácie na nové lieky, elektrotechniky na nové vodiče, batérie alebo v strojárstve na nové, pevnejšie materiály pre stroje. Na miesto náročného a zdĺhavého vývoja, experimentovania, skúmania, prepočítavania vzorcov a otestovania každej jednej zlúčeniny v laboratóriu, môžeme jednoducho nechať stroju vygenerovať, resp. prefiltrovať možné zlúčeniny, ktoré budú mať požadované vlastnosti. Výsledne vzorce zlúčenín vygenerované, resp. prefiltrované strojom sú ručne otestované, či náhodou nemajú iné nevhodné vlastnosti, ktoré by sa nehodili.

Spôsob, akým dostaneme zo stroja tie správne molekuly, tak tento algoritmus má názov vo farmácií virtuálny screening alebo preosievanie molekúl. Na vstupe očakáva databázu molekúl na skontrolovanie a nastavenia, podľa ktorých vyhodnocuje vstupnú databázu.

V prípade, že spojíme virtuálny screening molekúl a správny algoritmus na generovanie rôznorodých molekúl, na ktoré sa uplatní virtuálny screening, vznikne veľmi silný nástroj na vytváranie zlúčenín, pravdepodobne aplikovateľných v rôznych odvetviach.

Táto práca sa zameriava na vyššie spomenutý algoritmus na generovanie rôznorodých molekúl, nie na vytvorenie úplne nového algoritmu, ale rozšírenie už existujúceho programu, pomocou ďalších evolučných algoritmov. Presnejšie ich podmnožinou známou ako genetické algoritmy, ktoré budu simulovať životný cyklus molekúl v rôznom počte generácií, kde molekuly reprezentujú jedincov v jednej generácií. Dosiahneme tým väčšiu diverzifikáciu molekúl a zároveň rôzne výsledné molekuly.

Kapitola 1

Cieľ práce

Cieľom práce je nadviazať na prácu *Genetické algoritmy pro generování molekul* kolegu Jonatana Matějku. Rozšíriť už ním implementované genetické techniky a zistiť, či rozšírenie novými technikami dopomôže k lepším výsledkom než aké boli dosiahnuté v pôvodnej implementácii. použitím týchto techník sa môže dôjsť ku lepším výsledkom ako u výsledkov v pôvodnej implementácie. Taktiež ho rozšíriť o techniku ostrovov, kde bude prebiehať viacej generácií, na sebe nepriamo závislých. V poslednom rade pridať do programu elementárnu chémiu, aby sa mohli kontrolovať alebo generovať aj rozmanitejšie molekuly, než len molekuly náhodne vytvorené z počiatočnej databázy. Všetky spomenuté rozšírenia budú písane v jazyku C a prepojené pomocou shellovských skriptov tak, ako to bolo aj v pôvodnej práci. Výsledný program si musí zachovať preložiteľnosť medzi unixovskými systémami, byť jednoduchý a zrozumiteľný. Všetky rozšírenia budú náležite zdokumentované v štandardných manuálových stránkach.

Kapitola 2

Genetické algoritmy

Prapočiatok genetických algoritmov sa datuje ku 50-tym a 60-tym rokom 20. storočia, kedy skupina vedcov z oblasti informatiky študovala evolučné systémy s myšlienkou použiť ich v praxi na riešenie inžinierskych problémov[1]. Cieľom všetkých týchto evolučných systémov bolo, nechať jedincov reprezentujúcich riešenie nejakého problému, vyvíjať sa pomocou nástrojov inšpirovaných genetickou rozmanitosťou a prirodzenou selekciou v prírode.

Postupom rokov sa začínalo zaujímať čoraz viac vedcov o aplikovanie evolučných stratégií inšpirovaných prírodou do rôznych odvetví. Ingo Rechenberg a následne Hans-Paul Schwefel použili evolučné techniky na optimalizáciu parametrov profilu krídel. Ďalšia skupina vedcov[1], taktiež inšpirovaná evolučnými algoritmami, sa rozhodla pre ich použitie v oblasti strojového učenia. Bohužiaľ ich práca neupútala pozornosť v porovnaní s inými prácami založenými na evolučných algoritmoch, evolučnom programovaní a genetických algoritmoch.

Genetické algoritmy, skrátene GA, boli vytvorené Johnom Hollandom v 60-tych rokoch 20. storočia. Vznik GA nebola čisto jeho práca, ale aj jeho kolegov a študentov z Michiganskej univerzity v 60-tych a 70-tych rokoch. Na rozdiel od evolučných stratégií a evolučného programovania, Hollandov pôvodný plán nebol určený ku vytvoreniu algoritmu, ktorý dokáže riešiť iba jeden špecifický problém, ale pochopiť proces fungovania adaptácií v prírode tak, aby sa mohli používať v obore informatiky vo všeobecnosti.

2.1 Časti genetických algoritmov

Po zhrnutí svojich vedomostí nadobudnutých z analyzovania priebehu evolúcií v prírode, kde prežíva najsilnejší jedinec, sa Holland rozhodol prezentovať prvotné základy pre GA.

Najzákladnejšou časťou je jedinec. Jedinec je reprezentovaný ako chromozóm, napríklad binárny reťazec fixnej dĺžky. Jeho chromozóm, tiež známy ako genotyp, charakterizuje vlastnosti jedinca. Pomocou aplikovania fitnes funkcie na daného jedinca, teda na jeho chromozóm, získame fitnes hodnotu reprezentujúcu vhodnosť jeho vlastnosti.

[1]Boli to G. E. P. Box, G.J. Friedman, W.W. Bledsoe, H.J. Bremermann a ďalší vedci.

Každý GA potrebuje viac ako jedného jedinca, inak by tento algoritmus nefungoval tak, ako by mal. V prípade, že máme viac ako jedného jedinca, získavame populáciu. Podľa môjho názoru sa to už začína podobať reálnemu svetu, ale chýba tomu jedna dôležitá premenná, ktorou je čas. Čas neberie ohľady na nikoho, plynie bez zastavenia. Ak pridáme čas do zatiaľ definovaného algoritmu, dostaneme pojem generácia. V každej generácií sa populácia mení, pretože čas je neúprosný a plynie.

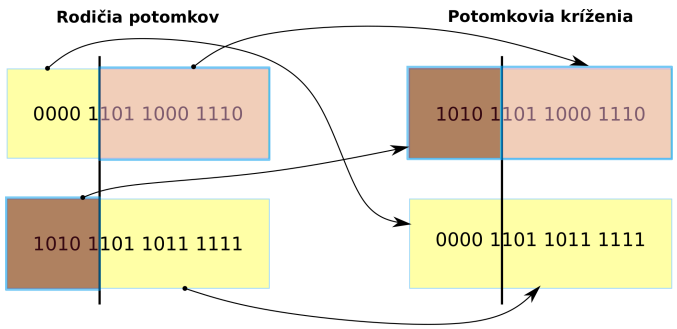
Zostáva už iba zadefinovať priebeh medzi generáciami a ten je realizovaný nasledovnými operátormi:

Selekcia Jeden zo základných operátorov GA, ktorý vyberá jedincov do kríženia pre ďalšiu generáciu. Existuje veľa druhov selekčných operátorov, ale zvolenie toho správneho závisí od úlohy a tiež subjektívneho postoja ku danému typu selekčného operátora.

Kríženie Operátor, ktorý sa stará o spájanie chromozómov dvoch alebo viacerých jedincov. Existuje viacero spôsobov, ako sa môže kríženie uskutočniť. Priebeh jednobodového kríženia začína vytvorením kópií jedincov získaných zo selekcie. Následne sa zväčša náhodne umiestní bod do ich chromozómov. Tento bod vyznačuje miesto, kde sa chromozóm jedinca rozdelí na dve časti. Takto rozdelené chromozómy jedincov sa v konečnej fáze spájajú s chromozómami druhých jedincov. Ukážka prezentujúca tento operátor sa nachádza na obrázku 2.1.

Mutácia Stará sa o zmenu jedného alebo viacerých bitov v chromozómoch jedinca, kde sa vyberú dané bity a tie sa invertujú.

Inverzia Dnes už nepoužívaný operátor, tiež známy ako operátor preusporiadania. Používal sa na zmenu poradia bitov v časti chromozómu. Tento operátor vznikol inšpirovaním sa z reálnej genetiky, kde nezáleží na zmene poradia génov v lokálnej časti. Pre vizuálny popis odporúčam si pozrieť obrázok 2.2.



Obr. 2.1: Obrázok znázornenia priebehu kríženia na dvoch jedincoch, ktorých genotyp je reprezentovaný binárnym reťazcom. Čiernou čiarou je znázornený rez cez dvoch jedincov. Následne sa prerezaný genotyp prvého jedinca skombinuje s genotypom druhého. Tak vzniknú nový dvaja potomkovia krížením. Tento typ kríženia sa nazýva jednobodové, pretože bolo použité iba jedno rozdelenie genotypu.

Obr. 2.2: Na obrázku je možné vidieť znázornenie fungovania inverzie u genotypu jedného jedinca. Na začiatku sa vyberie časť jedincovho genotypu. V nej sa uskutoční preusporiadanie poradia bitov. Následne takto preusporiadané bity vrátime do pôvodnej lokácie v genotype.

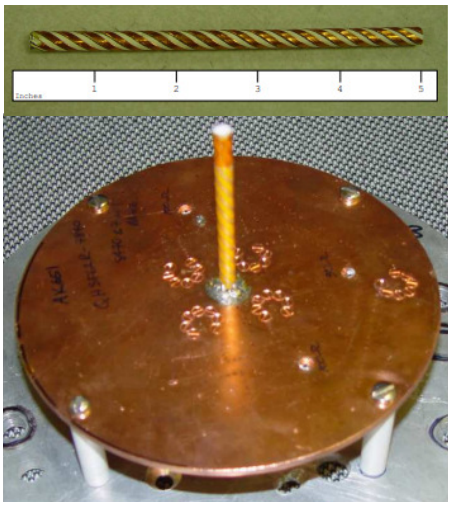
2.2 Vylepšenie antény pomocou GA

NASA[2] potrebovala stále lepšie a výkonnejšie zariadenia, aby dokázala navigovať a kontrolovať vesmírne projekty. Jedna z najznámejších aplikácií GA v praxi sa spája práve s NASA[2] a vesmírnym projektom Space Technology 5 (ďalej ST5). Potrebovali vyvinúť lepšiu anténu a skúsili použiť evolučné algoritmy (ďalej ako EA), či by im neprinieslo vylepšenie v technológiach. Presnejšie použili GA a genetické programovanie na vytvorenie dvoch nových antén, obrázok 2.4, ktoré by mohli nahradiť bežné, časovo a odborne náročné na výrobu antény, dizajnované práve človekom, obrázok 2.3.

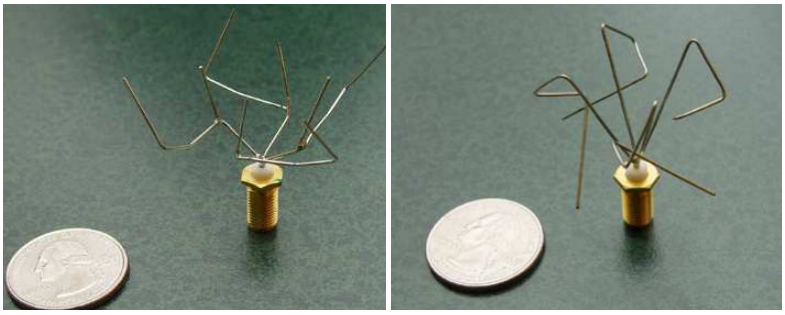
Výsledky evolučne generovaných antén dopadli úspechom. Pri analyzovaní antény ST5-3-10, 2.4 **a)**, sa zistilo, že disponuje viacerými vlastnosťami, ktoré bežne navrhnutá anténa nemá. Mala nasledujúce výhody:

- Anténa vytvorená pomocou EA má nižšiu spotrebu energie.
- Je to jednoduchšie vytvoriteľný prototyp.
- Má spoľahlivý výkon.
- Čas na návrh dizajnu bol zredukovaný z piatich na tri mesiace práce vykonávanej jedným človekom.

