Univerzita Komenského v Bratislave
Fakulta matematiky, fyziky a informatiky

Vizualizácia evolučných histórií genómov s viacerými chromozómami

Bakalárska práca

2017
Radoslav Chládek

Univerzita Komenského v Bratislave
Fakulta matematiky, fyziky a informatiky

Vizualizácia evolučných histórií genómov s viacerými chromozómami

Bakalárska práca

Študijný program:	Informatika
Študijný odbor:	2508 Informatika
Školiace pracovisko:	Katedra informatiky
Školiteľ:	doc. Mgr. Bronislava Brejová, PhD.

Bratislava, 2017
Radoslav Chládek

Univerzita Komenského v Bratislave
Fakulta matematiky, fyziky a informatiky

ZADANIE ZÁVEREČNEJ PRÁCE

Meno a priezvisko študenta: Radoslav Chládek
Študijný program: informatika (Jednoodborové štúdium, bakalársky I. st., denná forma)
Študijný odbor: informatika
Typ záverečnej práce: bakalárska
Jazyk záverečnej práce: slovenský
Sekundárny jazyk: anglický

Názov: Vizualizácia evolučných histórií genómov s viacerými chromozómami
Visualization of evolutionary histories of multichromosomal genomes

Cieľ: Cieľom práce je rozšíriť existujúci nástroj EHDraw na zobrazovanie evolučných histórií tak, aby zobrazoval aj histórie, v ktorých je zobrazovaná DNA rozdelená na niekoľko chromozómov. V tomto rozšírení vzniká otázka, ako preusporiadať jednotlivé chromozómy v každom bode histórie tak, aby v zobrazení nevznikali zbytočné kríženia čiar reprezentujúcich jednotlivé gény. Cieľom bude tento problém sformulovať ako optimalizačný problém, nájsť v literatúre alebo navrhnúť algoritmy na jeho presné alebo približné riešenie a vybraný algoritmus aj implementovať.

Vedúci: doc. Mgr. Bronislava Brejová, PhD.
Katedra: FMFI.KI - Katedra informatiky
Vedúci katedry: prof. RNDr. Martin Škoviera, PhD.

Dátum zadania: 12.10.2016

Dátum schválenia: 17.10.2016 doc. RNDr. Daniel Olejár, PhD.
garant študijného programu

.....
študent

.....
vedúci práce

Poďakovanie: Ďakujem mojej školiteľke, doc. Mgr. Bronislave Brejovej, PhD. za všetky cenné rady, odbornú pomoc a čas, ktorý mi venovala.

Abstrakt

V práci sa zaoberáme problémom vizualizácie zmien poradia génov v genómoch, ku ktorým dochádza počas evolúcie. Práca rozširuje existujúci nástroj na vizualizáciu evolučných histórií *EHDraw* tak, aby nástroj zobrazoval aj evolučné histórie genómov s viacerými chromozómami. Hlavným výsledkom práce je implementácia optimalizačneho algoritmu na preusporiadanie, otočenie a rezanie chromozómov tak, aby výsledný obrázok bol čo najprehľadnejší, obsahoval minimálny počet krížení a zároveň celá informácia, uložená v evolučnej histórii v ňom zostala zachovaná. Náplňou práce bolo taktiež testovanie algoritmu na evolučných históriách s rôznymi parametrami a hľadanie závislosti medzi danými parametrami a počtom krížení pred a po zbehnutí nášho optimalizačného algoritmu.

Kľúčové slová: vizualizácia, evolučná história, minimalizácia počtu krížení

Abstract

In the thesis, we consider visualisation of changes in the gene order, that occur during evolution. We extend the existing tool *EHDraw* for visualisation of evolutionary histories, so that it can visualize histories of genomes with multiple chromosomes. The main result of the thesis is an implementation of an optimization algorithm for reordering and changing orientation of chromosomes, so that resulting image is comprehensible, contains the minimum number of crossings and there is no loss of information stored in the evolutionary history. The thesis also involves testing of the algorithm on evolutionary histories with various parameters, as well as identifying dependence between given parameters and the number of crossings before and after running our optimization algorithm.

Keywords: visualisation, evolutionary history, crossings minimization

Obsah

Úvod **1**

1 Úvodné pojmy **3**

2 Nástroj EHDraw a jeho rozšírenie **9**
2.1 Nástroj EHDraw 9
2.1.1 Vstup 9
2.1.2 Výstup 10
2.2 Rozšírenie nástroja EHDraw 11
2.2.1 Vstup 11
2.2.2 Výstup 11
2.2.3 Ovládanie cez príkazový riadok 12
2.2.4 Triedy a dátové štruktúry 13

3 Vizualizácia grafov **15**
3.1 Pojmy z teórie grafov 15
3.2 Hierarchické kreslenie grafov 15
3.3 Problém minimálneho počtu krížení 16
3.3.1 Jednostranný minimalizačný problém 18
3.3.2 Viac-vrstvový minimalizačný problém 19

4 Minimalizácia krížení evolučnej histórie **20**
4.1 Graf evolučnej histórie 21
4.2 Problém poradia chromozómov 21
4.3 Problém otočenia chromozómov 23
4.4 Problém rezania cirkulárnych chromozómov 24
4.5 Celková optimalizácia 24

5 Experimentálne vyhodnotenie **27**
5.1 Pomocné triedy a ďalší použitý softvér 27
5.2 Generované dáta 28
5.2.1 Závislosť skóre s_m a s_{rm} 31

5.3 Reálne dáta 31

Záver **34**

Elektronická príloha **38**

Zoznam obrázkov

1.1 Štruktúra DNA 4
1.2 Kvasinkový fylogenetický strom 6
1.3 "Vláčikový"spôsob zobrazenia kvasinkovej evolučnej histórie 7
1.4 Výstup nástroja EHDraw 8

2.1 Ukážka vstupu nástroja EHDraw 10
2.2 Ukážka výstupu a grafickej aplikácie nástroja EHDraw 11
2.3 Ukážka vstupu rozšíreného nástroja EHDraw 12
2.4 Ukážka výstupu rozšíreného nástroja EHDraw 13

3.1 Postup Sugiyamovej metódy 17

4.1 Priebeh barycentrovej heuristiky 22
4.2 Ukážka optimalizácie otáčaním chromozómov 23
4.3 Ukážka optimalizácie rezania cirkulárnych chromozómov 24
4.4 Ukážka optimalizácie väčšej histórie 26

5.1 Graf závislosti skúmaných skóre a počtu listov evolučnej histórie 29
5.2 Graf závislosti skúmaných skóre a počtu génov v genómoch evolučnej histórie 29
5.3 Graf závislosti skúmaných skóre a počtu DCJ operácií medzi vrstvami evolučnej histórie 30
5.4 Graf závislosti skúmaných skóre a počtu chromozómov vo vrstvách evolučnej histórie 30
5.5 Grafy zobrazujúce pomery minimalizovaných skóre pre rôzne parametre generovania evolučných histórií 32

Zoznam tabuliek

5.1 Tabuľka skúmaných hodnôt skóre na reálnych vstupoch 33

Úvod

Teória evolúcie bola pôvodne formulovaná Charlesom Darwinom v 19. storočí a spresňovaná a rozširovaná generáciami vedcov. Táto teória hovorí o tom, že všetky biologické druhy vznikli z iných biologických druhov vďaka zmenám v ich genetickej informácii. K týmto zmenám dochádza najmä vďaka vonkajším vplyvom pôsobiacim na mnoho generácií po sebe. Tieto vplyvy postupne vedú k zmene genetickej informácie jedincov do takej miery, že vznikne nový biologický druh.

Genetická informácia je súbor génov obsiahnutý v bunkách živých organizmov, prenášaný z generácie na generáciu. Genetická informácia je zapísaná v molekule DNA, ktorá je uložená v štruktúrach, ktoré sa nazývajú chromozómy. Evolučnou históriou viacerých biologických druhov nazveme históriu ich genetickej informácie, t.j. popis stavu genetickej informácie v každom z biologických druhov a ich spoločných predkov a popis zmien, ktoré sa postupne udiali. Postupom rokov biológovia, matematici a informatici vyvinuli rôzne metódy, ktoré rekonštruujú evolučné histórie na základe dostupných dát o dnešných druhoch. Takto vytvorené evolučné histórie je potrebné vhodným spôsobom zobrazovať, čo je aj námetom tejto práce.

Cieľom tejto bakalárskej práce je rozšírenie existujúceho nástroja EHDraw [12], ktorý vizualizuje evolučné histórie druhov, ktorých genetická informácia je uložená v jednom chromozóme. Zdrojový kód sme upravili tak, aby nástroj zobrazoval aj histórie druhov, ktorých genetická informácia pozostáva z množiny viacerých chromozómov. Pri takomto rozšírení vzniká otázka, ako preusporiadať chromozómy v každom bode evolučnej histórie, aby výsledný obrázok obsahoval čo najmenej krížení. Hlavným cieľom práce je túto otázku previesť na optimalizačný problém, nájsť riešenie tohto problému a výsledný algoritmus aj implementovať.

Text práce je rozdelený na päť kapitol. V prvej kapitole zadefinujeme pojmy z biológie a bioinformatiky, ktoré súvisia so zobrazovaním evolučných histórií. V druhej kapitole opíšeme stav nástroja EHDraw pred aj po jeho rozšírení. V tretej kapitole si povieme niečo viac o vizualizácii grafov, predstavíme problém minimálneho počtu krížení hrán grafu a v literatúre nájdeme a popíšeme vhodné algoritmy, ktoré tento problém riešia. Vo štvrtej kapitole zadefinujeme graf evolučnej histórie a problém minimálneho kríženia prevedieme na optimalizačný problém na tomto grafe. Opíšeme

algoritmus, ktorý rieši daný optimalizačný problém a tento algoritmus aj následne implementujeme. V poslednej, piatej kapitole algoritmus otestujeme na rôznych typoch evolučných histórií.

Kapitola 1

Úvodné pojmy

V tejto kapitole oboznámime čitateľa so základnými pojmami dôležitými pre túto prácu a predstavíme súčasný stav problematiky.

DNA (Deoxyribonukleová kyselina) je nositeľka genetickej informácie, ktorá riadi rast, delenie a regeneráciu všetkých bunkových organizmov. Skladá sa z dvoch komplementárnych vlákien (reťazcov nukleotidov), ktoré sú usporiadané v tvare dvojzávitnice. Nukleotidy sú zložené z cukru deoxyribózy, fosfátovej skupiny a jednej zo štyroch nukleových báz, t.j. adenín (A), cytozín (C), guanín (G) a tymín (T). Na základe komplementárnosti nukleotidov A-T a C-G vieme vytvoriť k jednému vláknu DNA jeho komplementárne vlákno. Sekvenciu nukleotidov, ktorá je ekvivalentná s pôvodnou sekvenciou, ale nachádza sa na komplementárnom vlákne DNA, nazveme sekvenciou s **opačnou orientáciou**. Získame ju tak, že každý z nukleotidov nahradíme jeho komplementárnym nukleotidom a otočíme poradie sekvencie. Napríklad pre sekvenciu ATGCAC je jej sekvencia s opačnou orientáciou GTGCAT.)

Jednotlivé molekuly DNA sú usporiadané do štruktúr, ktoré sa nazývajú **chromozómy** (viď obr. 1.1). Chromozómy môžeme rozdeliť do dvoch skupín - lineárne a cirkulárne. Cirkulárny chromozóm je štruktúra, ktorá obsahuje DNA v tvare kruhu (bez voľných koncov). Po „prerezaní“ takéhoto chromozómu na niektorom mieste sa chromozóm zmení z cirkulárneho na lineárny. Skutočné lineárne chromozómy majú na koncoch špeciálne štruktúry nazývané teloméry.

Gén je úsek vlákna DNA, ktorý kóduje tvorbu jednej bielkoviny. V tejto práci budeme zobrazovať poradie génov na chromozóme, ale v princípe môžeme zobrazovať nielen gény, ale aj iné úseky DNA.

Genóm je súbor chromozómov, ktorý obsahuje celú genetickú informáciu jedinca. Genóm budeme reprezentovať ako množinu chromozómov, pričom chromozómy budeme reprezentovať ako postupnosť génov respektíve úsekov DNA. Gény reprezentujeme po-

Obr. 1.1: Molekula DNA sa nachádza v chromozóme a má tvar dvojzávitnice zloženej z dvoch komplementárnych vlákien, tvorených nukleotidmi s bázami A,C,G,T. (Zdroj: https://www.genome.gov/glossary/)

mocou ich ID s kladným alebo záporným znamienkom. ID génu je celé číslo, špecifické pre istý gén alebo úsek DNA. Chromozóm môže obsahovať viacero génov s rovnakým ID, čo sú kópie s podobnou sekvenciou. Záporné znamienko udáva fakt, že príslušný gén sa nachádza na opačnom vlákne DNA ako gény s kladným znamienkom.