[2]National Aeronautics and Space Administration



Obr. 2.3: Anténa známa pod menom Quadrifilar Helix Antenna [2]. Bola vytvorená človekom podľa bežných postupov pre potreby NASA. Vytvorenie takejto antény je časovo náročné a zároveň si vyžaduje skúseného odborníka.



a) b)

Obr. 2.4: Vľavo sa nachádza prototyp antény ST5-3-10 [2]. Vytvorená bola pomocou genetického programovania, využívajúc techniku pridávania nových vetiev a ich rotovania. Anténa vpravo je prototyp ST5-4W-03 a bol vytvorená pomocou GA.

Kapitola 3

Analýza

Táto práca nadväzuje na prácu kolegu Jonatana Matějku, o ktorej sa zmieňujem už na začiatku. Tiež musím dodať, že nadväzujem nepriamo aj na prácu *Distribuovaná infrastruktura pro virtuální screening molekul* kolegu Štěpána Sršňa, keďže celý program kolegu Jonatana Matějku funguje správne za predpokladu, že bude fungovať aj práca Štěpána Sršňa. V nasledujúcich častiach sa pokúsim zovšeobecniť a popísať v jednoduchosti už vytvorenú prácu spomenutých kolegov.

3.1 Distribuovaná infrastruktura pro virtuální screening molekul

Práca kolegu Štěpána Sršňa sa zaoberá vytvorením programu `vms`, ktorý očakáva databázu molekúl s požiadavkou, podľa ktorej má ohodnotiť poskytnuté molekuly [3]. Molekuly očakávané na vstupe musia byť v správnom formáte a to je XYZ formát. Ako už z názvu môže intuitívne vyplývať, tak tento formát obsahuje výpis atómov molekuly v karteziánskom súradnicovom systéme [4].

Ohodnotenia z programu `vms` sú závislé na externom programe menom ORCA, tiež známym ako chemický kvantový program. Je to voľne dostupný program pre akademické účely. ORCA nielenže dokáže skontrolovať pomocou kvantových výpočtov správnosť poskytnutej molekuly, vyhodnocovať jej vlastnosti, ale aj vykonať geometrické optimalizácie a iné zaujímave techniky [5].

3.2 Genetické algoritmy pro generování molekul

Kolega Jonatan Matějka vytvoril program `mm` [6] na generovanie molekúl, využívajúci GA, ktorý používa mimobežné shellovské programy hlavne `vms` a OpenBabel [7]. `vms` sa používa na ohodnotenie a skontrolovanie poskytnutej molekuly. Na druhej strane sa OpenBabel používa na korekciu, opravu a zahashovanie poskytnutých molekúl. Spomenuté hashovanie sa využíva na ušetrenie výpočtových prostriedkov, aby sa zbytočne viackrát nevyhodnocovali tie isté molekuly.

Reprezentácia molekúl je realizovaná grafom. Vstupné molekuly sú pomocou nástroja OpenBabel osekané o atómy vodíka a výsledná molekula sa konvertuje do grafu. Atómy predstavujú uzly v grafe a väzby sú reprezentované pomocou hrán medzi uzlami [6].

Tento program na generovanie molekúl sa skladá z viacerých oddelených častí, preto si musia nejakým spôsobom zdielať molekuly. Robí sa to za pomocí ukladania molekúl do formátu SDF, keďže všetky oddelené časti sú schopné prijímať a produkovať práve tento formát molekuly [6]. Program `mm` sa skladá z nasledujúcich častí:

Inicializácia Začiatok programu `mm`, v ktorom sa inicializujú potrebné časti a vyhodnotí nultá generácia. Tá následne pokračuje na spracovanie GA.

Selekcia GA na výber jedincov z generácie. Využíva sa náhodný výber jedincov do turnaja. Veľkosť turnaja je špecifikovateľná užívateľom. Najlepší jedinec z turnaja vyhráva, postupuje ďalej do kríženia.

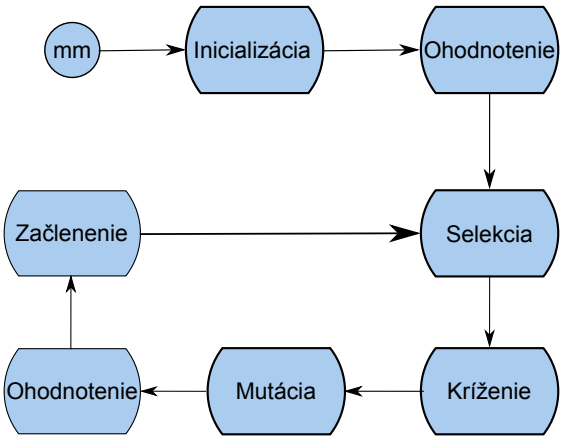
Kríženie GA na kríženie jedincov z vybranej generácie. V tomto krížení je dostupné jednobodové, ale aj viacbodové kríženie. Jedinou podmienkou pri krížení je nutnosť zachovania násobnosti väzieb medzi atómami, kde bola molekula prerezaná a zároveň sa musí zachovať typ atómu, ktorý bude naväzovať na prerezanú molekulu.

Mutácia GA na mutáciu jedincov pochádzajúcich z kríženia. Mutácia používa poskytnutú databázu fragmentov molekúl, ktoré sa v prípade uskutočnenia mutácie pripoja ku mutovanej molekule. Tiež môže nastať prípad, že molekula sa počas mutácie nezväčší o fragment, ale naopak zmenší. Nájde sa náhodný hranový rez, ktorý delí molekulu na dve časti, menšia časť sa zahodí a väčšia nahradí pôvodnú molekulu.

Začlenenie Podprogram zabezpečujúci vytvorenie novej nastávajúcej generácie z nových potomkov a predchádzajúcej generácie.

mm Skript riadiaci vyššie spomenuté komponenty, preto aj samotný program nesie názov `mm`.

Priebeh programu generovania molekúl začína spustením skriptu `mm`, ktorému pri štarte do parametrov zadá užívateľ voliteľné nastavenia pre beh programu. O zvyšnú prácu sa postará skript `mm`, ktorý koordinuje priebeh všetkých častí programu. Vizualizácia priebehu skriptu `mm` sa nachádza na obrázku 3.1.



Obr. 3.1: Zjednodušený diagram behu `mm` skriptu, popisující komponenty `mm`.

Kapitola 4

Návrh

Program `mm` obsahuje základné GA, ktoré fungujú podľa výsledkov zaujímavo dobre. Napriek tomuto faktu, som uvažoval, čo by stálo za to zmeniť, aby sa program posunul o úroveň vyššie. Limitovaný vedomosťami chémie som sa rozhodol upriamiť pozornosť na rozšírenie už existujúcich GA, ale aj pridanie nových tak, aby som minimálne pracoval s časťami založenými na základoch chémie, ale aby som využil najmä vedomosti z oboru informatiky. Po zvážení týchto bodov som došiel ku nasledujúcim možným zmenám.

4.1 Selekcia

Selekcia je silný nástroj, ktorý ovláda vývoj generácií. Pri použití slabej, neúčinnej selekčnej techniky, môže dôjsť ku spomaleniu vývoja generácií [1]. Existuje pomerne veľké množstvo selekčných techník. V programe `mm` sa používa iba selekcia založená na turnaji. Preto by stálo za to zvážiť, či pridaním novej selekčnej techniky by sa konvergencia ku najlepšiemu jedincovi neurýchlila.

4.1.1 Ruleta

Stará nezáživná hra ako ruleta sa dá použiť aj u selekcie molekúl. Tiež je známa ako selekcia úmerná fitnes hodnote a je jednoducho implementovateľná. Populácia sa rozdelí po častiach na kruhu, kde každá časť kruhu zodpovedá práve veľkosti fitnes daného jedinca a to v pomere ku súčtu fitnes hodnôt všetkých zúčastnených jedincov.

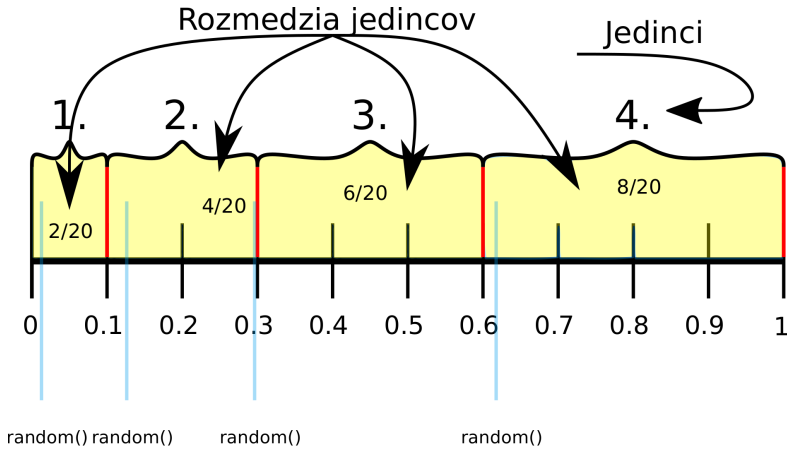
Ako to vychádza z názvu, po rozdelení populácie sa začne točiť kruhom, čo v našom prípade znamená náhodné generovanie hodnoty intervalu v akom sa nachádza rozdelená populácia. V bežnom prípade býva interval [0,360], pretože sa jedná o kruh. Existujú aj iné možnosti použitých intervalov. Napríklad po úsečke dĺžky [0,1], kde tento interval predstavuje percentuálny výsek od 0% až po 100%. Ukážka použitia tejto alternatívy je zobrazená na obrázku 4.1. Po vygenerovaní hodnoty nájdeme jedinca, pod ktorého táto hodnota spadá a pridáme ho ku jedincom, ktorí boli v tejto generácií vybraní. Toto generovanie sa opakuje toľkokrát, koľko je potrebných jedincov na kríženie[1].

Výhody:

- Je jednoducho implementovateľný.
- Jedinci s vyššou fitnes majú väčšiu šancu.

Nevýhody:

- Pri malej populácií sa môže vyskytnúť veľa duplicít.
- Nerovnomerný výber jedincov, ktorý nemusí mať žiaden vzťah ku fitnes hodnotám v danej generácii.



Obr. 4.1: Zobrazenie selekcie ruletou. Žlto vyfarbené časti obsahujú zlomok, ktorý predstavuje fitnes hodnotu jedinca ku sume fitnes všetkých jedincov. Modrou čiarou sú znázornené náhodne vygenerované selekcie jedincov. Na obrázku si možno všimnúť nerovnomerného výberu jedincov, čo je jeden z nedostatkov tejto techniky.

4.1.2 Stochastic Universal Sampling

V podstate preložiteľné ako univerzálne náhodná selekcia, bola predstavená Jamesom Bakerom v roku 1987 ako vylepšená verzia rulety. Uchováva si črty rulety, ako rozdelenie jedincov do častí po kruhu vzhľadom ku ich fitnes hodnote, ale na druhej strane sa točenie ruletou vykoná iba raz. Následne sa posúva o konštantnú dĺžku krok za krokom. Každý krok zodpovedá veľkosti získanej zo vzorca 4.2, kde dĺžkou telesa je myslené teleso, na ktoré zobrazujeme časti jedincov.

Počet krokov je ekvivalentný hodnote počtu jedincov získaných zo selekcie mínus jeden, ktorého sme dostali už náhodným generovaním. Názornú ukážku je možné vidieť na obrázku 4.3.

Výhody:

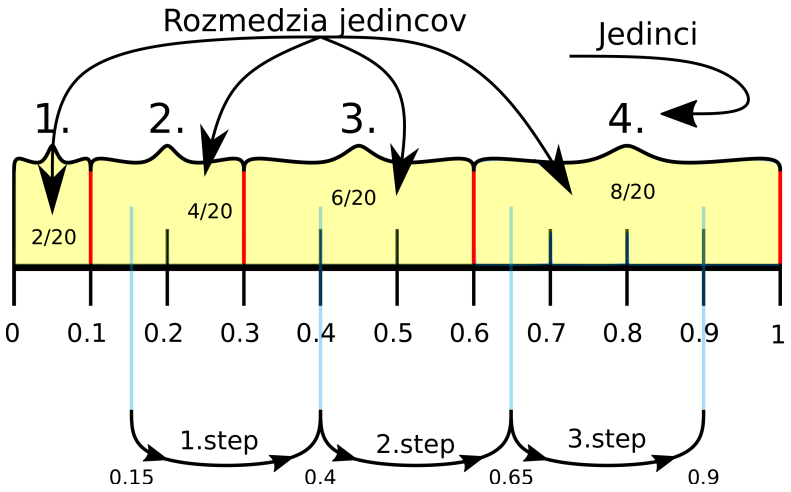
- Zložitejší ako normálna ruleta, ale stále jednoducho implementovateľný.
- Jedinci s vyššou fitnes majú väčšiu šancu.
- Pri výbere sa zachováva pomer jedincov vzhľadom ku ich fitnes, teda rovnomerný výber.
- Funguje správne aj pri malej populácii.