Evolučná história je postupnosť udalostí - genetických modifikácií, pri ktorých v určitom čase dochádza k zmene genómu. Udalosťou môže byť aj zámena jednej bázy za inú, v práci sa však budeme venovať vizualizácii udalostí, ktoré menia počet alebo umiestnenie génov na chromozómoch. Takými môžu byť napríklad:

- **duplikácia** - skopírovanie jedného alebo viacerých susedných génov (súvislého úseku) na iné miesto v genóme (Príklad: Chromozóm reprezentovaný postupnosťou génov $1, 2, 3, 4$ sa duplikáciou úseku s génmi $2, 3$, ktorá kopírovaný úsek umiestni na koniec chromozómu, zmení na chromozóm s postupnosťou $1, 2, 3, 4, 2, 3.$),
- **inzercia** - vloženie jedného alebo viacerých nových génov (nového úseku) na niektoré miesto v genóme (Príklad: Do chromozómu reprezentovaného postupnosťou génov $1, 2, 3, 4$ sa vloží gén s ID 5 na druhé miesto, čím sa postupnosť génov zmení na $1, 5, 2, 3, 4.$),
- **delécia** - odstránenie jedného alebo viacerých génov (zmazanie súvislého úseku) z genómu (Príklad: Majme opäť chromozóm reprezentovaný postupnosťou génov $1, 2, 3, 4$. Delécia úseku, ktorý obsahuje gén s ID 3 zmení postupnosť génov na $1, 2, 4.$),
- **inverzia** - zmena orientácie génu alebo skupiny susedných génov, t.j. daná sekvencia sa nahradí opačnou sekvenciou. Každý zo skupiny susedných génov nahradíme tým istým génom s opačnou orientáciou a následne otočíme poradie génov v

skupine. (Príklad: Chromozóm s postupnosťou $1, 2, 3, 4$ sa inverziou úseku, ktorý pokrýva gény $2, 3, 4$, zmení na postupnosť génov $1, -4, -3, -2$),

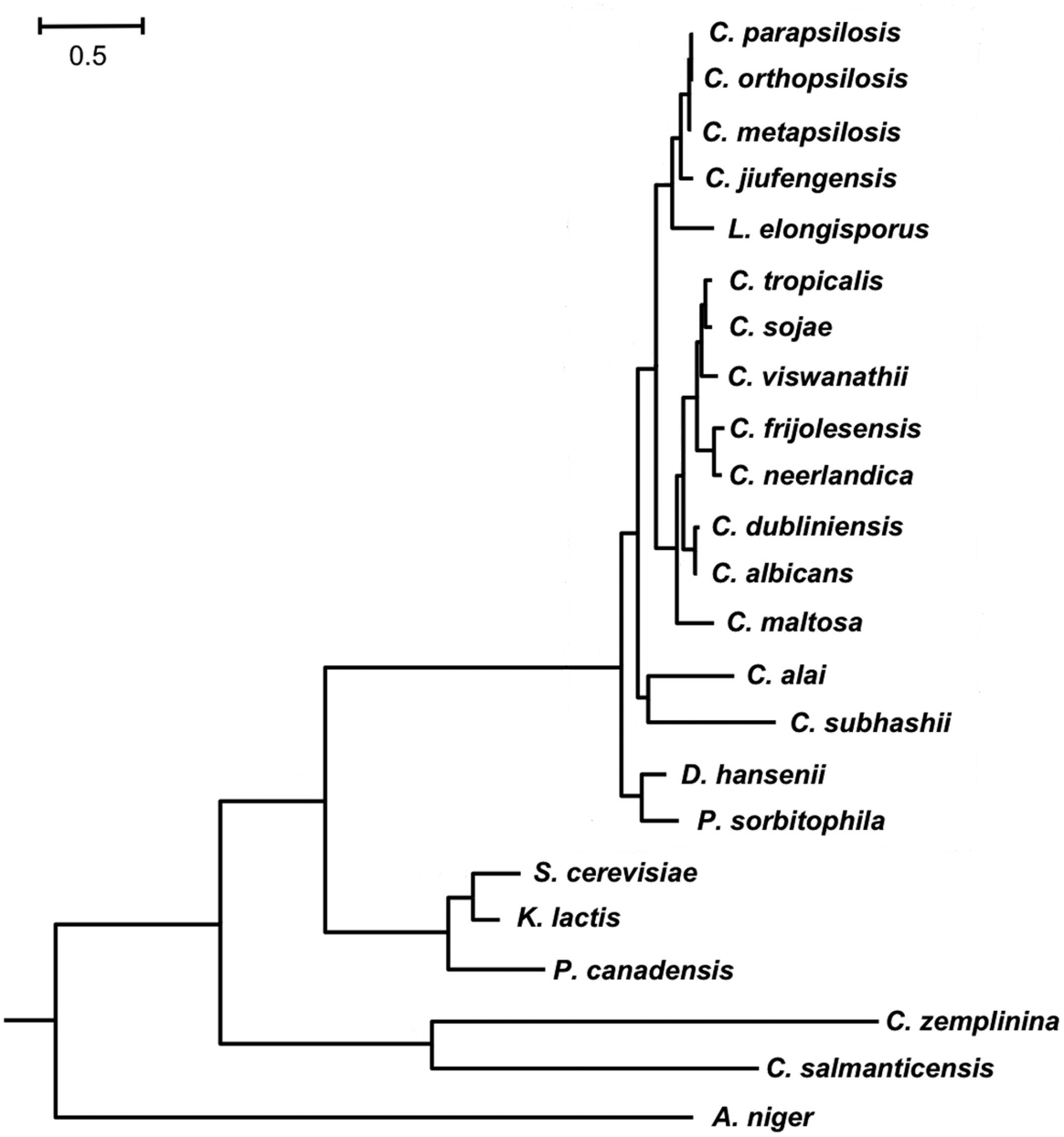
- **speciácia** - udalosť predstavujúca vznik nového druhu. Evolučná história sa vďaka speciáciám rozvetvuje a teda od času nastania každej speciácie obsahuje história taktiež informáciu o zmenách v genóme novovzniknutého druhu.

Fylogenetický strom je grafické zobrazenie evolučných vzťahov viacerých biologických druhov. Listy stromu reprezentujú biologické druhy a vnútorné vrcholy stromu predstavujú spoločných predkov. Ak je známy najbližší spoločný predok celej sady biologických druhov, tak sa nachádza v koreni stromu a strom sa nazýva *zakorenený*. Fylogenetický strom väčšinou nezobrazuje všetky udalosti, ktoré sa udiali v evolučnej histórií, ale zobrazuje všetky speciácie. Zvyčajne strom obsahuje aj informáciu o čase, kedy sa každá zo speciácií udiala, respektíve škáluje dĺžky hrán stromu podľa rozdielu časov medzi speciáciami (viď obrázok 1.2).

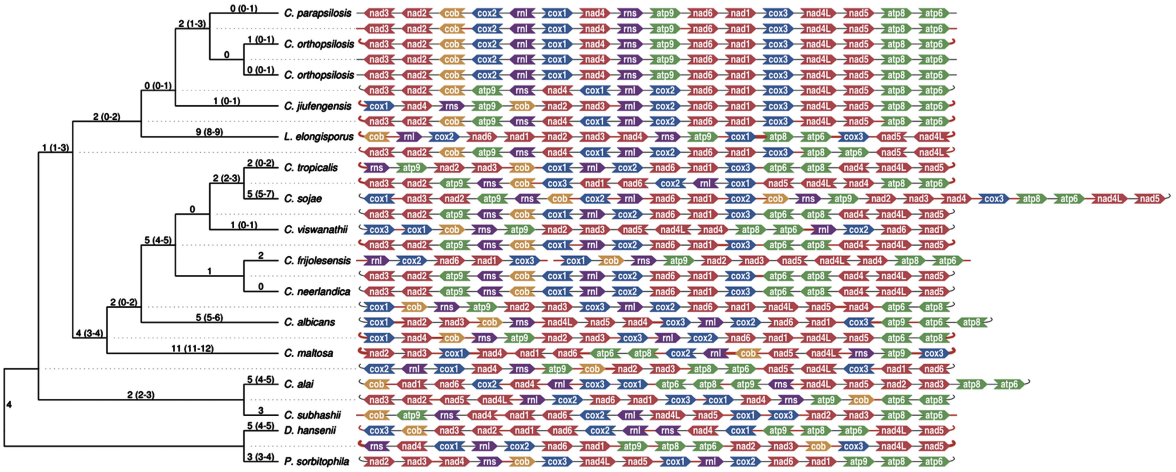
Vizualizáciou evolučnej histórie sa rozumie grafické zobrazenie genómov biologických druhov v každom čase danej evolučnej histórie. Zobraziť genóm bude znamenať vykreslenie génov, ktoré tento genóm obsahuje, v poradí, v akom sa nachádzajú v sekvencií DNA a zobraziť udalosti, ktoré menia poradie alebo počet týchto génov. Gény v genóme môžu byť prípadne rozdelené do viacerých chromozómov. Inými slovami, chceme zobraziť fylogenetický strom, ktorý navyše obsahuje informácie o genóme každého z biologických druhov a ich spoločných predkov a o evolučných zmenách na týchto genómoch.

Príklad možného zobrazenia sa nachádza na obrázku 1.3, kde sa na ľavej strane nachádza kvasinkový fylogenetický strom (z obrázku 1.2) a na pravej strane sú zobrazené genómy (vo forme "vláčikov") pre každý zo súčasných druhov a ich spoločných predkov. Jeden vláčik predstavuje jeden chromozóm spojený z génov, ktoré majú svoj názov a orientáciu. Ak je chromozóm cirkulárny, z oboch strán vláčika sú čiary zakrútené a ak je chromozóm lineárny, tak sú tieto čiary rovné. Z obrázku napríklad vidno, že genómy všetkých zobrazených druhov sú tvorené jedným chromozómom, okrem druhu *C. frijolesensis*, ktorého genóm je tvorený dvoma lineárnymi chromozómami. Takáto vizualizácia síce poskytuje informácie o genómoch druhov a ich predkov, ale neukazuje presne ku akým zmenám medzi jednotlivými genómami došlo.

Existuje viacero nástrojov na vizualizáciu fylogenetických stromov. Väčšina z nich však nezobrazuje udalosti, ktoré sa udiali v genómoch počas evolúcie. Medzi tieto nástroje patria napríklad *FigTree* [1], *phylo.io* [10] alebo *Archaeopteryx* [3]. My sa budeme zaoberať nástrojom EHDraw [12], ktorý na rozdiel od predošlých, zobrazuje aj udalosti, ktoré spôsobili zmeny medzi genómami jednotlivých druhov. Nevýhodou



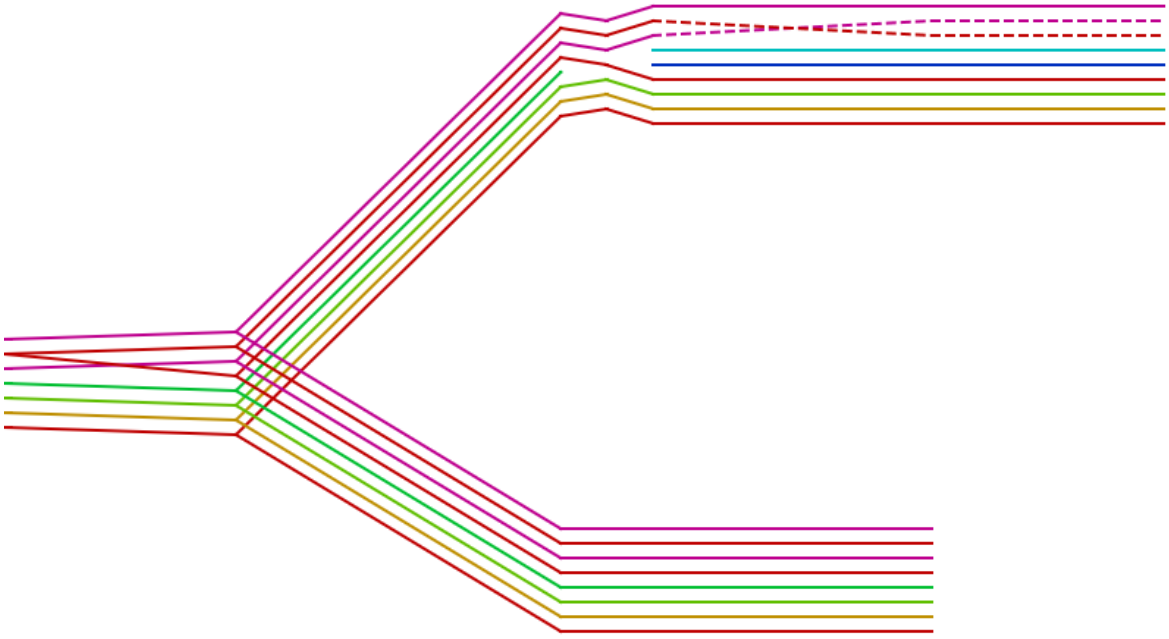
Obr. 1.2: Zakorenený fylogenetický strom, zobrazujúci evolučný vzťah viacerých druhov kvasiniek. (Zdroj: [15])



Obr. 1.3: "Vláčikový"spôsob zobrazenia kvasinkovej evolučnej histórie. (Zdroj: [15])

nástroja je to, že nezobrazuje genómy, ktoré sú tvorené viacerými chromozómami. Výstup nástroja EHDraw možno vidieť na obrázku 1.4. X-ová os predstavuje čas, na y-ovej osi sa nachádzajú gény v takom istom poradí, v akom sa reálne nachádzajú v genómoch. Gény sú na y-ovej osi oddelené menšími medzerami a genómy väčšími medzerami. Jedna čiara na obrázku teda reprezentuje históriu jedného génu. Čiary rovnakej farby predstavujú histórie génov s rovnakým ID (gény sú z hľadiska ich funkcie rovnaké alebo tak podobné, že ich nepotrebujeme na obrázku rozlišovať) a prerušované čiary značia gény s opačnou orientáciou.

Na obrázku 1.4 vidíme evolučnú históriu dvoch druhov. Prvá udalosť, ktorá nastala, bola duplikácia druhého génu a uloženie jeho novej kópie na 4. miesto medzi gény. Druhou udalosťou bola speciácia - rozdelenie genómu na dva genómy s tým istým zoznamom génov. V genóme druhého biologického druhu sa po speciácii už nič neudialo. V genóme prvého druhu nasledovala ihneď po speciácii delécia piateho génu. Potom sa do zoznamu génov vložili dva nové gény (inzercia) a nakoniec nastala inverzia druhého a tretieho génu.



Obr. 1.4: Výstup nástroja EHDraw - evolučná história dvoch druhov.

Kapitola 2

Nástroj EHDraw a jeho rozšírenie

Nástroj na zobrazovanie evolučných histórií, s ktorým budeme pracovať, sa nazýva EHDraw (Evolution History Draw). Tento nástroj vznikol v rámci bakalárskej práce Dávida Simeunoviča [12] v roku 2016 na Katedre informatiky Fakulty matematiky, fyziky a informatiky Univerzity Komenského v Bratislave. Táto kapitola čitateľovi stručne predstaví nástroj EHDraw a zmeny v nástroji, potrebné na účely tejto práce.

2.1 Nástroj EHDraw

Nástroj slúži na vizualizáciu evolučných histórií, kde genóm nie je rozdelený na viacero chromozómov. Jeho vstupom je evolučná história a výstupom je obrázok, ktorý zobrazuje danú evolučnú históriu. Naše rozšírenie tohto nástroja sa týka jeho vstupnej aj výstupnej časti. Pre získanie predstavy o ich pôvodnom stave si obe tieto časti popíšeme.

2.1.1 Vstup

Vstupom nástroja EHDraw je evolučná história v podobe textového súboru. Ukážku jednoduchého vstupu nástroja EHDraw ilustruje obrázok 2.1. Riadok vstupu reprezentuje jednu udalosť v evolučnej histórií pomocou nasledujúcich parametrov:

1. **Názov druhu**
2. **ID udalosti**
3. **ID predošlej udalosti**, ktorá sa udiala bezprostredne pred touto udalosťou, na tej istej línii v rámci fylogenetického stromu
4. **Čas**, v ktorom sa udalosť udiala

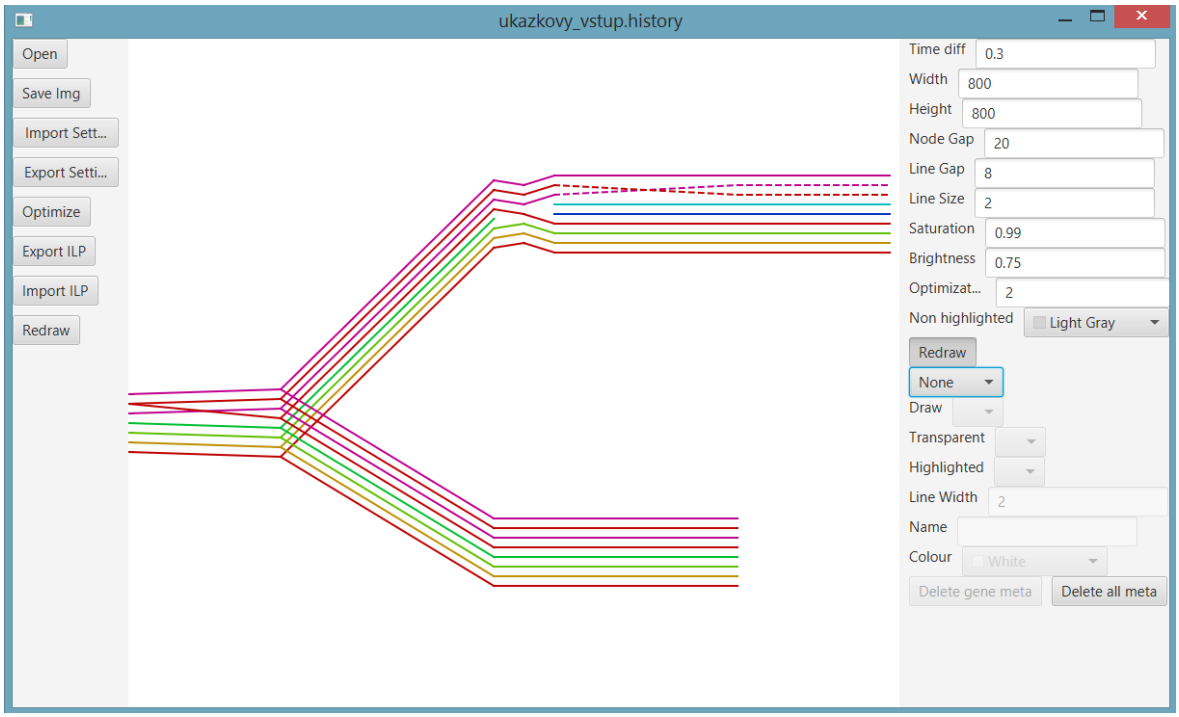
```
predok   e1  root  0     root  1 2 1 5 4 3 2      #  -1 -1 -1 -1 -1 -1 -1
predok   e2  e1    0.05  dup   1 2 1 2 5 4 3 2    #  0 1 2 1 3 4 5 6
clovek   e3  e2    0.12  sp    1 2 1 2 5 4 3 2    #  0 1 2 3 4 5 6 7
clovek   e4  e3    0.13  del   1 2 1 2 4 3 2      #  0 1 2 3 5 6 7
clovek   e5  e4    0.14  ins   1 2 1 6 7 2 4 3 2  #  0 1 2 -1 -1 3 4 5 6
clovek   e6  e5    0.2   inv   1 -1 -2 6 7 2 4 3 2  #  0 2 1 3 4 5 6 7 8
clovek   e7  e6    0.25  leaf  1 -1 -2 6 7 2 4 3 2  #  0 1 2 3 4 5 6 7 8
simpanz  e8  e2    0.12  sp    1 2 1 2 5 4 3 2    #  0 1 2 3 4 5 6 7
simpanz  e9  e8    0.2   leaf  1 2 1 2 5 4 3 2    #  0 1 2 3 4 5 6 7
```

Obr. 2.1: Ukážka vstupu nástroja EHDraw.

5. **Typ udalosti**, ktorý predstavuje jednu z evolučných zmien, ktoré sme si už predstavili v predošlej kapitole (duplikácia, inzercia, delécia, inverzia, speciácia) alebo jednu z udalostí:
   - root - špeciálna udalosť pre koreň
   - leaf - udalosť, ktorá má za cieľ určiť čas skončenia vetvy
   - other - udalosť, ktorú nevieme presne nazvať alebo predstavuje viac udalostí naraz
6. **Zoznam ID všetkých súčasných génov** po tom, ako sa daná udalosť udiala.
7. **Pozície súčasných génov** pred odohraním udalosti t.j. poradové číslo príslušného génu v zozname pre predošlú udalosť. Slúži na určenie všetkých zmien, ktoré sa v genóme udiali. Číslo -1 znamená, že adekvátny gén zo zoznamu génov sa v predkovi nenachádzal, ale v tejto udalosti novo vznikol.

2.1.2 Výstup

Ukážku grafickej aplikácie aj s výstupom je možné vidieť na obrázku 2.2. Výstup sa nachádza na bielej ploche v strede grafickej aplikácie a je výstupom pre vstup z obrázku 2.1. Grafická aplikácia nástroja EHDraw poskytuje rôzne spôsoby ako výstup upraviť. Dáva možnosť prispôsobiť si šírku a výšku výstupného obrázku, šírku čiar, vzdialenosť medzi génmi alebo vetvami, či meniť rôzne nastavenia pre každý gén zvlášť. Taktiež dáva možnosť zobraziť minimálny počet génov tak, aby všetky udalosti v evolučnej histórií ostali viditeľné. Tieto úpravy sú dôležité najmä pri väčších vstupoch, kedy sa celý výstup nemusí zmestiť na obrazovku. Taktiež je možné výstupný obrázok uložiť vo formáte SVG cez tlačítko *Save Img* a následne si obrázok pozrieť alebo upraviť v editore vektorovej grafiky.



Obr. 2.2: Ukážka grafickej aplikácie a výstupu nástroja EHDraw so vstupom z obrázku 2.1.

2.2 Rozšírenie nástroja EHDraw

V tejto časti ukážeme, ako sme rozšírili nástroj EHDraw, aby zobrazoval aj evolučné histórie genómov s viacerými chromozómami.

2.2.1 Vstup

Vo vstupe nastala len jedna zmena a to konkrétne v šiestom stĺpci, t.j. v zozname súčasných génov (viď obr. 2.3). Gény rozdelíme do chromozómov tak, že podzoznam génov patriaci jednému chromozómu zakončíme znakom „$“ alebo „@“. Znak „$“ značí, že predošlé gény patria do chromozómu, ktorý je lineárny a „@“ znamená, že tento chromozóm je cirkulárny. Vstup na obrázku 2.3 teda obsahuje len lineárne chromozómy.

2.2.2 Výstup

Výstup rozšíreného nástroja EHDraw (viď obr. 2.4) vizualizuje evolučnú históriu tak, že pokiaľ je niektorý genóm rozdelený na viacero chromozómov, medzi týmito chromozómami sa zobrazuje medzera. Veľkosť tejto medzery je možné zadefinovať pomocou nastavenia *Chromosome Gap* na pravej strane grafickej aplikácie. Ďalej, aby sme vyznačili, či sú chromozómy v danom genóme cirkulárne alebo lineárne, vyobrazujeme okolo nich obdĺžnik pre lineárne a obdĺžnik so zaoblenými rohmi pre cirkulárne chromozómy.

```
predok  e1  root  0     root  1 2 1 5 $ 4 3 2 $      #  -1 -1 -1 -1 -1 -1 -1
predok  e2  e1    0.05  dup   1 2 1 2 5 $ 4 3 2 $    #  0 1 2 1 3 4 5 6
clovek  e3  e2    0.12  sp    1 2 1 2 5 $ 4 3 2 $    #  0 1 2 3 4 5 6 7
clovek  e4  e3    0.13  del   1 2 1 2 $ 4 3 2 $      #  0 1 2 3 5 6 7
clovek  e5  e4    0.14  ins   1 2 1 6 7 2 $ 4 3 2 $  #  0 1 2 -1 -1 3 4 5 6
clovek  e6  e5    0.2   inv   1 -1 -2 6 7 2 $ 4 3 2 $ #  0 2 1 3 4 5 6 7 8
clovek  e7  e6    0.25  leaf  1 -1 -2 6 7 2 $ 4 3 2 $ #  0 1 2 3 4 5 6 7 8
simpanz e8  e2    0.12  sp    1 2 1 2 5 $ 4 3 2 $    #  0 1 2 3 4 5 6 7
simpanz e9  e8    0.2   leaf  1 2 1 2 5 $ 4 3 2 $    #  0 1 2 3 4 5 6 7
```

Obr. 2.3: Ukážka vstupu rozšíreného nástroja EHDraw. Genómy sú tu v každom bode histórie tvorené dvoma lineárnymi chromozómomami.

V grafickej aplikácii pribudli tiež tlačítka *Minimize Crossings* a *Export history*. Tlačítko *Minimize Crossings* minimalizuje kríženia na výstupe a *Export history* umožňuje uložiť evolučnú históriu po minimalizácií krížení do textového súboru (vo vstupnom formáte pre EHDraw). O minimalizácii krížení si povieme viac v nasledujúcich kapitolách.

Poznámka: Optimalizácia výstupu implementovaná v pôvodnom programe (vykreslenie minimálneho počtu génov, aby všetky udalosti ostali viditeľné), nie je vždy kompatibilná s novou verziou EHDraw, pretože môže spojiť viaceré chromozómy do jedného, napr. keď v niektorom z chromozómov nedošlo v histórii k žiadnej udalosti.