Nevýhody:

- Žiadne o ktorých by sme vedeli.

$$velkostKroku = \frac{1 * dlzkaTelesa}{pocetSelectovanychJedincov} \tag{4.1}$$

Obr. 4.2: Vzorec na výpočet kroku.



Obr. 4.3: Ukážka realizácie selekcie stochaic universal. Žlto vyfarbené časti obsahujú zlomok, ktorý predstavuje fitnes hodnotu jedinca ku sume fitnes všetkých jedincov. Prvá modrá čiara bola náhodné generovaná, hodnota 0.15. Ostatné boli dosiahnuté pomocou krokov od počiatočnej modrej čiary. Modré čiary poukazujú do rozmedzia jedincov, ktorí sú vybraní zo selekcie.

4.2 Mutácia

Mutácia by sa mohla rozšíriť o elementárnu chémiu, v zmysle nahradenia terajšej mutácie fragmentov za mutáciu atómov v molekule. Čiže miesto odpojenia jedného fragmentu a pripojenia iného z cudzej molekuly na toto miesto, by sa mohol prechádzať priestor uzlov. V prípade, že daný uzol nereprezentuje vodík a máme ho zmutovať, tak ho nahradíme iným prvkom z rovnakej kategórie podľa periodickej tabuľky. Navyše pri výbere náhradného atómu by sme mohli brať ohľad aj na ďalšie vlastnosti ako sú elektronegativita, oxidačné čísla prvkov, vodivosť a iné vlastnosti, ktoré sa nachádzajú v periodickej tabuľke prvkov (ďalej ako PTP).

Kategórie nahradného atómu podľa :

- periódy
- rodiny/triedy
- kovosti - kovy, nekovy, polokovy
- iných vlastností

Použitím prvkov s podobnými vlastnosťami, by sa teoreticky mohla zvýšiť šanca reálnej existencie danej zlúčeniny, čo je vlastne veľmi žiadaná schopnosť tohto algoritmu. Problémom nie je generovať rôzne molekuly, ale ohodnotenie vygenerovaných molekúl[6].

Výhody:

- Mutácia atómov bude robiť iba malé zmeny, na rozdiel od mutácie fragmentami.
- Umožní algoritmu sa vyhnúť lokálnemu minimu.

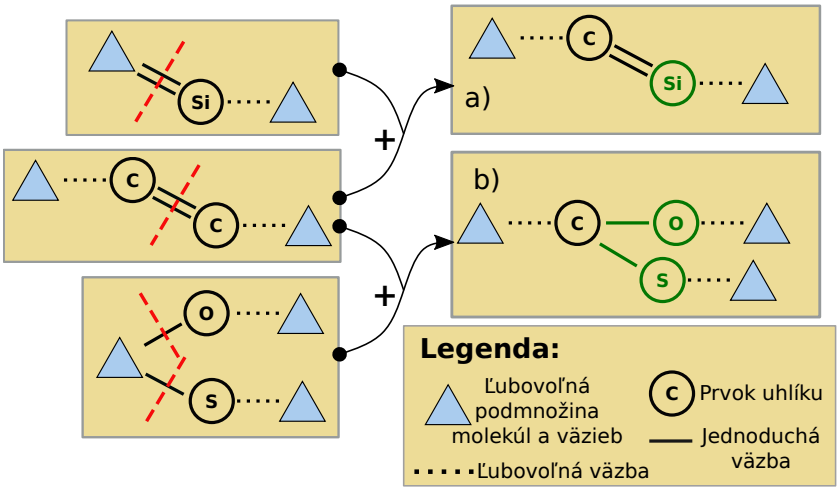
Nevýhody:

- Vygenerované molekuly môžu byť úplne nepoužiteľné.

4.3 Kríženie

Ďalšia možnosť by bola vylepšenie operátora kríženia. Podmienka zachovania násobnosti väzieb, počtu väzieb a atómov v bode rezu by sa zredukovala iba na zachovanie počtu väzieb. Teda jedna dvojnásobná väzba môže byť nahradená dvomi jednoduchými väzbami a zároveň sa nemusia zachovať prvky z pôvodných väzieb.

Využitie výhod potlačením podmienok kríženia sa dá realizovať iba čiastočne v prípade jednobodového kríženia a to využitím nezachovania prvkov pôvodných väzieb. Na druhej strane pri viacbodovom krížení využije sa táto technika bez problémov. Znázornenie týchto dvoch situácií sa nachádza na obrázku 4.4.



Obr. 4.4: Ukážka použitia nového kríženia. Krížením prvých dvoch jedincov vznikne molekula **a)** a krížením druhého a tretieho jedinca vznikne molekula **b)**. V prvom prípade sa nezachovali pôvodné atómy v mieste kríženia a v druhom prípade sa nezachovala násobnosť väzieb ani typ atómov v mieste kríženia. Dodržuje sa iba pravidlo počtu väzieb.

Výhody:

- Zvyšuje sa rozmanitosť krížených molekúl.
- Zvyšuje sa šanca na kríženie.

Nevýhody:

- Vygenerované molekuly môžu byť úplne nepoužiteľné.
- Je komplikovaný na implementáciu.

4.4 Technika ostrovov

Technika ostrovov [8] sa pokúša napodobniť rozdelenie populácie medzi civilizácie a jej samostatný vývoj podľa reálneho sveta. V našom prípade civilizácie budú reprezentovať ostrovy. Ostrovy budú fungovať navzájom nezávisle, pokiaľ nepríde okamih, kedy sa stretnú a premiešajú. Buď príde genocída pri strete civilizácií ako španielskí conquistadori alebo dôjde ku zmysluplnej komunikácii[3]. Napríklad medzi európskou a ázijskou civilizáciou, ktorá neprebiehala podobne ako vyššie spomínaná genocída pri strete civilizácií na rozdielných vývojových úrovniach. Obidve možnosti sú priaznivé GA, keďže

[3]Týmto myslím napríklad obdobie 13. storočia, kedy Európan Marco Polo žil v Ázii. Získaval celkom cenné informácie, ktoré sa dostali do Európy. Je to jeden z mála príkladov spolupráce civilizácií, pretože v histórií sa zväčša jedná o dominanciu jednej civilizácie nad druhou a nie o ich spolupráci.

v konečnom dôsledku to môže iba vylepšiť fitnes aspoň jednej generácie. Ak by sme zobrali príklad spomínanej genocídy jednej populácie druhou, tak za cenu vyvraždenia a zotročenia veľkej časti populácie Indiánov sa populácia premiešala a jej vývojová úroveň[4] sa dostala postupne k úrovni Európy. V prípade, že by táto kolízia nenastala, tak by Indiáni museli na miesto naháňania a upaľovania čarodejníc naháňať svojích domorodých susedov s následným obetovaním ich životov Bohu.

Výhody:

- Jednoducho pochopiteľná myšlienka techniky.
- Zvyšuje šancu na kríženie.

Nevýhody:

- Môže vyvolať neočakávané komplikácie pri implementácií, keďže ostrovy musia fungovať nezávisle a tým pádom používané prostriedky by mali byť oddelené od ostatných ostrovov.

[4]Vývojová úroveň je u GA myslená ako fitnes hodnota.

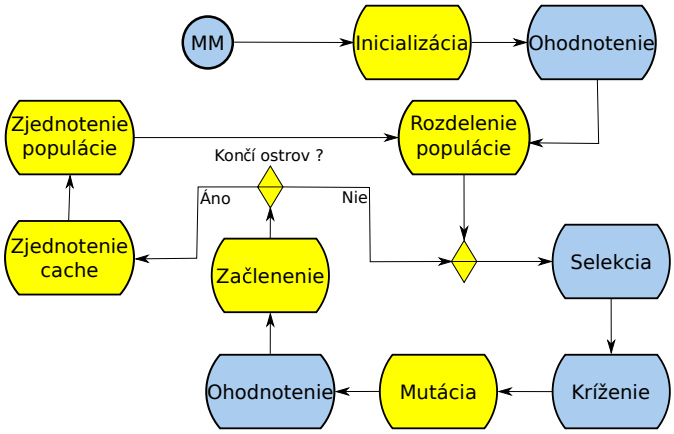
Kapitola 5

Implementácia

Pri implementácií som sa rozhodol zachovať všetky prvky predchádzajúcich bakalárskych prác. Inými slovami som zachoval používanie programovacieho jazyka shell pri riadiacom skripte `mm` a jazyka C pri genetických operátoroch.

Tak ako aj v pôvodnej verzii, aj táto pokračuje modelom, že každý z genetických operátorov je samostatná komponenta, ktorá je riadená skriptom `mm`. Obsahuje všetky pôvodné nástroje a to selekciu, kríženie, mutáciu a začlenenie. Z týchto spomenutých nástrojov sú upravené mutácie, začlenenie a skript `mm`. Navyše do skriptu `mm` boli pridané ďalšie nástroje, implementované v jazyku shell. Sú to nástroje rozdelenia populácie, zjednotenia cache a zjednotenia populácie. Nástroje boli vytvorené z dôvodu uľahčenia a vylepšenia pôvodnej práce o rozšírenú mutáciu, a o pridanie techniky ostrovov.

Popis týchto spomenutých nových aj upravených nástrojov sa bude rozoberať v nasledujúcich sekciách. Tieto časti sú tiež vizualizované na obrázku 5.1, pri čom popisujú procesy v novom skripte `mm`.

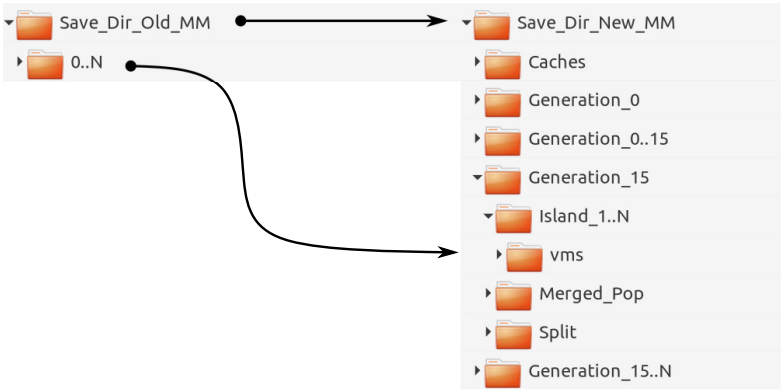


Obr. 5.1: Zjednodušený diagram nového behu `mm` skriptu, ktorý vychádza z pôvodného programu `mm` na obrázku 3.1. Modré časti reprezentujú staré časti a žlté úplne nové alebo upravené.

5.1 Inicializácia

V inicializácií sa zmenilo vytvorenie priečinkovej[5] štruktúry výsledkov. Namiesto obsahovania všetkých výsledných súborov v jednom priečinku som sa rozhodol rozdeliť výsledné súbory vytvorené `mm` do viacerých priečinkov, viď. obrázok 5.2. Dôvodom tohto rozhodnutia bola nečitateľnosť pri väčšom množstve generácií a zároveň kvôli pridaniu techniky ostrovov, kde je potrebné nechať fungovať vývoj ostrovov nezávisle na ostatných ostrovoch a tým pádom je nutné, aby každý ostrov si ukladal dáta samostatne. Inak by sa čitateľnosť adresára výsledkov starého `mm` ešte viac znížila.

Rozdelenie závisí od čísla generácie a následne od čísla ostrova[6], v ktorom sa proces práve nachádza. Táto štruktúra generácií ostrovov sa vytvára pred ohodnotením prvotnej generácie poskytnutej užívateľom. Navyše v niektorých generáciách existujú priečinky Merged_Pop[7], Split[8] a `vms`[9], avšak tieto priečinky sa vytvárajú až vo chvíli, keď skript `mm` potrebuje uložiť dáta do daného priečinka.



Obr. 5.2: Porovnanie pôvodnej a novej štruktúry v skripte `mm`. Adresárová štruktura nového `mm` sa pomerne zväčšila po premiestnení súborov do správnych generácií a ostrovov, ale v konečnom dôsledku je lepšie čitateľnejšia, kvôli novej štruktúre.

5.2 GA technika ostrovov

Technika ostrovov sa dá realizovať buď sekvenčne alebo paralelne. Hoci u mňa býva väčšinou zvykom vyhýbať sa paralelným algoritmom, v tomto prípade som sa rozhodol ho implementovať práve paralelne. Vydal som sa zmysluplnejšou cestou.

Prispôsobil som pôvodný beh (obrázok 3.1) tak, že som genetické operátory zahrnul do funkcie v jazyku shell a následne vytvoril cyklus, ktorý forkuje funkciu zahrňujúcu

[5] Synonymum ku slovu adresár.
[6] Vysvetlené v sekcii GA technika ostrovov.
[7] Merged_Pop využíva nástroj na zjednotenie populácie.
[8] Split využiva nástroj na rozdelenie populácie.
[9] Priečinok `vms` využiva nástroj na ohodnotenie populácie, `vms`.

genetické operátory s potrebnými argumentami obsahujucími hodnoty generácie, ostrova a taktiež to ako dlho má daný ostrov pracovať samostatne. Po naforkovaní všetkých ostrovov čaká hlavné vlákno na ostrovy, aby dobehli.

Až po realizácií forkov som narážal na rôzne problémy. Najdôležitejšie bolo cachovanie vypočítaných molekúl, rozdelenie populácie a spajanie populácie po ukončení forkov. Kvôli týmto problémom som bol nútený vytvoriť nástroje, ktoré riešia dané problémy. Implementáciu techniky ostrovov som rozdelil do viacerých častí:

- Rozdelenie populácie
- Zjednotenie cache
- Zjednotenie populácie

Ku všetkým trom zo spomenutých problémov sa budem venovať ešte v nasledujúcich podsekciách.

Z dôvodu pridania techniky ostrovov som pridal potrebné prepínače do skriptu `mm`, ktoré ovládajú fungovanie techniky ostrovov. Sú to nasledujúce prepínače:

`-a --num-islands island_size` Hodnota `island_size` definuje, koľko ostrovov sa bude používať počas behu programu `mm`.

`-d --islands-merge island_life_length` Hodnotou `island_life_length` špecifikuje dĺžku životností ostrova, teda po koľkých generáciach sa má spojiť s ostatnými ostrovmi.

`-h --population-fraction population_fraction` Nastavením hodnoty `population_fraction` sa definuje, koľko jedincov by mal obsahovať ostrov z pôvodnej generácie.

5.2.1 Rozdelenie populácie

Poskytnutím užívateľovi možnosť špecifikovať veľkosť populácie práve na každom ostrove sa vytvoril menší, ale riešiteľný problém. Preto som sa rozhodol rozdeliť pôvodnú generáciu náhodne medzi ostrovy. Použitý princíp rozdeľovania populácie medzi ostrovmi sa zakladá na jedinečnosti jedincov na ostrove. Algoritmus rozdelenia funguje nasledovne:

1. Pre každý ostrov vytvoriť súbor obsahujúci očislujúce riadky od 1 až N, kde N je počet jedincov v pôvodnej populácií.
2. Začneme prvým ostrovom a nechám si vygenerovať náhodné číslo veľkosti rovnej počtu riadkov v súbore z prvého bodu pre daný ostrov.
3. Vygenerované číslo reprezentuje riadok v súbore z prvého bodu pre daný ostrov. Pomocou tohto čísla nájdem jedinca v pôvodnej populácií a zkopírujem ho do prvého ostrova. Taktiež skopírujem jeho fitnes hodnotu do súboru obsahujúceho fitnes hodnoty danej populácie.

4. Po vykonaní daný riadok zmažem a začínam znova druhým bodom, pokiaľ nebudem mať požadovaný počet jedincov.

5. Po vykonaní predchádzajúcich operácií pre prvý ostrov, pokračujem v daných krokoch pre ďalšie ostrovy, pokiaľ nerozdelím pôvodnú populáciu pre všetky ostatné ostrovy.

Vytvorený postup na rozdelenie pôvodnej populácie medzi ostrovy nie je najšikovnejší, ale spĺňa účel. Určite zabezpečuje, že sa na žiadnom ostrove nenájde duplicita. Taktiež generuje náhodne a nemalo by sa pri generovaní stať, že sa na nejakú dobu zacyklí, kvôli zabráneniu duplicity.

5.2.2 Zjednotenie cache

Z dôvodu možnosti viacerých ostrovov je pridelená každému ostrovu samostatná cache, ktorá sa po určitom počte generácií spojí do hlavnej cache a z nej sa opäť vytvoria cache pre ostatné ostrovy. Tento cyklus sa pravidelne zachováva a snaží sa aspoň čiastočne šetriť program `vms` pred náročným prepočítavaním molekúl, ktoré sa už vyhodnotili.

5.2.3 Zjednotenie populácie

Zjednotenie populácie bol tiež menší oriešok. Postup funkcie zjednotenia je nasledovný:

1. Pomocou programu OpenBabel sa vytvoria hashe pre každého jedinca z populácie každého ostrova a uložia do súboru.

2. Ku každému hashu pridá jeho ohodnotenie, číslo jedinca na ostrove a číslo ostrova.

3. Zjednotia sa všetky tieto vytvorené hashe a zoradia sa podľa fitnes zostupne.

4. Tento zoradený súbor sa oreže na počet riadkov, koľko obsahovala pôvodná generácia.

5. Následne sa z tohto súboru vytvorí nová generácia. Definície molekúl sa získajú z pôvodných ostrovov a k tomu sa pridá súbor obsahujúci ohodnotenie novej generácie.

5.3 Mutácia

Poslednú dôležitú časť v práci, ktorú som menil bol genetický operátor mutácie. Realizoval som zmeny podľa návrhu. Rozšíril som mutáciu o nový prepínač a napísal novú funkciu, ktorá riadi mutáciu atómov[10].

Funkcia mutácie atómov potrebovala ovládať elementárne znalosti chémie. Tým pádom som implementoval nové zdrojové súbory obsahujúce základy chémie využívajúc

[10]Na rozdiel od mutácie fragmentov molekúl, kde je pre každú molekulu šanca mutovať iba raz, pri mutovaní atómov má šancu každý nevodíkový atóm.

vlastnosti PTP z obrázku 5.4. Dané prvky z tabuľky sú napísane do statického poľa a navyše sú tam pripravené funkcie, ktoré pracujú nad statickými poľami. Prvky v zdrojových súboroch obsahujú nasledujúce vlastnosti:

- Skratku prvku.
- Číslo prvku v PTP.
- Periódu[11] do ktorej patrí daný prvok.
- Rodinu, resp. triedu[12] do ktorej patrí daný prvok.
- Podkategóriu[13] daného atómu.
- Existenciu[14] oxidačných čísel pre každý atóm.

S pripravenými funkciami obsahujúcimi vedomosti o PTP, ktorých priebeh je znázornený na obrázku 5.3, boli pridané aj prepínače špecifikujúce, na základe ktorej kategórie vlastností chce užívateľ vyberať množinu substitučných atómov[15]. Následne sa podľa ďalšieho prepínača rozhodne, či si z množiny substitučných atómov má vyberať náhodne alebo pomocou najbližšieho oxidačného čísla atómu.

Nové prepínače sú špecifikované v skripte `mm` následovne:

`-k --atom-mutation atom_state` Hodnota `atom_state` prepína medzi mutáciou fragmentov a atómov.

`-o --oxid-state-mutation oxid_state` Nastavením tohto prepínača na 1 sa uprednostní výber nového atómu pomocou oxidačného čísla namiesto náhodného výberu.

`-p --atom-mutation-type mutation_type` Špecifikuje, aký typ kategórie sa má používať pri náhradných atómoch.

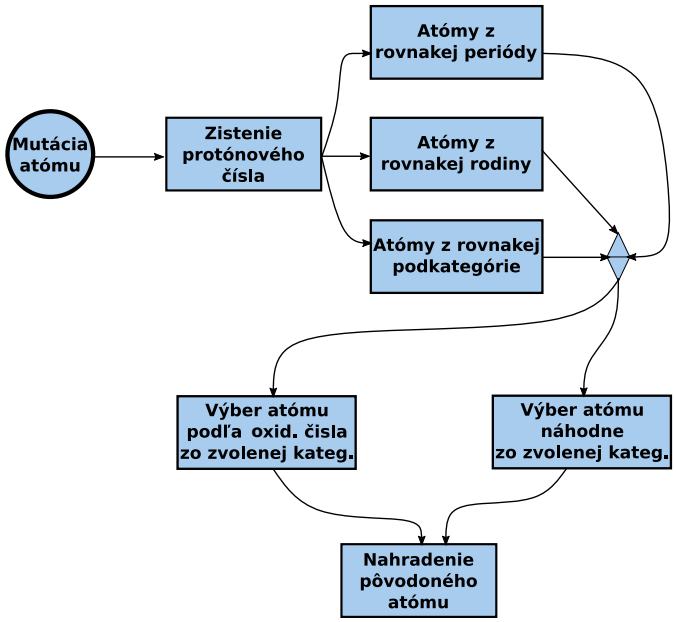
[11] Vyznačené na obrázku 5.4 ako `period` a reprezentuje riadky v PTP.

[12] Vyznačené na obrázku 5.4 ako `family or group`a reprezentuje stĺpce v PTP.

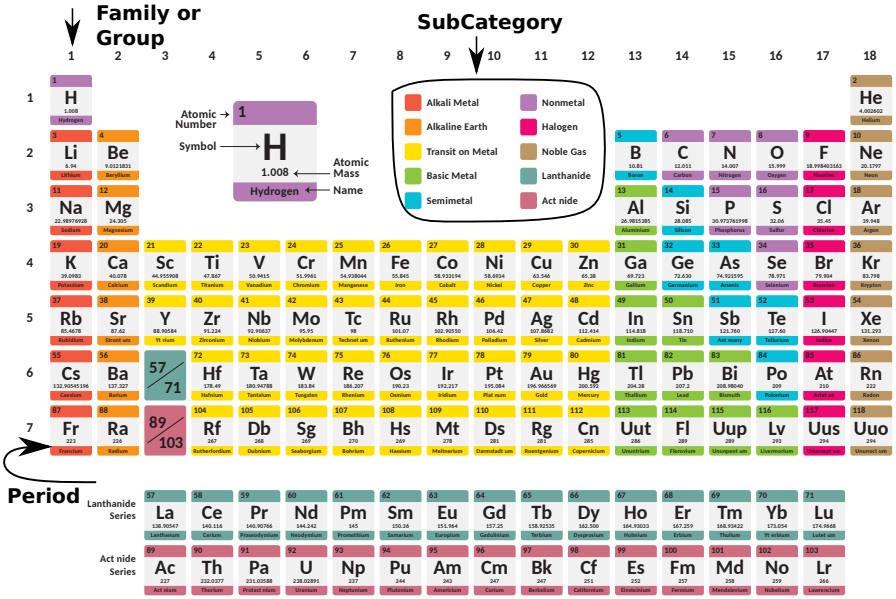
[13] Vyznačené na obrázku 5.4 ako `subcategory`.

[14] Hodnoty oxidačných čísel som získal z [9], hoci sú veľmi nepravdepodobné, chcel som aspon väčšiu vzorku oxidačných čísel než je základná.

[15] Na výber má možnosť si vybrať podľa periódy, rodiny alebo podkategórie.



Obr. 5.3: Zjednodušený diagram novej atómovej mutácie za pomoci určitých vlastností z PTP.



Obr. 5.4: Ukážka periodickej tabuľky [10] prvkov pre lepšiu predstavu.

5.4 Ostatné

Počas implementácií sa diali aj ďalšie zmeny, ale podľa môjho názoru si nevyžadovali samostatnú sekciu, preto sa ich pokúsim zhrnúť v krátkosti.

Všetky pôvodné jednoznakové prepínače skriptu `mm` boli rozšírené o dlhé prepínače. Mal som pocit, že krátke prepínače už nemusia byť jednoznačné pri toľkých prepínačoch, tak som ich rozšíril a to isté platí aj u nových.

Niektoré časti kódu dostali dokumentáciu, ktorú som sám napísal, aby som pochopil presnejšie, ako fungujú určité časti. Pre ďalších odhodlancov, ktorí sa rozhodnú pokračovať v tejto práci, by to mohlo byť nápomocné.