2.2.3 Ovládanie cez príkazový riadok

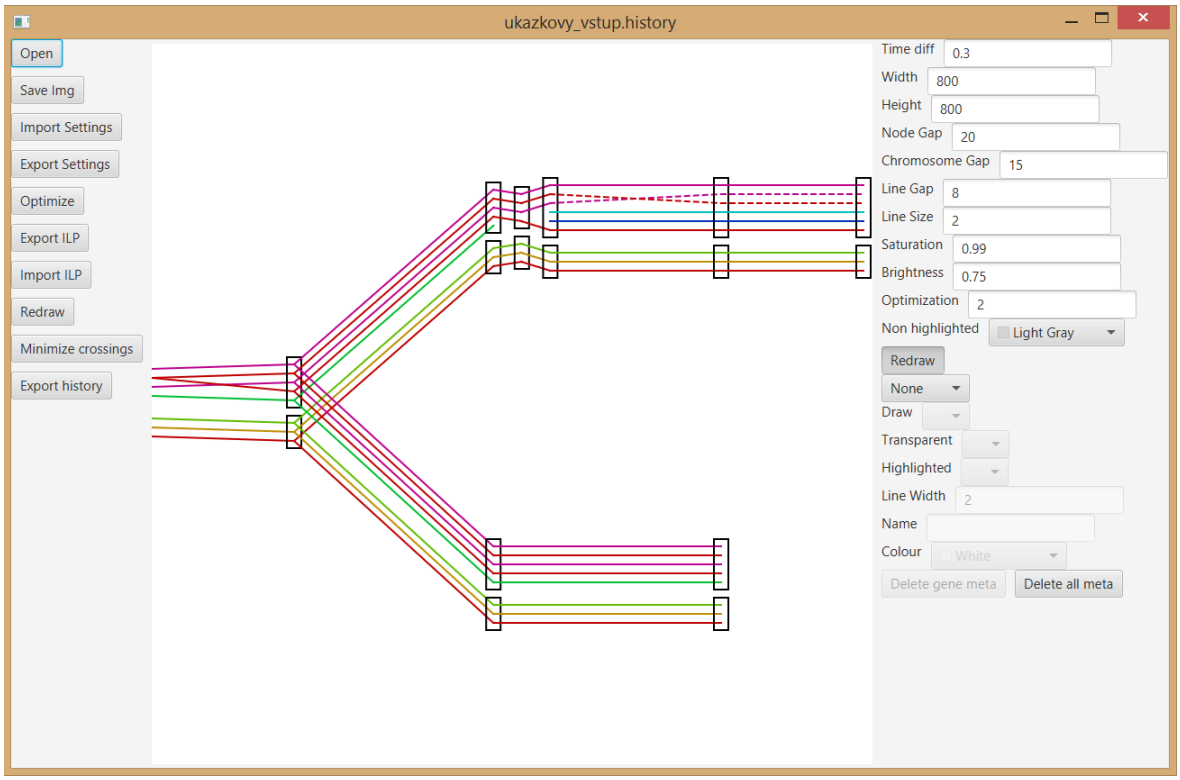
Nástroj EHDraw poskytuje aj možnosť ovládania cez príkazový riadok, t.j. bez použitia grafickej aplikácie. Takýto spôsob ovládania je využiteľný najmä ak chceme použiť niektorú z funkcionalít nástroja na viacero vstupov. V rozšírení pribudli medzi argumenty príkazového riadku nasledovné nastavenia:

-minimized_ output:filepath.history, ktorý určí, do akého súboru sa vypíše nová história po minimalizácií krížení,

-exportminimized, ktorý minimalizuje kríženia vstupnej evolučnej histórie a vypíše zmenenú históriu buď do predvoleného súboru *minimized.history* alebo do súboru špecifikovaného v argumente *-minimized_ output*,

-crossings_ output:filepath.txt, ktorý určí, do akého súboru sa vypíše počet krížení danej evolučnej histórie,

-exportcrossings, ktorý vypíše počet krížení histórie buď do predvoleného súboru *crossings.txt* alebo do súboru špecifikovaného argumentom *-crossings_ output*. Ak sa tento argument použije zároveň s argumentom *-exportminimized*, vypíše sa počet krížení histórie po minimalizácií.



Obr. 2.4: Ukážka výstupu rozšíreného nástroja EHDraw so vstupom z obrázku 2.3

2.2.4 Triedy a dátové štruktúry

Nástroj EHDraw je implementovaný v jazyku Java a medzi triedy, ktoré tvoria štruktúru programu, patria:

- *EHDraw* - hlavná trieda, ktorá načíta vstupné dáta na základe dodaných argumentov príkazového riadku a vytvára grafickú aplikáciu,
- *EvolutionTree* - trieda reprezentujúca evolučnú históriu v podobe binárneho stromu, ktorá obsahuje vnorenú triedu *EvolutionNode* predstavujúcu jednu udalosť v evolučnej histórií (jeden riadok vstupu),
- *DrawFactory* - abstraktná trieda návrhového vzoru Factory, ktorej podtriedy SVGDrawFactory a FXDrawFactory vykresľujú výstup vo formáte SVG obrázku alebo do grafickej aplikácie JavaFX,
- *Settings* - trieda obsahujúca globálne premenné a nastavenia,
- *GeneMeta* - trieda obsahujúca nastavenia pre konkrétny gén.

V rámci rozšírenia došlo ku zmenám len v triedach *EHDraw* a *EvolutionTree* a pridali sme novú triedu *Chromosome*, ktorá predstavuje jeden chromozóm v genóme po odohraní niektorej z udalostí. Všetky algoritmy pre minimalizáciu krížení sme implementovali ako metódy triedy *EvolutionTree*. Zároveň vznikli nové pomocné triedy s

metódou main - *HistoryRandomizer* a *StatisticsGenerator*. O ich funkcionalite si povieme viac v kapitole 5.

Kapitola 3

Vizualizácia grafov

V tejto časti sa zoznámime s grafovou terminológiou a zásadami, ktoré zvyšujú prehľadnosť a čitateľnosť grafov pri ich vizualizácii. Nakoniec si predstavíme problém minimálneho kríženia a algoritmy, ktoré tento problém riešia.

3.1 Pojmy z teórie grafov

Pre úplnosť spomenieme pár definícií z teórie grafov, ktoré vystupujú v tejto kapitole. Graf je tvorený množinou vrcholov a množinou hrán. Vrcholy reprezentujú určité entity a hrany predstavujú vzťahy medzi týmito entitami. V *orientovanom grafe* majú hrany navyše priradený smer. *Acyklický orientovaný graf* (*DAG* - z angl. directed acyclic graph) je graf s orientovanými hranami, ktorý neobsahuje žiadny orientovaný cyklus.

Vrstvový graf (layered graph) je DAG, v ktorom je množina vrcholov V rozdelená do vrstiev $V_1, V_2, ..., V_k$ takých, že $V = V_1 \cup V_2 \cup ... \cup V_k$ a pre $i \neq j$ platí, že $V_i \cap V_j = \emptyset$. Všetky hrany v tomto grafe smerujú rovnakým smerom, konkrétne z vrcholov vrstvy V_i do vrcholov vrstvy V_j, pre $i < j$.

Riadny vrstvový graf (proper layered graph) je vrstvový graf, ktorý má navyše vlastnosť, že všetky jeho hrany spájajú vždy len vrcholy zo susedných vrstiev V_i a V_{i+1}.

3.2 Hierarchické kreslenie grafov

V mnoho prípadoch orientovaný graf predstavuje akúsi hierarchiu a preto ho chceme hierarchickým spôsobom aj vykresliť. Príklady takýchto hierarchií sú napríklad diagramy tried objektovo-orientovaného jazyka, grafy volania funkcií, či fylogenetické stromy. Pri vizualizácii týchto grafov chceme dbať na to, aby ich grafická reprezentácia bola čo najprehľadnejšia. Najpopulárnejšou metódou vykresľovania orientovaných grafov je Sugiyamova metóda [13], ktorá rozdeľuje vrcholy do vrstiev. Táto metóda

dbá na to, aby boli splnené nasledujúce vlastnosti, ktoré robia graf prehľadnejším:

- všetky hrany smerujú tým istým smerom
- hrany sú čo najkratšie a sú vykreslené pomocou rovných čiar
- hrany sa navzájom čo najmenej krížia
- vrcholy sú na obrázku rozmiestnené rovnomerne

Sugiyama navrhuje nasledovný postup (viď obr. 3.1). Majme ľubovoľný orientovaný graf. Odstránením cyklov získame DAG, následne sa vrcholy rozdelia do vrstiev, čím získame vrstvový graf. Vrcholy každej vrstvy grafu budú zobrazené na jednej vodorovnej úrovni. Tento graf však ešte nespĺňa podmienku riadneho vrstvového grafu, lebo obsahuje aj tzv. „dlhé hrany“, ktoré prechádzajú cez viacero vrstiev. Takýchto hrán sa zbavíme, keď na každej vrstve, ktorou dlhá hrana prechádza, dodefinujeme nový pomocný vrchol. Keď už máme graf v podobe riadneho vrstvového grafu, môžeme preusporiadať vrcholy vo vrstvách za účelom minimalizácie počtu križení hrán grafu.
Poznámka: Vrstvy môžu byť vykresľované zhora nadol alebo zľava doprava. V našom programe sa vrstvami prechádza zľava doprava. To znamená, že vrcholy v jednej vrstve majú rovnakú x-ovú súradnicu a teda pre nás budú zaujímavé len ich y-ové súradnice.

3.3 Problém minimálneho počtu križení

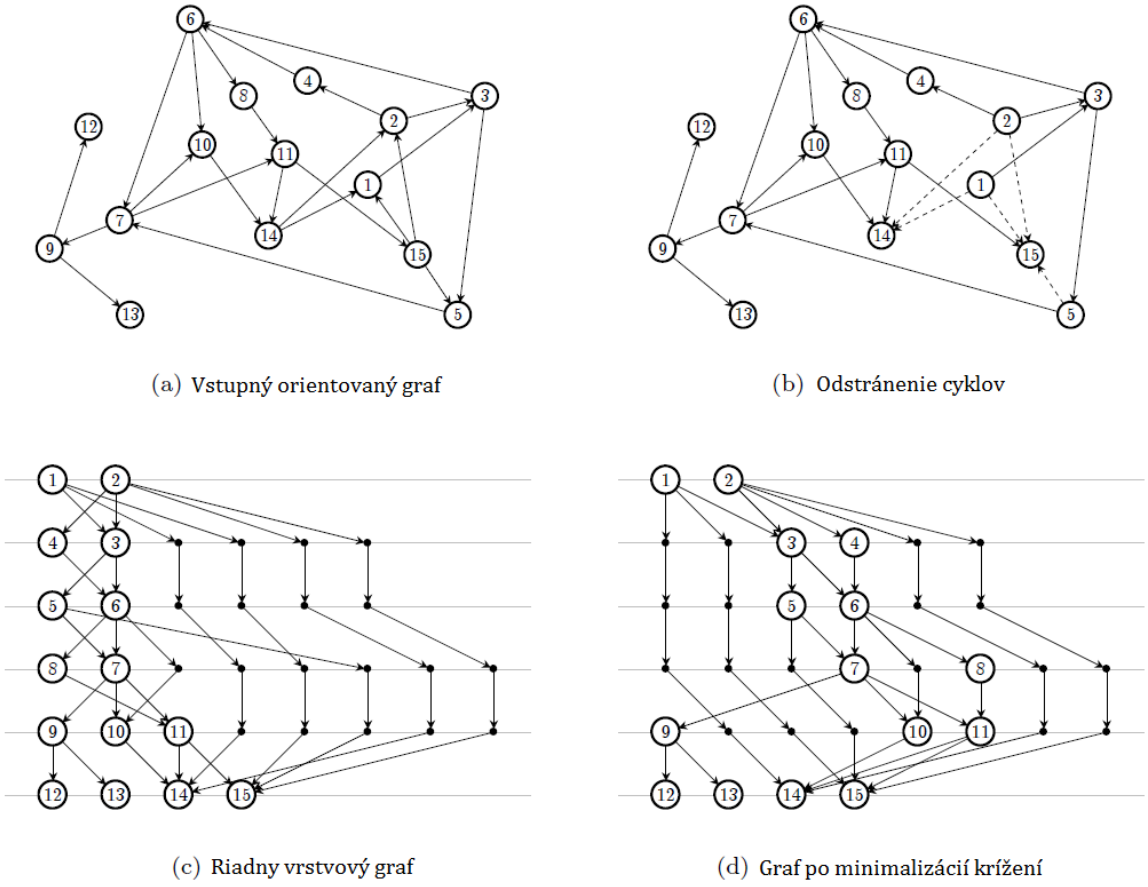
Definícia problému: Majme riadny vrstový graf G s vrstvami $V_1, ..., V_k$. Nájdite poradie vrcholov vo vrstvách $V_1, ..., V_k$ také, aby počet križení hrán v G bol minimálny.

Minimalizácia počtu križení je problém, ktorý nevieme v rozumnom čase vyriešiť. Preto sa musíme obrátiť na heuristické riešenia, ktoré síce neposkytnú optimálne riešenie, ale zbehnú v rozumnom čase a značne zmenšia počet križení na grafe. Najskôr si popíšeme algoritmus na spočítanie križení, následne predstavíme možné spôsoby riešenia problému na dvoch vrstvách a viac-vrstvových grafoch.

Počítanie počtu križení

Počet križení celého vrstvového grafu získame tak, že spočítame kríženia medzi každými dvoma susednými vrstvami grafu a výsledkom bude suma týchto čiastkových križení.