Komponenta začlenenia bola rozšírená o prepínač, ktorý povoľuje rast populácie o hodnotu špecifikovanú užívateľom. Funguje celkom jednoducho. Od začiatku až ku koncu pracuje komponenta zlúčenia rovnako. Až pred koncom vyhodnotí, či bol nastavený prepínač na rast populácie. Ak bol, tak sa pokúsi pridať ďalších jedincov do novej generácie, ktorí ešte neboli použití. V prípade, že hodnota prepínača na rast populácie je väčšia než počet jedincov doposiaľ nepoužitých, tak ich pridá iba toľko, koľko ma jedincov ku dispozícii.

Toto rozšírenie bolo pridané z dôvodu vyhnutiu sa klesania veľkosti populácie. Napríklad v prípade, ak by sa populácia rozdelila rovnomerne medzi ostrovy a po ich spojení existovalo veľké množstvo duplicít, ktoré by boli následne vyradené.

Kvôli pridaniu prepínača do začlenenia bol taktiež pridaný prepínač do skriptu `mm`.

`-l --growing-pop growth_size` Prepínač na spustenie rastu populácie v nástroji začlenenia s hodnotou `growth_size`.

Kapitola 6

Testovanie

Rozšírený program bol úšpešne nainštalovaný a spustený spoločne so softwarom `vms` a dalšími závislosťami ako sú OpenBabel a ORCA[16] na nasledujúcich systémoch:

- Gentoo 4.9.95 (x86_64)
- OpenSUSE 42.3 (x86_64)
- MacOS 10.6.8 (x84_64)

Rozšírený program bol úšpešne nainštalovaný a spustený bez softwaru `vms` na nasledujúcich systémoch:

- Ubuntu 16.04 (x86_64)
- Debian 9.4.0 (amd64)

Na testovanie sa používala rovnaká databáza, s približne rovnako dlhou dobou a s podobnými počiatočnými nastaveniami ako u kolegu Matějku. Všetky testy prebiehali na výpočtovej technike laboratória SAGElab[11]. Vstupnou databázou pre genetický algoritmus bol zoznam molekúl použitý mojím predchodcom a pochádzal z článku *Computational Design and Selection of Optimal Organic Photovoltaic Materials*[12]. Nastavenie na vyhodnocovanie molekúl, ktorý používa `vms` je na obrázku 6.1. Je to zadanie z práce *Genetické algoritmy pre generovanie molekúl*[6]. Táto požiadavka špecifikuje, že sa molekuly majú vyhodnocovať na základe intenzity absorpcie elektromagnetického žiarenia v zadanom intervale vlnových dĺžok, teda v našom prípade dĺžky 0–800 nm.

[16] Bola použitá verzia 3, ktorá je kompatibilná s `vms`.

```
# General mandatory directives
job     excited
method  BP86

# General optional directives
basis   def2-SV(P)
memory  2048

# Job specific mandatory directives
nroots  10

# Job specific optional directives
iroot   3
maxdim  150

# Job specific optional filters
absorb 0 800 0.001 1.00 1.00
```

Obr. 6.1: Kópia požiadavky[6], podľa ktorého vyhodnocoval `vms`.

Celkovo bolo vykonaných päť testov väčšinou so štandardnými nastaveniami. Zmenilo sa počet generácií na 90, šanca mutácie 3% a kopírovala sa cache už vypočítaných molekúl[17] na včasné dopočítanie testov. Boli urobené tieto testy:

Originál Test zahrňujúci pôvodne nastavenia na otestovanie funkčnosti, výsledky na grafe 6.2.

Ostrovy 1 Test sa spúšťal s 5 ostrovami, ukončovaním ostrovov po 15 generáciách a s rozdelením populácie po 30%. Zvyšné nastavenia neboli zmenené.

Ostrovy 2 Test sa spúšťal s podobnými nastastaveniami ako u ostrovov 1, ale mal zvýšené rozdelenie populácie na 75%.

Mutácia atómov 1 Vychádza z testu originál s výnimkou zapnutej mutácie atómov, používajúc atómy z rovnakej podkategórie a s nasledovným vybratím podľa oxidačného čísla[18].

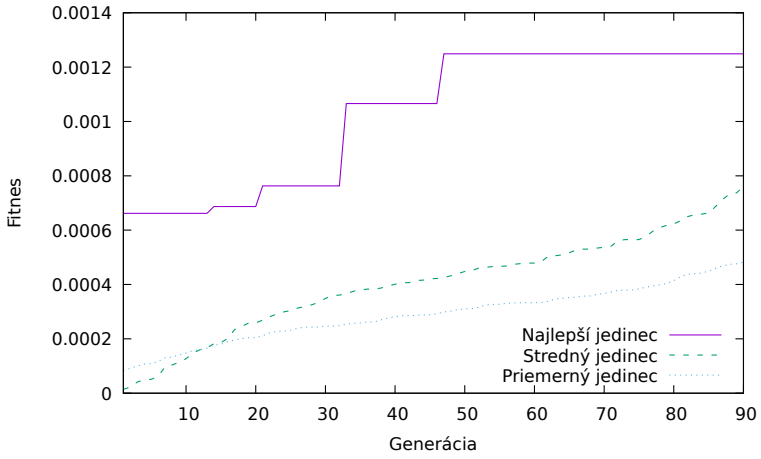
Mutácia atómov 2 Podobne ako test mutácia atómov 1, len nepoužíva oxidačné čísla, ale vyberie náhodne náhradníka z rovnakej podkatégorie.

Počiatočná populácia obsahovala 132 molekúl a pre každý test bolo vypočitaných 90 generácií[19]. Doba testovania presiahla približne 265 hodín. Ohodnotenie jednej mole-

[17] Obsahuje pôvodne okolo 50 tisíc vyhodnotených molekúl.

[18] Experimentálny test, či to má reálne zmysel, tým myslím výber atómu podľa najbližšieho oxidačného čísla.

[19] Táto hodnota bola zvolená z dôvodu časovej náročnosti a za druhé sa ukázalo z predchádzajúcej práce [6], že už sa generácie razantne nemenia.



Obr. 6.2: Graf zobrazujúci výsledky testu, ktorý bol spustený s pôvodnými nastaveniami a technikami. Je vidno, že podľa najlepšieho jedinca, graf dosiahol postačujúcich výsledkov. Výsledky tohto testu sa budu používať ku porovnaniu rozdielov medzi originálnym GA a GA rozšíreným o ďalšie techniky.

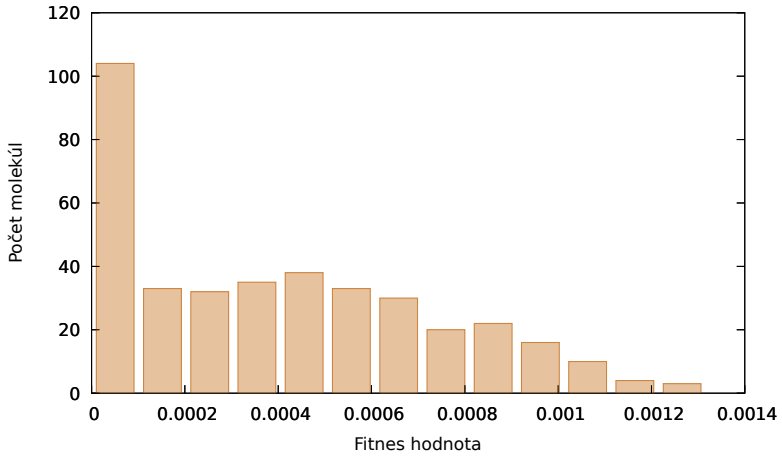
kuly trvalo priemerne 77 sekúnd. Výpočet jednej generácie trval priemerne 44 minút[20]. V porovnaní s prácou kolegu Matějku, by bolo možné vidieť, že výsledné hodnoty sú niekoľkonásobne vyššie. Našťastie sa nejedná o žiaden problém, len o výchylku vo výpočte časových dát. Technika ostrovov používa ostrovy, ktoré sú forknuté a fungujú ako samostatné procesy. Po dokončení svojej úlohy sa dokončené procesy zjednotia, ale celkovo sa čaká na dokončenie posledného ostrova aby sa mohlo pokračovať v novej ostrovnej etape.

Podiel na dobe výpočtu ohodnotenia jednej generácie je možné vidieť z grafu 6.9. Jedná sa o dáta iba z testu ostrovy 2. Taktiež je tam vidno spomenuté výchylky každú 15 generáciu, kde sa čaká na dokončenie všetkých ostrovov na rovnakej generačnej úrovni, aby sa mohli premiešať. Réžia `mm` pri výpočte jednej generácie v prípade upravenej techniky mutácie sa nezmenila v porovnaní s pôvodným algoritmom. Priemerne dosahuje 0.3% réžie celkového programu. V prípade techniky ostrovov sa réžia znížila na 0.027%[21] réžie celkového programu.

Na grafoch 6.4, 6.6 a 6.8 je zobrazený vývoj fitness vybraných jedincov v priebehu generácií. U prvých dvoch obrázkov sú zobrazené samotné priebehy populácií na ostrovoch a u tretieho obrázku je možné vidieť spojenie grafu originálnych nastavení a grafu ostrova s veľkou populáciou. Sledované hodnoty boli hodnotenia najlepšieho jedinca, stredného jedinca (medián) a priemerného jedinca.

[20] Najkratšie trvalo pod 1 sekundu. Bolo to v prípade už vypočítanej populácie uloženej v cache. Najdlhšie približne 3 hodiny

[21] Táto hodnota nezahŕňa réžie výpočtu generácií, kde bola cache využitá na viac ako 80% a v prípade, že sa v danej generácií vykonávalo spájanie populácií, čo predchádzalo čakaniu na ostatné ostrovy.

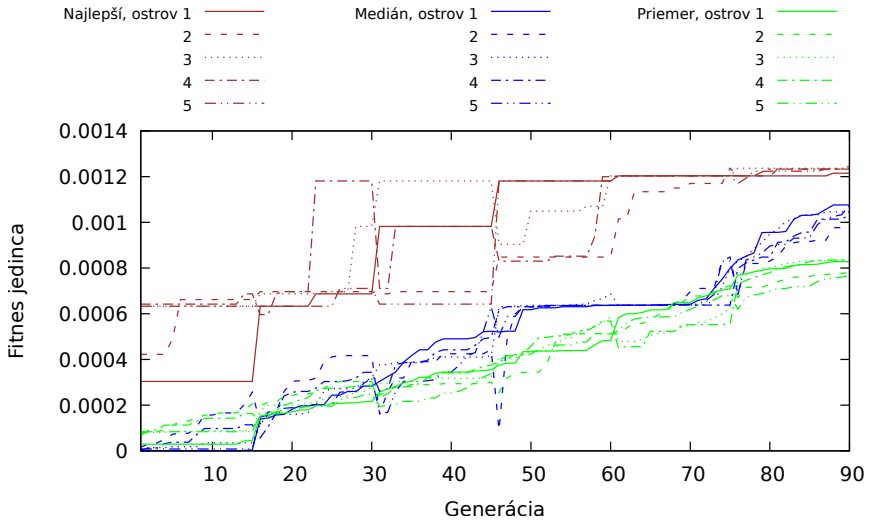


Obr. 6.3: Graf zobrazujúci výsledky testu originál, ktorý bol spustený s pôvodnou implementáciou a nastaveniami. Je vidno, že podľa najlepšieho jedinca, graf dosiahol postačujúcich výsledkov. Výsledky tohoto testu sa budu používať ku porovnaniu rozdielov medzi originálnym GA a jeho rozšírením ďalšími technikami.

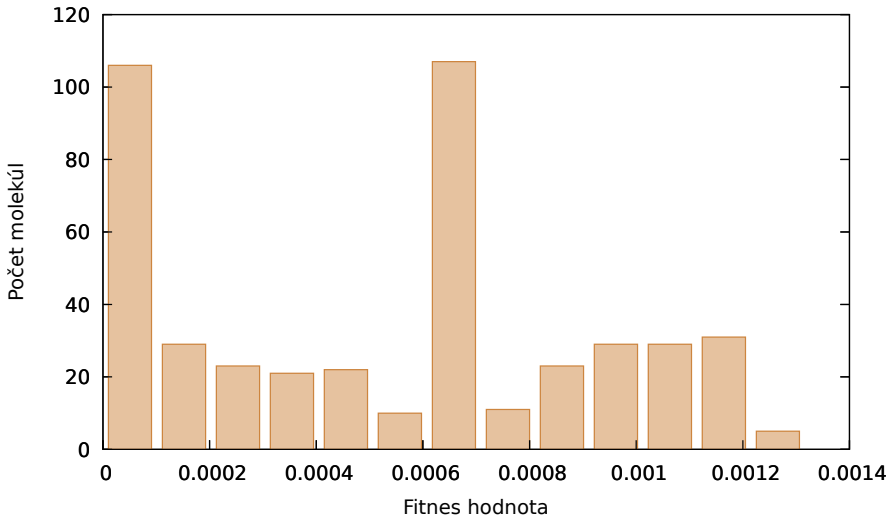
6.1 Testovanie techniky ostrovov

Testovanie techniky ostrovov zahŕňa dva testy. Rozdiel medzi testovaniami je iba v prerozdelení pôvodnej populácie medzi ostrovy. V prvom teste je táto hodnota nastavená na 30% a u druhého na 75%. Podľa výsledkov prvého testu na grafe 6.4 sú viditeľné výrazné poklesy a nárasty medzi ostrovami hlavne počas miešania populácie ostrovov. Taktiež sú viditeľné problémy na histograme 6.5. Rozdelenie celkovej vygenerovanej populácie počas generácií ukazuje generovanie najviac jedincov hodnotám blížiacim sa nule alebo hodnotám okolo 0.7 x 10^{-3}. Hodnoty blížiacim sa nule sú pochopiteľné, pretože generácie začínajú slabými jedincami a cieľom je ich vyšľachtiť ku maximu, čo sa nedá povedať o veľkom počte jedincov s priemernou fitnes.

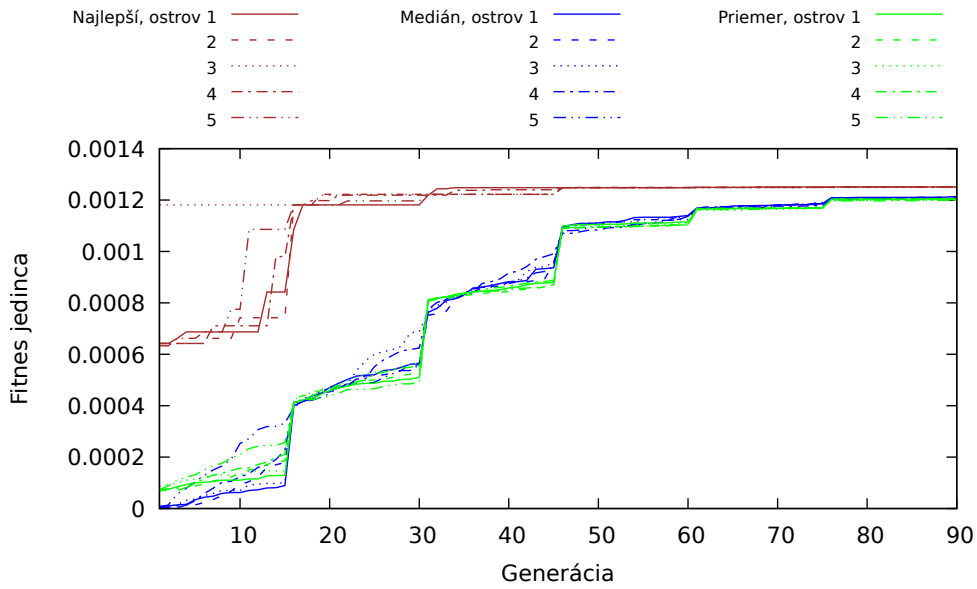
Druhý test vykazuje oveľa dôveryhodnejšie hodnoty na grafe 6.6. Je vidno rovnomerne rastúcu populáciu vo všetkých smeroch u každého ostrova na rozdiel od prvého testu 6.4. Histogram 6.7 tiež vykazuje nárast jedincov s elitnou fitnes hodnotou blížiacej sa nájdenemu maximu. Pri porovnaní druhého a pôvodného testu 6.2 s pôvodnou implementáciou, sa dá vidieť na grafe 6.8 jednoznačné zlepšenie pri použití techniky ostrovov na generovanie molekúl.



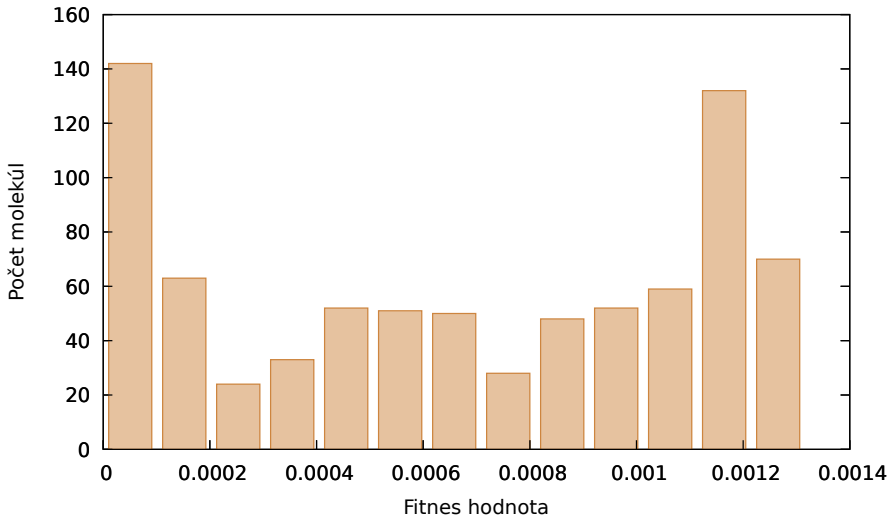
Obr. 6.4: Graf zobrazujúci vyhodnotenie testu ostrova s malou populáciou. Pridal som ho z dôvodu poukázania nezmyselnosti používania techniky ostrovov s malou populáciou. Je vidno ako sa hodnoty menia o viac ako tretinu za jednú generáciu.



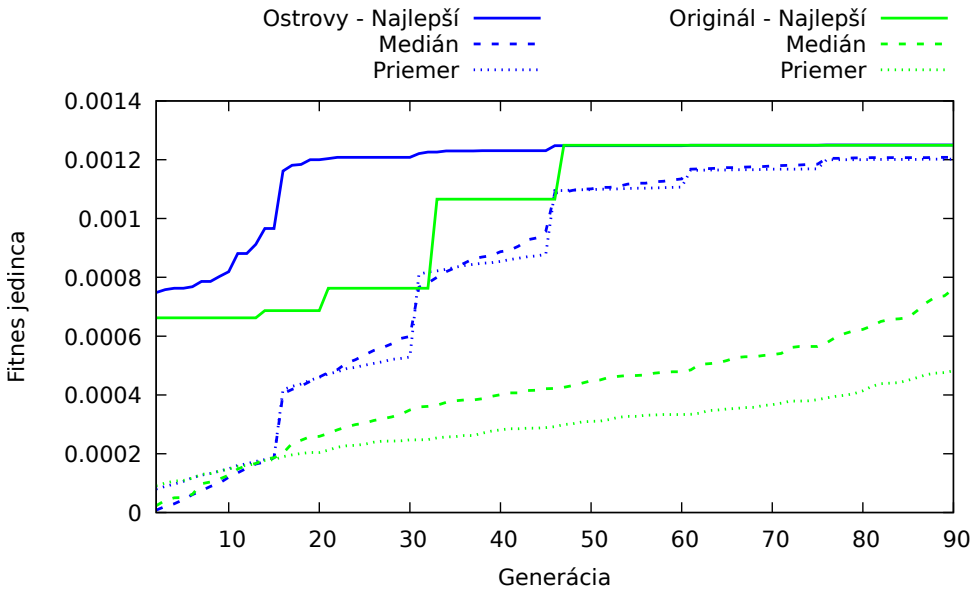
Obr. 6.5: Zobrazenie všetkých populácií vygenerovaných pomocou techniky ostrovov s menšou populáciou.



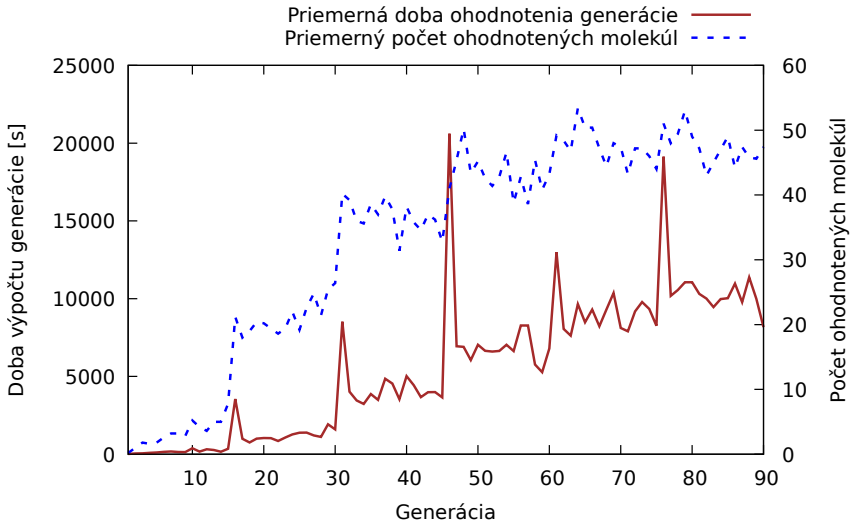
Obr. 6.6: Graf zobrazujúci vyhodnotenie testu ostrova s väčšou populáciou. Ukazuje presný opak od grafu 6.4. Populácia ostrovov pomaly stúpa a ako skupina konverguje ku maximálnej hodnote.



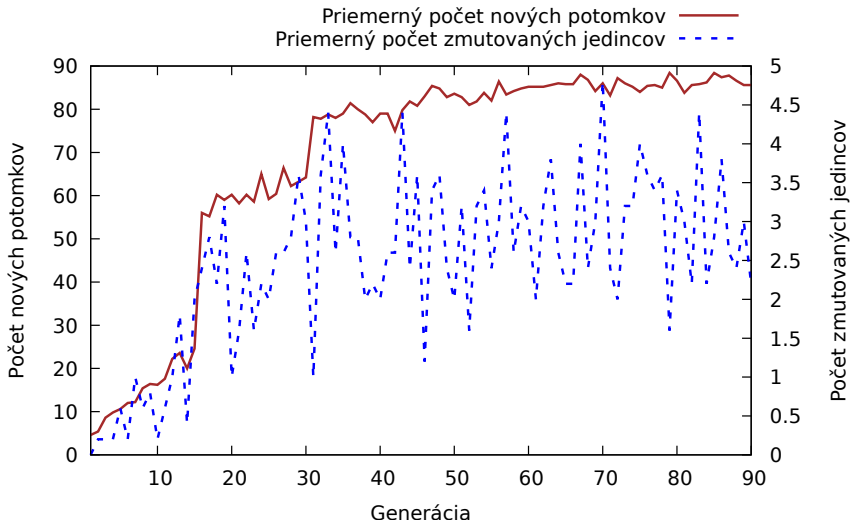
Obr. 6.7: Zobrazenie všetkých populácií vygenerovaných pomocou techniky ostrovov s väčšou populáciou.



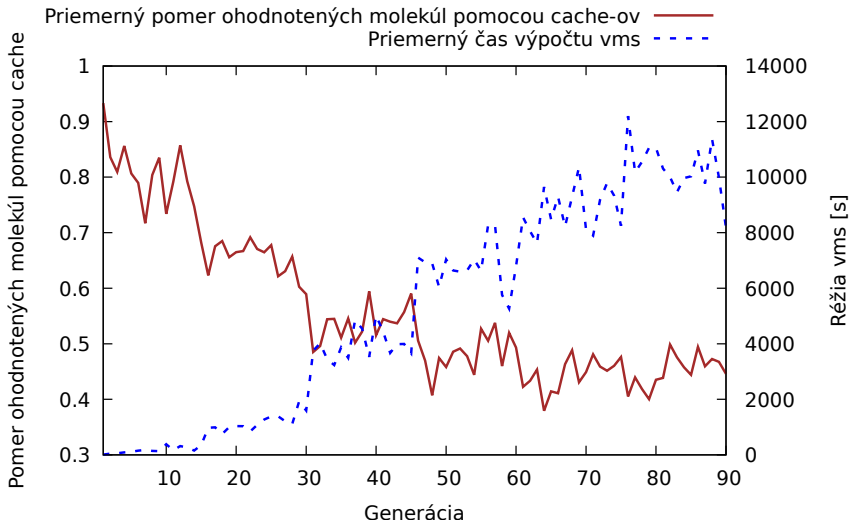
Obr. 6.8: Spojenie grafov s pôvodnými GA a s novo implementovanými GA. Na grafe je vidno, že novo vylepšený algoritmus sa zlepšuje nielen v rýchlosti konvergencií najlepšieho jedinca, ale tiež v zlepšení celkovej populácie. Najväčšie skoky vidno práve v násobkoch 15, čo je vlastne doba, kedy sa jedinci ostrovov miešajú.