Jednoduchý algoritmus, ktorý spočíta kríženia medzi dvoma vrstvami má časovú zložitosť $\mathcal{O}(|E|^2)$, kde E je množina hrán medzi dvoma vrstvami. Algoritmus porovnáva



(a) Vstupný orientovaný graf

(b) Odstránenie cyklov

(c) Riadny vrstvový graf

(d) Graf po minimalizácií krížení

Obr. 3.1: Znázornenie Sugiyamovej metódy (Zdroj: [14])

všetky hrany medzi dvoma vrstvami. Začnime hranami, ktoré vychádzajú z vrcholu s najmenšou y-ovou súradnicou prvej vrstvy. Označme y-ovú súradnicu vrchola, z ktorého práve porovnávaná hrana vychádza y_0 a do ktorého vchádza y_1. Následne sa pozrieme na všetky hrany, ktoré vychádzajú z vrcholov s y-ovou súradnicou väčšou ako y_0 a za každú hranu, ktorá vchádza do vrcholu s menšou y-ovou súradnicou ako y_1 pripočítame jedno kríženie.

Existujú aj komplikovanejšie algoritmy s menšími časovými zložitosťami [11, 9], tými sa však v tejto práci zaoberať nebudeme.

3.3.1 Jednostranný minimalizačný problém

Zjednodušenou verziou problému minimálneho počtu križení na celom grafe je minimalizovať kríženie hrán iba medzi dvoma jeho vrstvami. Konkrétnejšie, popíšeme variant, keď jedna z vrstiev má pevné poradie vrcholov. Riešenia tohto problému sa využívajú pri heuristickom riešení problému na celom grafe.

Definícia problému: Majme dvoj-vrstvový (bipartitný) graf s vrstvami L_0 a L_1, s daným poradím vrcholov na vrstve L_0. Vyberte poradie vrcholov na vrstve L_1 tak, aby počet križení medzi hranami bol minimálny.

Už tento problém je NP-ťažký [4]. Predstavíme si dve heuristiky, ktoré ho aproximujú. Popis ďalších heuristík môže čítateľ nájsť v literatúre [14].

Barycentrová heuristika

Najbežnejšie heuristiky pre tento problém sú obmeny barycentrickej metódy, ktorá pracuje nasledovne: y-ová súradnica každého vrcholu u z L_1 sa spočíta ako barycentrum (aritmetický priemer) y-ových súradníc susedov vrcholu u.

Takto spočítané y-ové súradnice vrcholov z vrstvy L_1 však nemusia byť použité priamo. Môžu slúžiť len ako „skóre“ pre každý z vrcholov a podľa tohto skóre sa potom vrcholy vzostupne zoradia.

Je dokázané, že počet križení po zbehnutí tejto heuristiky je najviac $\mathcal{O}(\sqrt{|L_0|})$ krát horší ako optimum [4].

Mediánová heuristika

Ďalšou, podobnou heuristikou je mediánová heuristika. Funguje tak, že y-ové súradnice vrcholu u z vrstvy L_1 vyberieme podľa mediánu y-ových súradníc susedov u.

Táto heuristika garantuje výsledný počet križení nanajvýš 3 krát horší ako optimum [4]. Aj keď je tento odhad lepší ako najlepší známy odhad pre barycentrovú heuristiku, v praxi sa barycentrová správa lepšie ako mediánová. [14]

Obe heuristiky (mediánová aj barycentrová) majú vlastnosť, že ak existuje také usporiadanie vrcholov, že graf nemá žiadne kríženia, tak toto usporiadanie nájdu. Spo-

čítanie skóre pre vrchol v má časovú zložitosť $\mathcal{O}(|N(v)|)$, kde $N(v)$ je množina susedov vrcholu v. Pre všetky vrcholy je teda zložitosť $\mathcal{O}(|E|)$, kde E je množina hrán medzi vrstvami L_0 a L_1. Usporiadanie vrcholov podľa vypočítaného skóre má časovú zložitosť $\mathcal{O}(|L_1| \cdot \log |L_1|)$.

3.3.2 Viac-vrstvový minimalizačný problém

Optimálne riešenie tohto problému permutuje vrcholy vo všetkých vrstvách naraz. Kvôli veľkej časovej zložitosti sa zvyčajne optimálne riešenie nehľadá. Najbežnejšou heuristikou je **prechod po vrstvách** (layer-by-layer sweep). Algoritmus prejde od prvej vrstvy po poslednú tak, že na vrstvách V_i a V_{i+1} vyrieši jednostranný minimalizačný problém, kde V_i je vrstva s pevným poradím vrcholov. Následne algoritmus prejde z poslednej do prvej vrstvy a po ceste znova rieši jednostranný minimalizačný problém na dvojiciach susedných vrstiev, kde má pevné poradie vrcholov vrstva s vyšším indexom. Celý algoritmus sa opakuje až kým nebude možné počet krížení viac vylepšiť. Inú heuristiku možno nájsť v literatúre [8].

Kapitola 4

Minimalizácia krížení evolučnej histórie

V druhej kapitole sme si ukázali rozšírený nástroj EHDraw. Umožňuje zobrazenie evolučných histórií, v ktorých je zobrazovaný genóm rozdelený na niekoľko chromozómov. Tieto chromozómy sa vykresľujú v poradí, v akom sú vo vstupnom súbore. Pretože genóm je tvorený množinou chromozómov, poradie ich vykresľovania môže byť ľubovoľné. Evolučné zmeny môžu prebiehať na génoch z rôznych chromozómov, čo znamená, že vo výstupe nášho nástroja existujú čiary, ktoré prechádzajú z jedného chromozómu do druhého. To môže spôsobiť, že výstupný obrázok bude obsahovať množstvo prekrížených čiar, čím sa zníži prehľadnosť výstupu. V tejto kapitole si uvedieme spôsoby, ako obmeniť a preusporiadať chromozómy v každom bode histórie tak, aby výstup obsahoval čo najmenej krížení čiar, ale aby stále predstavoval tú istú históriu.

Opíšeme graf evolučnej histórie, optimalizáciu počtu krížení v tomto grafe si rozdelíme na tri podproblémy a ich riešenia potom spojíme. Tieto tri podproblémy budú konkrétne problém poradia chromozómov, problém otáčania chromozómov a problém rezania cirkulárnych chrozómov. Spojením riešení týchto podproblémov získame algoritmus na celkovú optimalizáciu a tento algoritmus aj implementujeme.

Označenia

Predtým ako opíšeme graf, jednotlivé optimalizačné podproblémy a ako sme sa s nimi v práci vysporiadali, zaveďme si všeobecné označenia pre našu evolučnú históriu. Tieto označenia budú vystupovať najmä v odhadoch zložitosti algoritmov.

Ako M si označíme počet všetkých udalostí v evolučnej histórii. Budeme predpokladať, že všetky genómy sú približne rovnako veľké a maximálny počet génov v genóme označíme ako N. Do každého génu vchádza zľava najviac jedna čiara, počet čiar medzi dvoma susednými udalosťami histórie je teda tiež najviac N. Písmenom n označíme maximálny počet chromozómov v jednom genóme a C_{max} bude označovať počet génov v najväčšom chromozóme.

4.1 Graf evolučnej histórie

V predošlej kapitole sme si predstavili pojem riadneho vrstvového grafu a viaceré heuristiky optimalizujúce počet krížení na takom grafe. V tejto časti opíšeme graf, ktorý reprezentuje výstup EHDraw a ktorý sa v mnoho aspektoch podobá na riadny vrstvový graf.

Za vrstvy grafu budeme považovať genómy po odohraní niektorej evolučnej udalosti a vrcholy grafu budú gény, ktoré sa v týchto genómoch nachádzajú. Susednými vrstvami označujeme vrstvy, ktoré sú vo vzťahu predok-potomok. Kvôli speciáciám sa vrstvy vetvia a niektorí predkovia majú dvoch potomkov. Hrany spájajú gény zo susedných vrstiev práve vtedy, keď gén z genómu predka v rámci evolučnej udalosti, ktorá sa odohrala medzi týmito susednými vrstvami, buď zostal v genóme potomka na tej istej pozícii, alebo sa presunul na novú pozíciu, alebo sa dokonca skopíroval na viacero miest. Všetky hrany smerujú od predkov k potomkom.

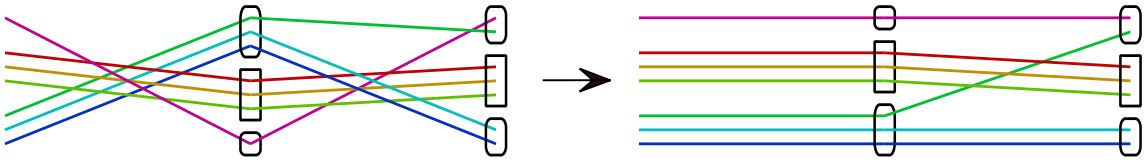
Kvôli väčšej prehľadnosti obrázku nebudeme v našich algoritmoch minimalizovať iba počet krížení, ale skóre, do ktorého zarátavame okrem počtu krížení hrán aj každý gén s opačnou orientáciou v každej vrstve grafu a taktiež každú zmenu orientácie génov spojených hranou. Cieľom tejto úpravy je minimalizovať čiarkované čiary, ktoré pôsobia rušivo.

Počítanie skóre

Skóre stromu získame tak, že rekurzívne prechádzame strom a počítame skóre medzi každými dvoma susednými vrstvami. Kríženia, ktoré vznikajú pri speciáciách medzi hranami vychádzajúcimi z predka do rôznych potomkov sa do celkového skóre nezapočítavajú. Na spočítanie počtu skóre medzi dvoma vrstvami sme implementovali jednoduchý algoritmus podobný algoritmu opísanému v predošlej kapitole, ktorý má zložitosť $\mathcal{O}(N^2)$. Skóre vytvorené génmi jedného chromozómu, spočítame so zložitosťou $\mathcal{O}(C^2)$, kde C je počet génov v danom chromozóme. Tento výpočet využijeme v druhom a treťom podprobléme.

4.2 Problém poradia chromozómov

Vyššie definovaný graf sa od riadneho vrstvového grafu odlišuje tým, že pri optimalizácii počtu krížení nemôžme poradie jeho vrcholov (génov) ľubovoľne meniť. Poradie chromozómov však meniť môžeme. Problémom poradia chromozómov nazveme problém, ktorý má za úlohu preusporiadať chromozómy v každej vrstve grafu evolučnej histórie tak, aby počet krížení tohto grafu bol minimálny. Na základe podobnosti nášho grafu a riadneho vrstvového grafu môžeme na riešenie tohto problému použiť heuristiky podobné s heuristikami, ktoré sa používajú na problém minimálneho počtu krížení pre



Obr. 4.1: Priebeh barycentrovej heuristiky. Zmena medzi prvým a druhým obrázkom je len v poradí chromozómov v prostrednej vrstve, ktorá sa optimalizovala vzhľadom na jej predka, t.j. vrstvu vľavo. Pre každý z chromozómov sa vypočíta barycnetrové skóre ako aritmetický priemer pozícií génov (vchádzajúcich do daného chromozómu) v predkovi. Potom sa chromozómy podľa tohto skóre zoradia.

riadny vrstvový graf.

Jednostranná optimalizácia

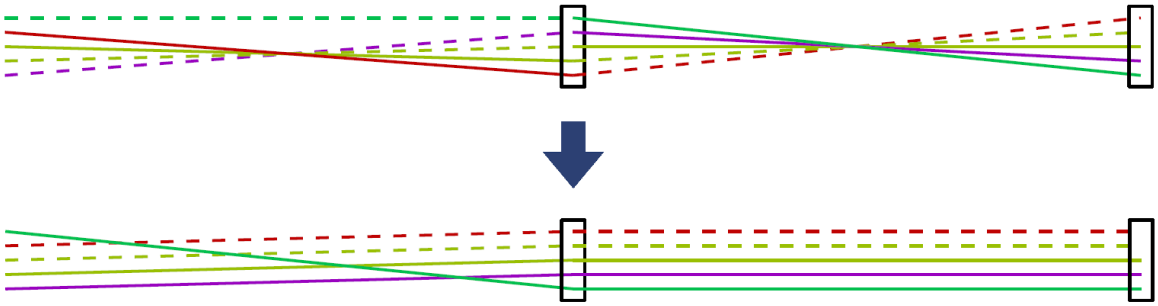
Ako riešenie jednostrannej optimalizácie sme pri probléme poradia chromozómov použili miernu obmenu barycentrovej heuristiky. Konkrétne, pre každý chromozóm voľnej vrstvy vypočítame barycentrové skóre tak, že spriemerujeme pozície všetkých génov z fixovanej vrstvy, ktoré sú spojené s génmi daného chromozómu z voľnej vrstvy. Následne chromozómy podľa tohto skóre usporiadame. Ak sa na voľnej vrstve udiala speciácia, optimalizujeme ju podľa dvoch fixných vrstiev, ktoré predstavujú jej potomkov. V tomto špeciálnom prípade počítame skóre chromozómov ako priemer prvého a druhého barycentrového skóre, ktoré by chromozóm získal za prvú a druhú fixnú vrstvu. Výsledný algoritmus má časovú zložitosť $\mathcal{O}(\max(N, n \log n))$.