Obr. 6.9: Graf zobrazujúci stúpanie časovej réžie programu vzhľadom ku počtu ohodnotených molekúl. Taktiež je možné pozorovať na grafe extrémne zmeny na dobe výpočtu pri násobkoch 15, čo sa deje úmyselne z dôvodu spájania ostrovov.



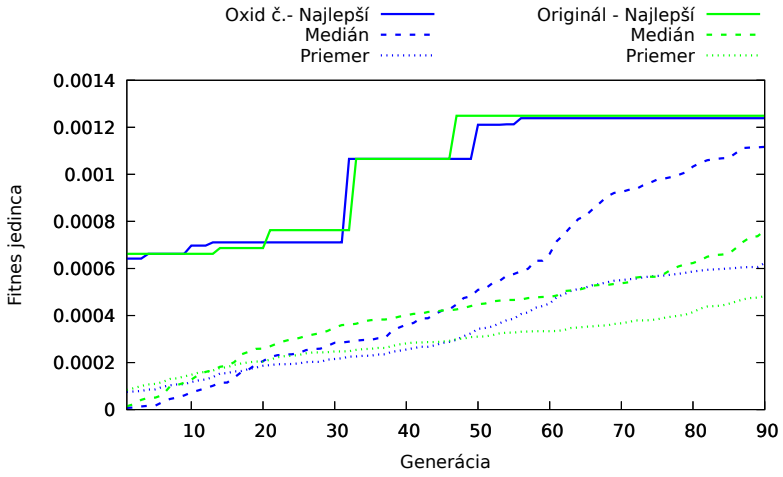
Obr. 6.10: Na grafe sa dá pozorovať slabá funkčnosť GA v prvých 15 generáciách, čo je zapríčinené jedincami jednoduchých molekúl, až časom je vidno nárast v krížení a tiež mutácií, ktorá si zachováva hodnotu. Z dôvodu pomalého kríženia na začiatku by sa mohla zvážiť zmena kríženia alebo zmenenie vstupnej databázy.



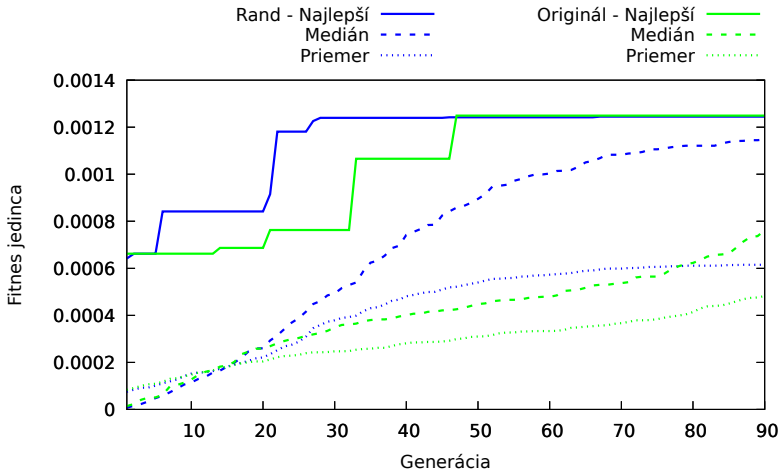
Obr. 6.11: Graf zobrazujúci logickú závislosť použitia cache pri výpočte molekúl. Je vidno klesanie využitia cache v pokročilých generáciach so stúpajúcou dobou využívania `vms`.

6.2 Testovanie atómovej mutácie

V testovaní atómovej mutácie sa uskutočnili dva testy s rozdielným typom mutácie atómov. Jeden sa zameriava na mutáciu atómov výberom pomocou oxidačných čísel 6.12. Druhý zas na mutáciu náhodným výberom atómov 6.13.

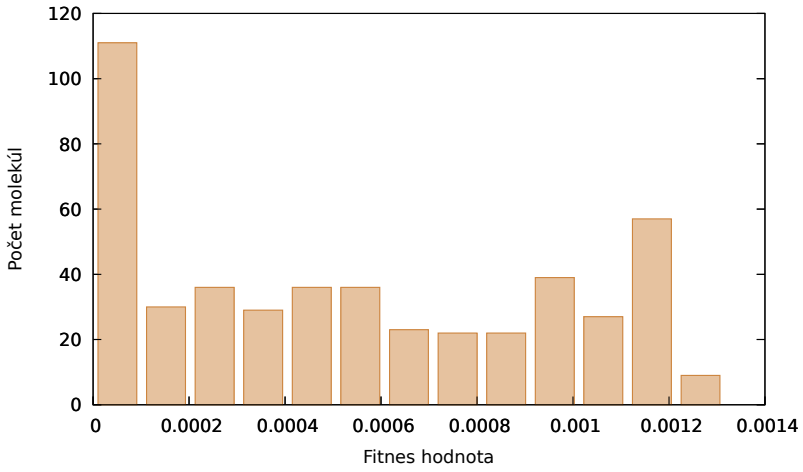


Obr. 6.12: Porovnanie priebehu pôvodnej mutácie s mutáciou atómami a oxidačnými číslami. Viditeľné zlepšenie na strane stredného jedinca. Na druhej strane nepopierateľné zhoršenie v elitných jedincoch a v priemere populácie.



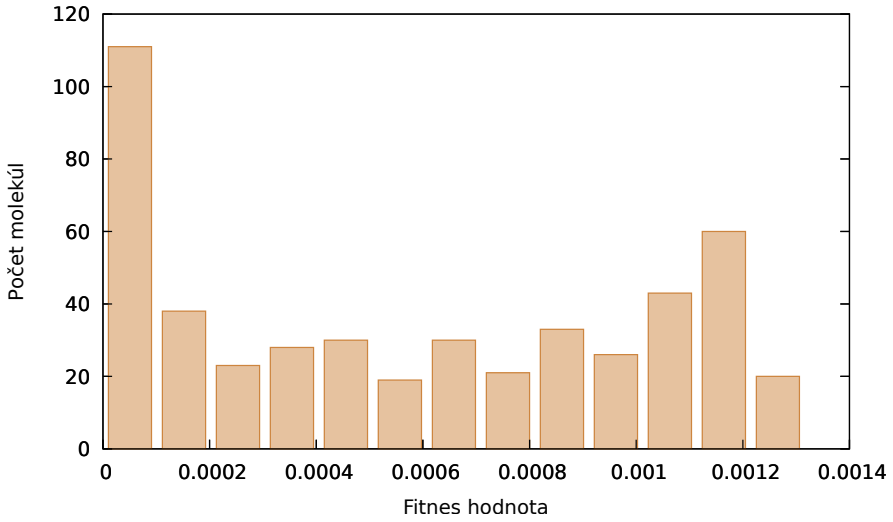
Obr. 6.13: Porovnanie priebehu pôvodnej mutácie s mutáciou atómami a náhodným výberom atómov. Viditeľné zlepšenie na strane stredného jedinca. Na druhej strane nepopierateľné zhoršenie u elitných jedincov a v priemere populácie.

Mutácia pomocou oxidačných čísel vykazuje zlepšenie oproti pôvodnému algoritmu. Je viditeľne lepší z dvoch kategórií porovnávaných na grafe 6.12. Tiež na histograme 6.14 je možné vidieť nárast generovania jedincov s vyššou fitnes hodnotou v porovnaní s histogramom 6.3 z originálneho behu. Jediná nevýhoda je, že nedosahuje najlepšieho jedinca porovnateľného s originálnym algoritmom na generovanie molekúl.

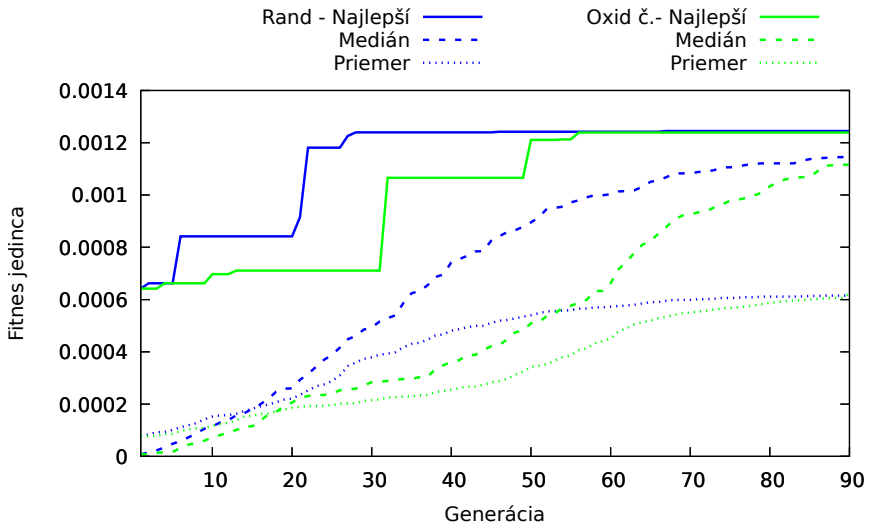


Obr. 6.14: Zobrazenie všetkých populácií vygenerovaných pomocou mutácií použitím oxidačného čísla.

Nie len mutácia pomocou oxidačných čísel vykazuje pozitívne výsledky, ale aj mutácia pomocou náhodného výberu atómu zo zvolenej kategórie. Na grafe 6.13 je vidno podobné zlepšenia ako u mutácie s oxidačným číslom. Taktiež to potvrdzuje histogram 6.15, kde je vidno jemné posunutie časti populácie napravo ku vyšším fitnes hodnotám. Nevýhoda je, že tento posun populácie je na úkor jedincov s priemernou hodnotou fitnes. Z grafu 6.16 by sa dalo povedať, že metóda mutácie s náhodným výberom atómov je vo všetkých smeroch lepšia než mutácia pomocou oxidačných čísel, ale je otázne či by to platilo aj pri behu dlhšom ako 90 generácií, keďže so zvyšujúcou generáciou sa približujú výsledky mutácie s oxidačnými číslami ku mutácii s náhodným výberom atómu.



Obr. 6.15: Zobrazenie všetkých populácií vygenerovaných pomocou mutácií použitím oxidáčného čísla.



Obr. 6.16: Zobrazenie všetkých populácií vygenerovaných pomocou mutácií použitím oxidáčného čísla.

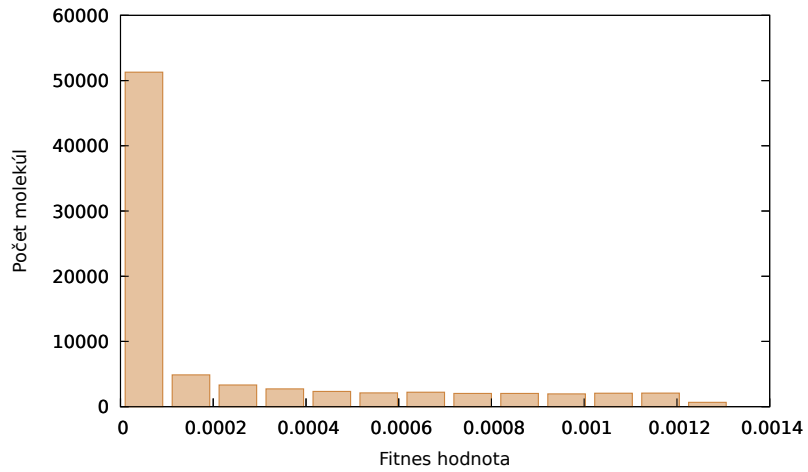
6.3 Zhrnutie výsledkov testov

Počas vývoja, ale aj samotného testovania bol použitý jeden a ten istý súbor ako spoločná fitnes cache. Taktiež obsahuje fitnes hodnoty z predchádzajúceho vývoja kolegu Matějku. Pred začatím vývoja obsahovala cache približne 50 tisíc záznamov o ohodnotených molekulách. Po ukončení testovania sa rozrástla na približne 80 tisíc záznamov. Využitie spoločnej fitnes cache neporovnateľne zrýchlilo testovanie 6.11 v počiatočných generáciach. Počas testovania spomenutých testov v laboratóriu bolo ohodnotených viac nez 100 tisíc molekúl, čo nezahŕňa počet molekúl testovaných počas vývoja. Z dát uložených do fitnes cache bol vytvorený histogram 6.17. Z vygenerovaných molekúl spomenutými testami s fitness väčšou ako 1.17 x 10^{-3} boli vyhľadané v databáze PubChem [13]. Z vyhladaných molekúl(celkom 230), boli nájdené tri z toho dve boli aj obchodne dostupné.