Malý príklad preusporiadania chromozómov podľa barycentrovej heuristiky možno vidieť na obrázku 4.1.

Prechod po vrstvách

Ako základný algoritmus pre minimalizáciu skóre na všetkých vrstvách grafu sme implementovali pre všetky podproblémy heuristiku prechodu po vrstvách. V tejto heuristike rekurzívne prejdeme celý vrstvový strom a medzi každými dvoma susednými vrstvami aplikujeme jednostrannú optimalizáciu.

Začneme od koreňa stromu, koreň zvolíme ako fixnú vrstvu, jeho potomok bude voľná vrstva, teda vrstva, ktorej vrcholy (chromozómy) permutujeme. Ak má vrchol dvoch potomkov, spustíme rekurziu na oba podstromy, ktorých koreňmi sú títo potomkovia. Keď v určitej vetve výpočtu zbehne jednostranná optimalizácia, kde voľná vrstva je list stromu, začneme sa vracať späť k najbližšej speciácii. Zafixovanými vrstvami teraz budú potomkovia a voľné vrstvy budú predkovia. Na vrstvu, v ktorej sa udiala speciácia, sa aplikuje verzia jednostrannej optimalizácie, kde sú dve fixované vrstvy. Poslednou operáciou je aplikovanie jednostrannej optimalizácie na koreň, vzhľadom na jeho potomkov. Jeden prechod tejto rekurzie cez graf má časovú zložitosť $\mathcal{O}(Mt)$, kde



Obr. 4.2: Optimalizácia otáčaním chromozómov. Jediný rozdiel medzi prvým a druhým obrázkom je ten, že sa otočil prostredný chromozóm.

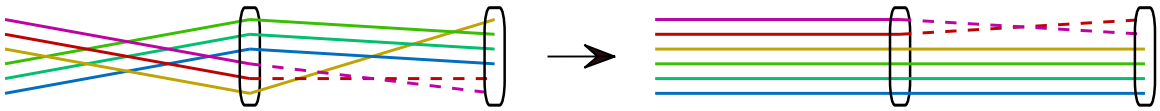
t je čas behu jednostrannej optimalizácie.

Po každom prejdení rekurzie celou evolučnou históriou spočítame jej celkové skóre. Prechod po vrstvách opakujeme až dovtedy, kým skóre po dvoch optimalizáciách histórie po sebe nebude rovnaké. Väčšina vstupov týmto spôsobom konverguje ku istej hodnote skóre, ale môže aj nastať prípad, že po niektorom prejdení algoritmu históriou sa skóre zväčší . Preto si v implementácií udržiavame v pamäti strom s najmenším nájdeným skóre a prechod po vrstvách opakujeme kým hodnota skóre neskonverguje alebo kým sa päťkrát nestane to, že prejdenie rekurzie zhorší predchádzajúci výsledok.

4.3 Problém otočenia chromozómov

Každý chromozóm pozostáva z dvoch komplementárnych vlákien DNA. Z postupnosti génov chromozómu vytvoríme ekvivalentnú postupnosť na komplementárnom vlákne tak, že otočíme poradie génov a zmeníme orientáciu všetkých génov na opačnú. Tieto dve postupnosti sú ekvivalentnými zápismi toho istého chromozómu. Napríklad chromozóm reprezentovaný postupnosťou génov $1, -2, 4, 3, -3$ môžeme rovnako reprezentovať aj postupnosťou $3, -3, -4, 2, -1$. Otočenie chromozómu v histórií môže viesť k menšiemu počtu križení hrán, zbaveniu sa génov s opačnou orientáciou (čiarkovaných čiar) a aj k redukcii počtu zmien orientácie génov v evolučnej histórii (viď. obrázok 4.2). Otáčaním redukujeme len počet takých križení, ktoré sú vytvorené medzi génmi daného chromozómu. Na kríženia spôsobené génmi z iných chromozómov teda otáčanie vplyv nemá.

Pre každý chromozóm sú len dve možnosti: otočiť alebo neotočiť. Pri jednostrannej optimalizácii pre každý chromozóm voľnej vrstvy spočítame skóre pred aj po jeho otočení a podľa toho sa rozhodneme, či ho nakoniec otočíme. Tým získame optimálne riešenie tejto jednostrannej optimalizácie. Časová zložitosť je $\mathcal{O}(n{C_{max}}^2)$.



Obr. 4.3: Malá evolučná história pred a po optimalizácii rezania chromozómov. Prostredný chromozóm sa prerezal v strede (pod tretím génom) a následne sa posledný chromozóm prerezal pod jeho štvrtým génom.

4.4 Problém rezania cirkulárnych chromozómov

Každý cirkulárny chromozóm si v našej reprezentácii môžeme predstaviť ako gény uložené v tvare kruhu. My ich v našom nástroji vykresľujeme rovnako ako lineárne chromozómy - pod sebou. Cirkulárny chromozóm môžeme teda „prerezať“ medzi ľubovoľnými dvoma jeho génmi, vďaka čomu ho vykreslíme ako lineárny. V tomto probléme chceme v každom cirkulárnom chromozóme zvoliť miesto prerezania tak, aby sme minimalizovali počet kríženi. Rezanie, rovnako ako otáčanie, minimalizuje len kríženia génov vychádzajúcich z chromozómu, ktorý režeme. Ukážku rezania v našom programe možno vidieť na obrázku 4.3.

Ak má určitý cirkulárny chromozóm na voľnej vrstve C génov, môžeme ho prerezať na C rôznych miestach. Implementovali sme riešenie, ktoré pre každú z možností spočíta kríženia medzi génmi tohto chromozómu a vyberie najlepšiu možnosť. Algoritmus pre jednostrannú optimalizáciu má tým pádom časovú zložitosť $\mathcal{O}(n{C_{max}}^3)$.

4.5 Celková optimalizácia

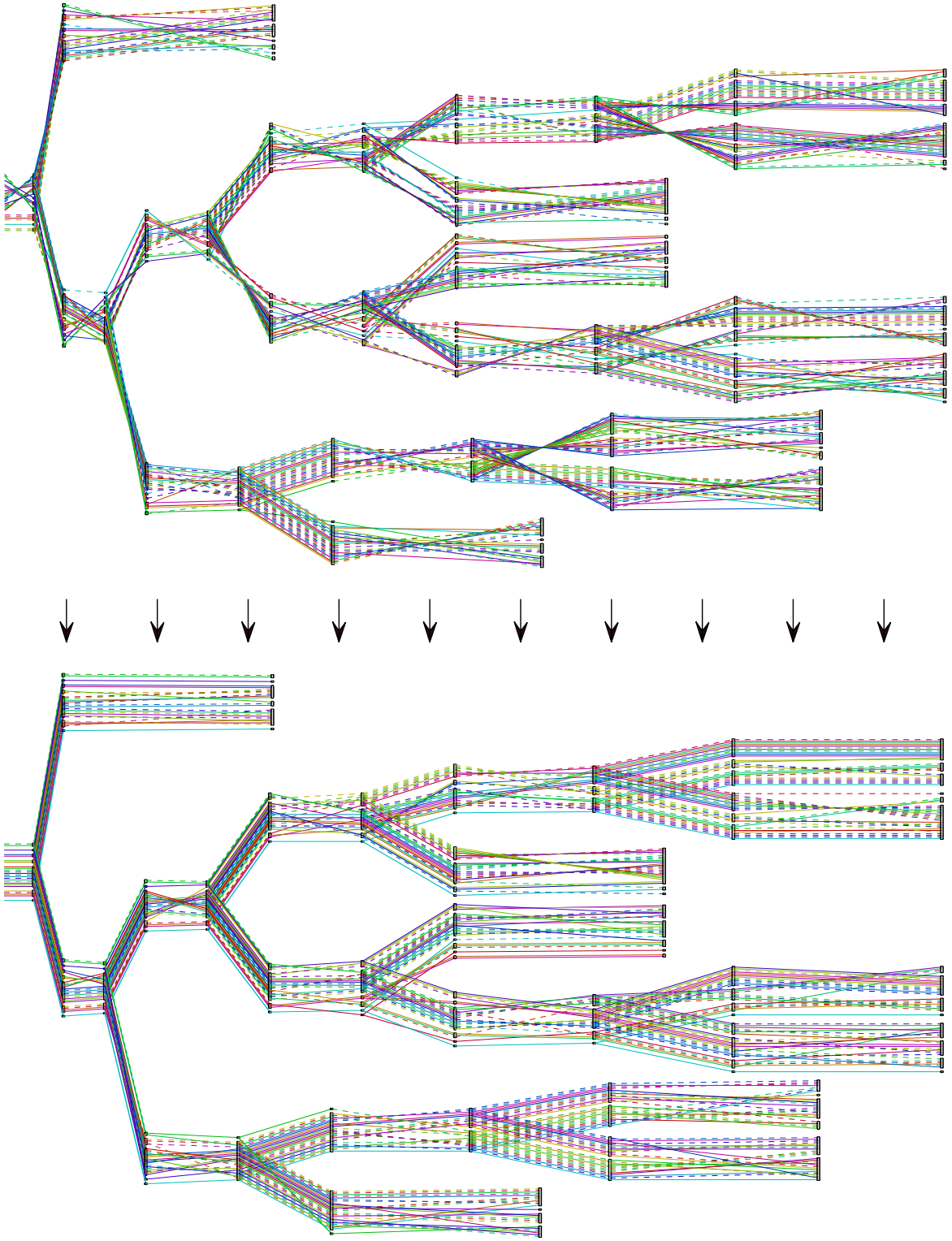
Cieľom celkovej optimalizácie je spojiť riešenia predchádzajúcich troch problémov do jedného algoritmu. Keďže sú predošlé problémy postavené na princípe prechodu cez vrstvy s jednostrannou optimalizáciou medzi dvoma vrstvami, stačí problémy spojiť v jednostrannej optimalizácii.

Otáčanie a rezanie chromozómov možno spojiť pomerne priamočiaro. Pozrieme sa na chromozóm a ak je lineárny, tak len skúsime, či sa oplatí ho otočiť. Ak je chromozóm cirkulárny, pri každom jeho možnom prerezaní sa rozhodneme, či ho chceme otočiť, zapamätáme si skóre pre toto prerezanie (resp. prerezanie + otočenie) a nakoniec vyberieme z nich to s najmenším skóre. Tento algoritmus sme v implementácii použili v jednostrannej optimalizácii pri prechode po vrstvách len jedným smerom.

Heuristiku jednostranného problému, ktorá preusporiadava chromozómy, môžeme zakomponovať pred alebo po optimálnom otočení a prerezaní všetkých chromozómov z vrstvy. Otáčanie a rezanie minimalizuje kríženia spôsobené krížením génov toho istého chromozómu a heuristika minimalizuje kríženia génov vychádzajúcich z rôznych

chromozómov. Vďaka tomu môžeme prísť k záveru, že na poradí týchto dvoch úkonov nezáleží.

Na obrázku 4.4 možno vidieť zobrazenie väčšieho fylogenetického stromu pred a po zbehnutí takto spojeného algoritmu.



Obr. 4.4: Optimalizácia na vygenerovanej, náhodne premiešanej histórii.

Kapitola 5

Experimentálne vyhodnotenie

V predošlej kapitole sme popísali implementáciu algoritmu minimalizujúceho počet krížení v grafe evolučnej histórie. Aby sme algoritmus čiastočne analyzovali, v tejto kapitole popíšeme správanie algoritmu na základe výsledkov testov na generovaných aj reálnych dátach. V testoch sledujeme skóre rôznych evolučných histórií pred aj po zbehnutí optimalizačného algoritmu. Skóre zahŕňa okrem počtu krížení aj počet génov s opačnou orientáciou a počet zmien orientácie medzi génmi susedných vrstiev v evolučnej histórii.

5.1 Pomocné triedy a ďalší použitý softvér

Okrem nástroja *EHDraw* sme pre tvorbu testov použili nasledujúce programy a triedy s metódou *main*:

StatisticsGenerator je trieda nástroja *EHDraw*, vytvorená v rámci tejto práce, ktorá vytvára rôzne súbory na základe testov popísaných v tejto kapitole.

HistoryRandomizer je trieda nástroja *EHDraw*, vytvorená v rámci tejto práce, ktorá náhodne permutuje, otáča a reže chromozómy v každej vrstve vstupnej evolučnej histórie. Tým získame histórie s veľkým počtom krížení.

EvolutionGenerator je pomocná trieda programu *PIVO2* [6]. *PIVO2* slúži na rekonštrukciu evolučných histórií a je vylepšenou verziou pôvodného programu *PIVO* [7]. Trieda generuje evolučné histórie genómov s viacerými chromozómami vo formáte špecifickom pre *PIVO* a berie tri vstupné parametre: počet listov (súčasných druhov), počet génov v genóme a maximálny počet DCJ operácií medzi dvoma vrstvami. DCJ (double cut and join) operácie režú genóm na menšie časti, ktoré sa následne spoja dokopy v inom poradí. Generátor sme upravili tak, aby bral aj štvrtý vstupný parameter - počet chromozómov, a zároveň aby generoval histórie, v ktorých sú genómy rozdelené na tento počet náhodne dlhých chromozómov. Chromozómy majú rovnakú pravdepodobnosť byť lineárne ako aj cirkulárne.

EHConverter [2] je program v jazyku Java, ktorý konvertuje evolučné histórie z rôznych formátov do formátu vstupu nástroja *EHDraw*. Tento program vznikol ako ročníkový projekt autora tejto práce.

5.2 Generované dáta

V prvom teste sme na generovaných dátach (evolučných históriách vygenerovaných triedou *EvolutionGenerator*) porovnávali štyri hodnoty:

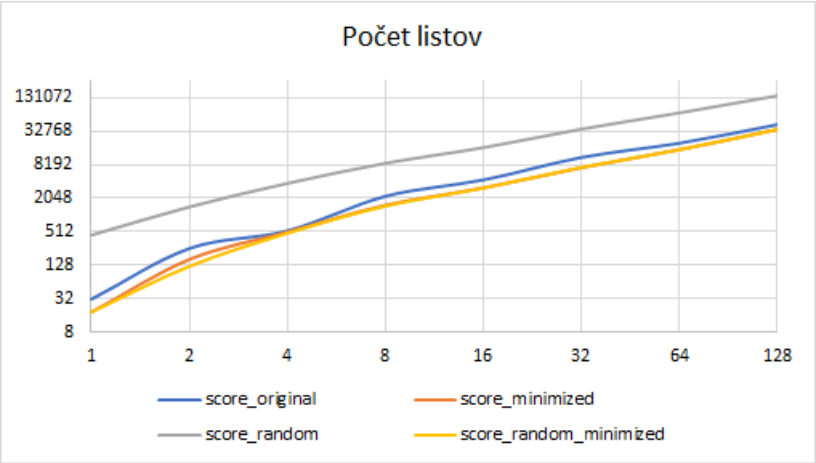
- s_i - skóre vstupnej evolučnej histórie, ktorú vypísal generátor alebo program na rekonštrukciu evolučných histórií,
- s_m - skóre evolučnej histórie po optimalizácii našim algoritmom,
- s_r - skóre evolučnej histórie znáhodnenej triedou *HistoryRandomizer*,
- s_{rm} - skóre po optimalizácii znáhodnenej evolučnej histórie.

Keďže pre evolučné histórie nepoznáme ich optimálne skóre, nemôžme porovnať optimálne skóre a skóre evolučnej histórie po optimalizácií našim algoritmom. Môžeme však porovnať tieto štyri hodnoty na rôznych vstupoch s cieľom zistiť, či náš algoritmus funguje na všekých vstupoch rovnako dobre ako aj na ich znáhodnených históriách. Ďalším cieľom je nájsť závislosť medzi štyrmi hodnotami a rôznymi parametrami evolučných histórií. Čo nás ešte môže zaujímať je koľko krát menšie skóre získame optimalizáciou znáhodného vstupu.

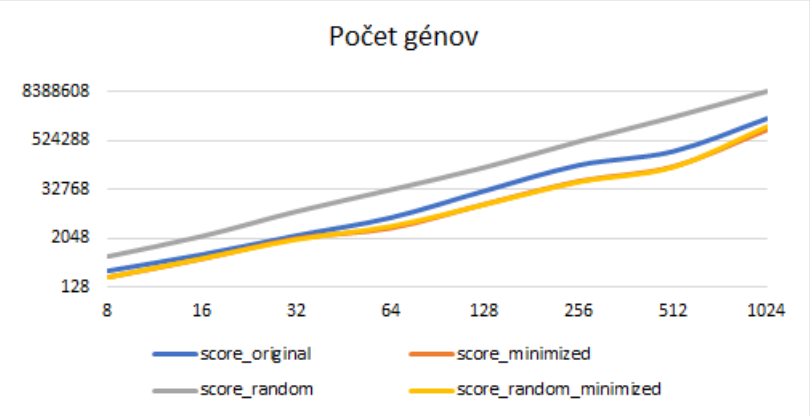
Požadované štyri hodnoty získavame z generovaných dát podľa parametrov, ktoré sme generátoru dodali. Formát vygenerovaných evolučných histórií vieme zmeniť pomocou programu *EHConvertor* na formát vstupu *EHDraw* a následne pomocou *EHDraw* a jeho triedy *HistoryRandomizer* získame požadované hodnoty pre danú evolučnú históriu.

Výsledkom prvého testu na generovaných dátach sú štyri grafy (obr. 5.1 - 5.4) zobrazujúce skúmané hodnoty pre rôzne hodnoty parametrov generátora. Každý z grafov zobrazuje vzťah skúmaných skóre (y-ová súradnica) a zvyšujúcej sa hodnoty jedného z parametrov (x-ová súradnica), kde ostatné parametre sú nastavené na predvolené hodnoty. Pre každú kombináciu parametrov sme vygenerovali jednu evolučnú históriu. Zvýšenie hodnoty niektorého z parametrov o 1 by počet krížení výrazne nezmenilo. Za hodnoty parametrov preto budeme dosádzať len mocniny dvojky.

Ako predvolené hodnoty parametrov sme pre počet listov zvolili hodnotu 8, pre počet génov hodnotu 32, pre počet DCJ operácií hodnotu 8 a pre počet chromozómov hodnotu $\frac{1}{4}$ počtu génov.



Obr. 5.1: Graf závislosti skúmaných skóre a počtu listov evolučnej histórie.



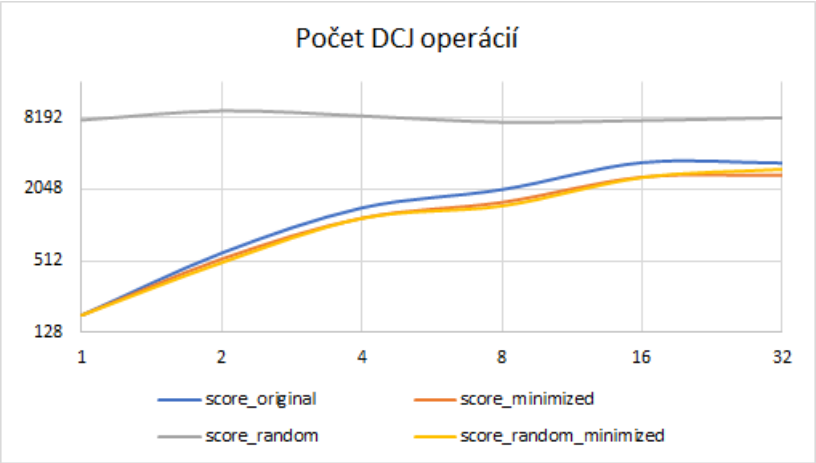
Obr. 5.2: Graf závislosti skúmaných skóre a počtu génov v genómoch evolučnej histórie.

Výsledné čiarové grafy sme vytvorili pomocou nástroja *Microsoft Excel 2016*. Aby bolo možné graf dobre vidieť aj pre menšie hodnoty, y-ovú os zobrazujeme tiež v logaritmickej mierke so základom 2. Názvy grafov zodpovedajú názvu parametra, ktorého hodnoty v danom grafe meníme.

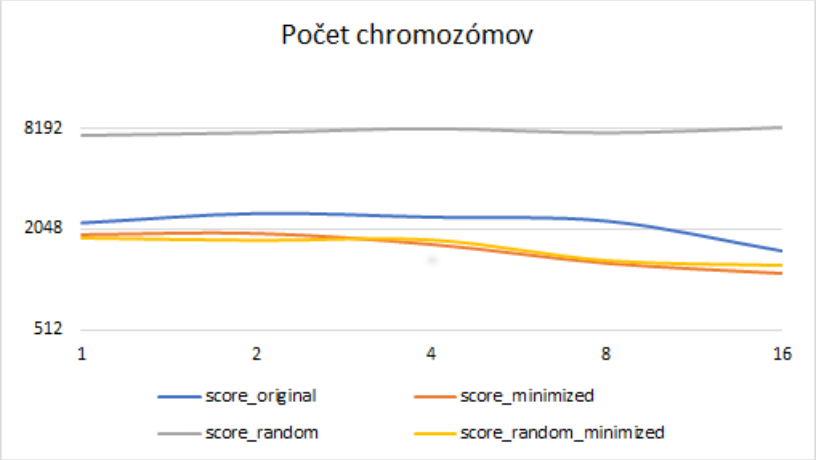
Na výsledných grafoch vidíme, že hodnoty s_m aj s_{rm} sú vždy menšie alebo rovné ako skóre s_i, ktoré je zároveň vždy omnoho menšie ako s_r. Hodnota s_{rm} je vo väčšine prípadov približne štyri krát menšia ako hodnota s_r. Hodnoty s_m a s_{rm} sú podobné, ale z týchto dát o ich vzťahu veľa povedať nevieme.

Prvý graf, v ktorom zvyšujeme počet listov generovanej histórie, sa nachádza na obrázku 5.1. Skúmame histórie s počtom listov s hodnotami $1, 2, 4, \ldots, 128$. Vidíme, že graf sa správa intuitívne, t.j. s rastúcim počtom listov rastú aj všetky hodnoty skúmaných skóre.

Graf, v ktorom meníme počet génov na hodnoty $8, 16, ..., 1024$, je vyobrazený na obrázku 5.2. Tento graf sa správa podobne ako predošlý graf. Rastúci počet génov má teda na skóre podobný vplyv ako rastúci počet listov. To je očakávané, pretože keď



Obr. 5.3: Graf závislosti skúmaných skóre a počtu DCJ operácií medzi vrstvami evolučnej histórie.



Obr. 5.4: Graf závislosti skúmaných skóre a počtu chromozómov vo vrstvách evolučnej histórie.

zdvojnásobíme počet génov v celej evolučnej histórií, počet všetkých čiar bude približne rovnaký ako keď zdvojnásobíme počet listov.

Tretí graf (obr. 5.3), zobrazujúci hodnoty skóre pre $1, 2, 4, ..., 32$ DCJ operácií sa správa od predošlých grafov inak. Skóre znáhodneného grafu je približne stále rovnaké, originálne a minimalizované skóre so zvyšujúcim sa počtom DCJ operácií rastú. Keďže DCJ operácie evolučnú históriu len znáhodňujú, s_r sa nemá prečo moc zvyšovať. Ostatné hodnoty skóre sa zvyšujú, lebo už na vstup prídu evolučné histórie znáhodnené DCJ operáciami a náš algoritmus nevie vracať tieto operácie naspäť. Keď počet DCJ operácií dosiahne hodnotu počtu génov, skóre by sa už ďalej nemali zväčšovať.

V poslednom grafe (obr. 5.4) meníme počet chromozómov, konkrétne od jedného chromozómu po polovicu počtu génov. Podobne ako v predošlom grafe, skóre znáhodneného grafu sa s rastúcim počtom chromozómov príliš nemení. Ostatné skóre sa však s

pribúdajúcim počtom chromozómov zmenšujú. To môže byť spôsobené tým, že krížení, ktorých sa nevieme zbaviť vo väčších chromozómoch otáčaním ani rezaním, sa zbavíme v históriách, kde sú tieto väčšie chromozómy rozdelené na malé, preusporiadaním malých chromozómov.

5.2.1 Závislosť skóre s_m a s_{rm}

Na základe prvého testu sme videli, že s_m je raz menší a raz väčší oproti s_{rm}. V tomto teste budeme skúmať, či neexistuje vzťah medzi pomerom týchto dvoch skóre a rôznymi hodnotami parametrov. Týmto testom vlastne zisťujeme, či je algoritmus citlivý na počiatočné usporiadanie chromozómov v histórii a či pri vstupe, ktorý už je pomerne dobrý (t.j. vstupná história) dostaneme lepšie alebo horšie výsledky ako na znáhodnenom vstupe.

Pre každú z kombinácií parametrov sme vygenerovali šestnásť evolučných histórií a dali do pomeru ich hodnoty s_m a s_{rm}. Aby sme získali všeobecnejšiu hodnotu s_{rm}, vypočítali sme ju tak, že históriu sme štyrikrát znáhodnili, pre každé znáhodnenie sme zistili skóre po minimalizácii a tieto štyri hodnoty sme spriemerovali a zaokrúhlili.

Výsledné grafy sú opäť štyri a pre každú hodnotu meniaceho sa parametra skúmame šestnásť pomerov skóre. Grafy sú narozdiel od predošlých bodové grafy. Y-ová súradnica bodov je dvojkový logaritmus pomerov skóre. Ak je y-ová súradnica bodu väčšia ako 0, znamená to, že $s_m > s_{rm}$, ak je menšia ako 0, tak platí opačná nerovnosť. Predvolené a skúmané hodnoty parametrov zostávajú rovnaké ako v prvom teste.