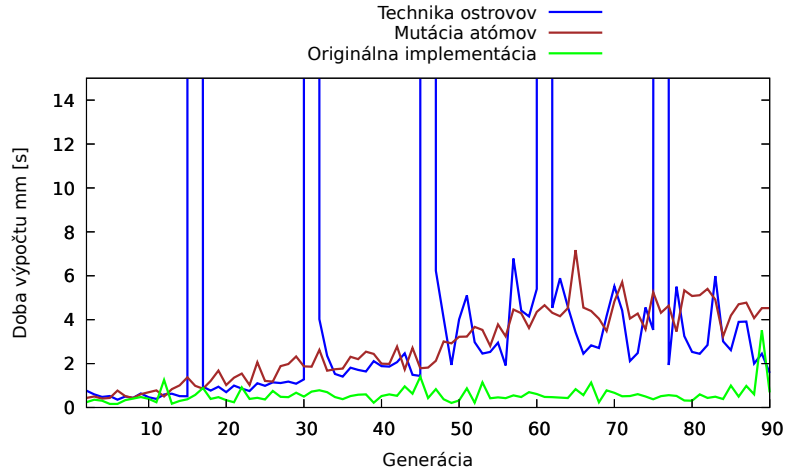
Rozšírenie pomocou techniky ostrovov sa preukázal na grafe 6.8 ako vhodná voľba do genetického algoritmu za predpoklad, že vstupná populácia je dostatočne veľká dokáže formovať populáciu ako celok a tým pádom dosiahnuť väčšie množstvo jedincov z ktorých môžu niektoré byť reálne použité. Tento poznatok je viditeľný na grafe 6.7. Približne štvrtina jedincov má fitnes väčsiu ako 1.1 x 10^{-3}. Je na zamyslenie akých hodnôt by dosahoval test s ostrovami, ak by sme ho nechali bežať až na 200 generácií.

Na rozdiel od techniky ostrovov, rozšírenie implementácie u mutácie na mutovanie atómov nepriniesla dych berúce výsledky. Podľa grafu 6.12 a 6.13 by sa dalo povedať, že mutácia atómami je porovnateľná s mutáciou fragmentov molekúl. Viditeľnú výhodu má mutácia atómov na generovanie viacej jednicov s väčšou fitnes hodnotou na úkor jedincou s priemernou hodnotou. Je to viditeľné na histograme 6.14 a 6.15.

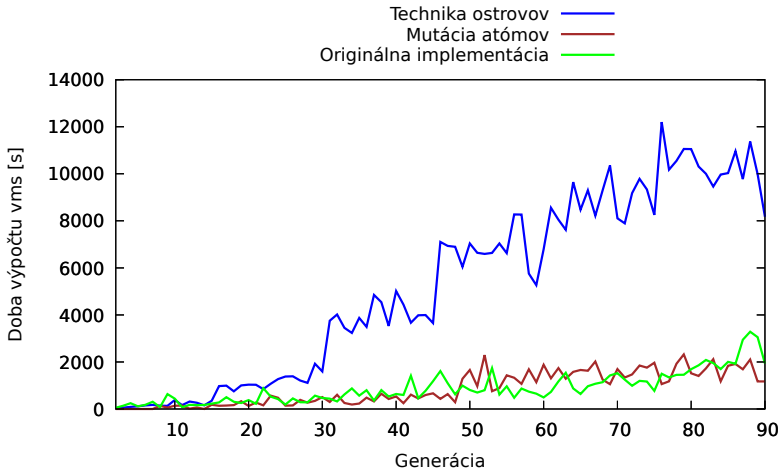
Časová réžia rozšíreného `mm` o nové techniky sa nijak špeciálne nezmenila. Podľa grafu 6.18 sa réžia `mm` približne zoštvornásobila oproti pôvodným výsledkom, ale v porovnaní ku výpočtom `vms` je to zanedbateľná hodnota, viď. 6.19. Na grafe 6.19 je viditeľná stúpajúca réžia oproti ostatným technikám. Podľa môjho názoru dôvod tohto navýšenia je možné vidieť v grafe 6.8, kde je pozorovateľné rovnomerné stúpanie fitnes populácie s každou generáciou. Tým pádom sa pravdepodobne zvyšuje komplexnosť molekúl a to zas spôsobuje zvýšenú réžiu `vms`, čo sa už predpokladá v úvode práce.



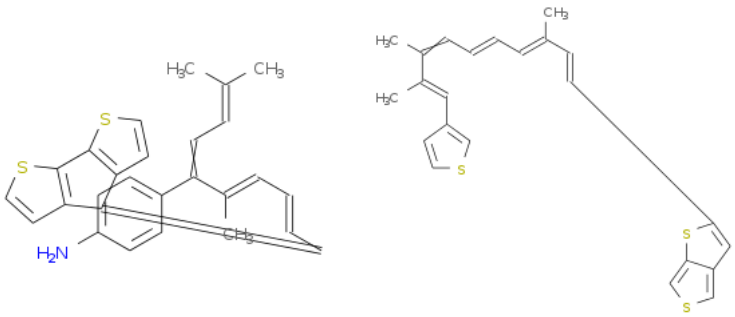
Obr. 6.17: Histogram počtu ohodnotených jedincov pre danú fitnes. Drvivá väčšina molekúl mala fitnes hodnotu blízku nule.



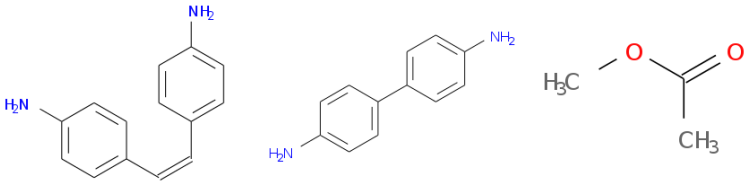
Obr. 6.18: Porovanie doby výpočtu programu `mm` u rozdielnych techník. Extrémne výkyvy techniky ostrovov sa konajú každú 15 generáciu a sú spôsobené čakaním na dobehnutie všetkých ostrovov.



Obr. 6.19: Porovanie doby výpočtu programu `vms` u rozdielnych techník.



Obr. 6.20: Dve najlepšie molekuly s fitnes aspoň 1.25 x 10^{-3} .



Obr. 6.21: Molekuly s fitnes aspoň 1.14 x 10^{-3} s existujúcim záznamom v databáze PubChem [13].

Záver

Cieľom práce bolo rozšírenie programu pre generovanie molekúl podľa požiadaviek užívateľa na základe predchádzajúcich prác kolegov Sršňa a Matějku. Po zodpovednom vykonaní rešerše boli účelovo zvolené rozšírenia, ktoré môžu rozšíriť množinu vytvorených molekúl a zároveň prispieť ku vytvoreniu celkovej elitnejšej populácie.

Výsledkom je rozšírený program na generovanie molekúl o techniku ostrovov a mutáciu atómov. Tieto rozšírenia sú napísane buď v programovacom jazyku C alebo skriptovacom jazyku shell. Rozšírenia sú riadne popísané v manuálových stránkach a nemajú vplyv na pôvodnú kompatibilitu s rôznymi unixovskými systémami. Určité časti kódu dostali aj malú dokumentáciu, v ktorej by sa mohlo pokračovať v budúcnosti. Tiež boli pridané dlhé prepínače pre jednoduchšie a intuitívnejšie ovládanie behu programu.

Do budúcna by som odporúčal sa zamerať na implementovanie grafického rozhrania, aby si bežný užívateľ mohol sám vyskúšať rôzne techniky a aj samotný program. Prepísať kód z jazyka C do objektového programovacieho jazyka, rozšíriť súčasne GA ako kríženie o základy chémie navrhované v mojej práci, vytvorenie globálnej knižnice molekúl, kde by sa ukladali molekuly aj s ich hashovacími hodnotami, z dôvodu jednoduchšieho nájdenia vyhodnotených molekúl a údajov získaných z `vms`.

Tiež by sa mohlo zamerať na ďalšie testovanie s rôznymi vstupnými hodnotami a analyzovať dopad týchto vstupných parameterov na výslednú populáciu.

Literatúra

[1] Mitchell, M.: *An introduction to genetic algorithms*. MIT press, 1998, ISBN 0-262-13316-4.

[2] Lohn, J. D.; Hornby, G. S.; Linden, D. S.: An Evolved Antenna for Deployment on NASA's Space Technology 5 Mission. Chapter 1. Technická zpráva, NASA Technical Reports Server, 2004, [Online]. Dostupné z: `https://ti.arc.nasa.gov/m/pub-archive/812h/0812%20(Lohn).pdf`.

[3] Štepán Sršeň: *Distrubuovaná infrastruktura pro virtuální screening molekul*. Bakalárska Práca ,České Vysoké učení technické v Praze, Fakulta Informačných Technológií, 2017.

[4] Open Babel: XYZ (format). [Online], 2007, [cit. 2.5.2018]. Dostupné z: `http://openbabel.org/wiki/XYZ_(format)`.

[5] Max-Planck-Institute for Chemical Energy Conversion: ORCA manual. [Online], [cit. 2.5.2018]. Dostupné z: `https://orcaforum.cec.mpg.de/`.

[6] Matějka, J.: *Genetické Algoritmy pro generování molekul*. Bakalárska Práca ,České Vysoké učení technické v Praze, Fakulta Informačných Technológií, 2017.

[7] O'Boyle, N. M.; Banck, M.; James, C. A.; aj.: Open Babel: An open chemical toolbox. *Journal of cheminformatics*, ročník 3, č. 1, 2011: str. 33, doi:10.1186/1758-2946-3-33. Dostupné z: `https://openbabel.org`.

[8] Coley, D. A.: *An introduction to genetic algorithms for scientists and engineers*. World Scientific Publishing Company, 1999, ISBN 981-02-3602-6.

[9] Oxidation period table. compoundchem.com, [Online], [cit. 29.4.2018]. Dostupné z: `http://www.compoundchem.com/wp-content/uploads/2015/11/The-Periodic-Table-Of-Oxidation-States-2016.png`.

[10] Periodic Table. Vecteezy.com, [Online], [cit. 29.4.2018]. Dostupné z: https://www.vecteezy.com/vector-art/86493-rainbow-periodic-table.

[11] CESNET, z. s. p. o.: SAGElab. [Online], 2018, [cit. 1.5.2018]. Dostupné z: https://sagelab.cesnet.cz/.

[12] O'Boyle, N. M.; Campbell, C. M.; Hutchison, G. R.: Computational design and selection of optimal organic photovoltaic materials. *The Journal of Physical Chemistry C*, ročník 115, č. 32, 2011: s. 16200–16210, doi:10.1021/jp202765c.

[13] National Center for Biotechnology Information: PubChem Compound Database. [Online], 2018, [cit. 2.5.2018]. Dostupné z: https://pubchem.ncbi.nlm.nih.gov/.

Dodatok **A**

Zoznam použitých skratiek

EA Evolučná algoritmus

GA Genetická algoritmus

PTP Periodická tabuľka prvkov

Dodatok **B**

Obsah priloženého DVD

- `readme.txt` ..... popis obsahu DVD a postup inštalácie
- `data`
  - `input` ..... vstupné dáta použité v testoch
  - `output` ..... výstupné dáta z vykonaných testov
- `src`
  - `impl` ..... zdrojové kódy implementácie
  - `thesis` ..... zdrojová forma práce vo formáte LaTeX
- `text`
  - `thesis.pdf` ..... text práce vo formáte PDF