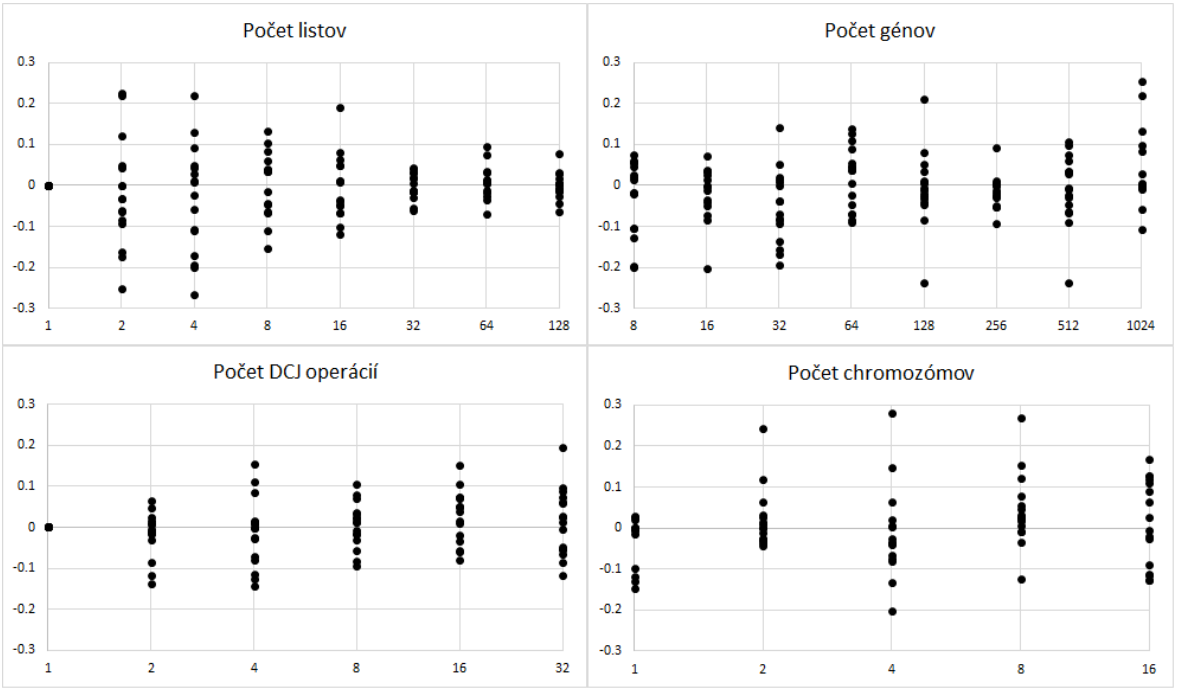
Výsledné grafy zobrazuje obrázok 5.5. Keďže v ani jednom grafe nevidno jasnú závislosť, môžeme prísť k záveru, že pomer minimalizovaných krížení nezáleží od parametrov, na základe ktorých sa vygeneruje evolučná história. Z toho vyplýva, že algoritmus minimalizuje skóre evolučných histórií s veľkým počiatočným skóre približne rovnako dobre ako evolučné histórie s menším počiatočným skóre, pričom nezáleží na parametroch týchto evolučných histórií.

5.3 Reálne dáta

Vďaka programu *EHConverter* môžeme pomocou *EHDraw* zobraziť a skúmať evolučné histórie, ktoré sme získali na základe reálnych dát vo formáte špecifickom pre *PIVO*. V tejto časti opíšeme skúmané evolučné histórie a bližšie sa pozrieme na ich hodnoty s_i, s_m, s_r a s_{rm}.

Evolučná história kvasiniek(z rodu *Candida*)

Dáta, podľa ktorých sa rekonštruovala táto evolučná história, pochádzajú z mitochondriálnych génov súčasných druhov kvasiniek, ktoré sa analyzovali vo vedeckom



Obr. 5.5: Grafy zobrazujúce pomery minimalizovaných skóre pre rôzne parametre generovania evolučných histórií.

článku od Valacha a spol. [15]. História bola rekonštruovaná pomocou pôvodného programu *PIVO*.

Je to história šestnástich druhov kvasiniek, kde genóm jedného z týchto druhov je tvorený dvoma chromozómami. Genómy ostatných druhov a všetkých spoločných predkov obsahujú len jeden chromozóm. Cirkulárne a lineárne chromozómy sa nachádzajú v histórií v približne rovnakom pomere. Genómy obsahujú priemerne dvadsaťpäť génov a približne polovica z nich sú gény s opačnou orientáciou.

Evolučné histórie cicavcov

Dáta, podľa ktorých sa rekonštruovali tieto evolučné histórie, predstavujú vybrané gény z istého chromozómu človeka, ktoré sa nachádzajú aj v genóme iných druhov cicavcov. Sú to teda histórie ľudských génov u viacerých cicavčích druhov. Evolučné histórie boli rekonštruované programom *PIVO2*. Poradia génov v dnešných druhoch pochádzajú z dát použitých v článku od Foote et al. [5].

Prvá história (mammal1) skúma tridsaťjeden génov z prvého chromozómu človeka a obsahuje sedem súčasných druhoch cicavcov v tomto poradí: človek, myš, lama, kosatka, krava, pes, mrož. Genómy niektorých druhov majú vybrané ľudské gény viac pokope ako iné, preto sa počet chromozómov v histórii u rôznych druhov značne líši. Navyše u niektorých druhov ešte nepoznáme celé chromozómy, ale iba časti nazvané kontigy, ktoré ale v obrázku tiež zobrazujeme ako chromozómy. Skutočný biologický počet chromozómov však môže byť nižší. Priemerný počet chromozómov resp. konti-

Tabuľka 5.1: Tabuľka skúmaných hodnôt skóre na reálnych vstupoch.

	s_i	s_m	s_r	s_{rm}
candida	2690	2329	9386	2321
mammal1	486	471	6531	659
mammal2	384	384	4997	455
mammal3	342	342	3278	342

gov vo vrstvách je približne 10 a v histórií sa nachádzajú len lineárne chromozómy. História neobsahuje takmer žiadne kríženia génov a približne polovica génov je opačnej orientácie.

Druhá história (mammal2) je história tridsiatich génov druhého chromozómu človeka v genómoch tých istých druhov ako v predošlej histórii. Od predošlej sa líši tým, že gény sú viac pokope a teda priemerný počet chromozómov v jednej vrstve je menší (približne 4). Ostatné parametre sú takmer rovnaké ako v prvej histórii.

Tretia história (mammal3) je história tridsiatich génov druhého chromozómu človeka v genómoch piatich druhov v tomto poradí: človek, myš, kosatka, krava, pes. Od predošlej sa líši tým, že sme brali iné gény z druhého ľudského chromozómu a tým, že sme odstránili druhy v ktorých boli gény rozdelené do najviac kontigov, konkrétne druhy lama a mrož. Priemerný počet chromozómov vo vrstvách je približne 3. Ostatné parametre sú takmer rovnaké ako v dvoch predošlých históriách.
Každú z histórií reálnych dát sme tridsaťdva krát znáhodnili a hodnoty s_r a s_{rm} sú priemery skóre týchto tridsaťdva evolučných histórií. Hodnoty skúmaných skóre pre evolučné histórie na základe reálnych dát vidíme v tabuľke 5.1. V kvasinkovej histórií (*candida*), ktorá je prakticky tvorená len vrstvami s jedným chromozómom, prebieha len k rezaniu a otáčaniu chromozómov. Vidíme, že aj túto históriu vieme oproti vstupnej vylepšiť rovnako dobre ako oproti jej znáhodnenej histórii. Evolučné histórie cicavcov sú tvorené viacerými, len lineárnymi chromozómami. Vidíme, že *mammal1*, ktorá obsahuje viac chromozómov ako ďalšie dve histórie cicavcov, má oproti nim všetky skóre väčšie. Tiež vidíme, že v históriách *mammal2* a *mammal3* sme skóre optimalizáciou vstupnej histórie nevylepšili. Kvôli tomu, že *mammal2* obsahuje viac chromozómov ako *mammal3*, znáhodnené skóre majú väčšie hodnoty. Zaujímavé je, že v *mammal3* vieme znáhodnené histórie opraviť rovnako dobre ako pôvodné, čo sa nám v *mammal1* ani *mammal2* ani zďaleka nepodarilo.

Poznámka: Obrázky vstupných histórií aj histórií po znáhodnení a minimalizácii sa nachádzajú v elektronickej prílohe.

Záver

Výsledkom tejto bakalárskej práce je rozšírený nástroj *EHDraw*, ktorý môže oproti pôvodnému nástroju zobrazovať evolučné histórie druhov s viacerými chromozómami. V literatúre sme našli heuristiky, ktoré minimalizujú počet križení na riadnom vrstvovom grafe. Definovali sme graf evolučnej histórie, ktorý spĺňa mnohé vlastnosti riadneho vrstvového grafu a preformulovali sme problém minimálneho počtu križení na optimalizačný problém preusporiadania chromozómov. Ďalej sme na grafe evolučnej histórie definovali optimalizačný problém otáčania chromozómov a rezania cirkulárnych chromozómov. Riešenie týchto troch problémov sme spojili do jedného optimalizačného algoritmu a tento algoritmus sme aj implementovali. Algoritmus sa môže aplikovať na obrázok evolučnej histórie klikom jedného tlačítka v grafickej aplikácií rozšíreného nástroja alebo pomocou nastavenia argumentov príkazového riadku. Obrázok obsahuje po optimalizácií značne menej križení čiar, čiarkovaných čiar a zmien orientácií génov, pričom zobrazuje tú istú evolučnú históriu. V rámci poslednej kapitoly sme algoritmus testovali porovnávaním rôznych hodnôt skóre na vygenerovaných evolučných históriách a na históriách vychádzajúcich z reálnych dát. Výsledkom testovania boli najmä grafy, ktoré zobrazujú správanie spomínaných hodnôt skóre na evolučných históriách vygenerovaných podľa rozličných parametrov. Hlavným zistením bolo, že algoritmus minimalizuje skóre evolučných histórií s veľkým počiatočným skóre rovnako dobre ako evolučné histórie s menším počiatočným skóre, pričom nezáleží na parametroch týchto evolučných histórií, aj keď na dvoch sadách reálnych dát sa tento trend nepotvrdil.

Je viacero možností, čo by sa dalo v rámci optimalizácie evolučných histórií ešte ďalej skúmať. Jednou z možností je pomocou celočíselného lineárneho programovania zistiť optimálne skóre pre evolučnú históriu a porovnávať toto optimum a skóre po minimalizácií našim algoritmom. Taktiež sa dá premyslieť, či neexistujú spôsoby, ako by sa nami implementovaný algoritmus dal vylepšiť. Pokračovaním by mohla byť implementácia ďalších heuristík jednostranného problému z literatúry a porovnávať výsledné minimalizované hodnoty skóre pre rôzne heuristiky. Bolo by možné aj vymyslieť a implementovať heuristiky, ktoré riešia viac-vrstvový problém pre náš špecifický graf a opäť porovnávať minimalizované skóre po zbehnutí rôznych heuristík na tých istých vstupoch.

Pokračovať by sa dalo aj inými smermi. Jedným z možných smerov je ďalej rozšíriť

nástroj *EHDraw*, aby obsahoval rôzne dodatočné informácie o evolučných históriách, respektíve aby používateľ vedel v rámci nástroja skúmať evolučné histórie interaktívnejším spôsobom, ako len na základe obrázku. Tým sa otvára otázka, akými zásadne inými spôsobmi môžeme evolučné histórie zobrazovať.

Ak vás niektorý z týchto možných smerov zaujal, zdrojové súbory nami rozšíreného nástroja *EHDraw* sa nachádzajú okrem elektronickej prílohy aj na stránke `https://github.com/rchladek/bak`.

Literatúra

[1] FigTree. `http://tree.bio.ed.ac.uk/software/figtree/`. Accessed May 8, 2017.

[2] Radoslav Chládek. Skripty na konverziu formátov evolučných histórií. `http://davinci.fmph.uniba.sk/~chladek8/rocnikovy_chladek`. Accessed May 15, 2017.

[3] Zmasek C.M. Archaeopteryx: Visualization, analysis, and editing of phylogenetic trees. `https://sites.google.com/site/cmzmasek/home/software/`. Accessed May 11, 2017.

[4] Peter Eades and Nicholas C Wormald. Edge crossings in drawings of bipartite graphs. *Algorithmica*, 11(4):379–403, 1994.

[5] Andrew D Foote, Yue Liu, Gregg WC Thomas, Tomáš Vinař, et al. Convergent evolution of the genomes of marine mammals. *Nature genetics*, 47(3):272–275, 2015.

[6] Albert Herencsár. An improved algorithm for ancestral gene order reconstruction. Master's thesis, Comenius University in Bratislava, 2014. Supervised by Broňa Brejová.

[7] Jakub Kovac, Brona Brejova, and Tomas Vinar. A Practical Algorithm for Ancestral Rearrangement Reconstruction. In Teresa M. Przytycka and Marie-France Sagot, editors, *Algorithms in Bioinformatics, 11th International Workshop (WABI)*, volume 6833 of *Lecture Notes in Computer Science*, pages 163–174, Saarbrücken, Germany, September 2011. Springer.

[8] Christian Matuszewski, Robby Schönfeld, and Paul Molitor. Using sifting for k-layer straightline crossing minimization. In *Graph drawing*, pages 217–224. Springer, 1999.

[9] Hiroshi Nagamochi and Nobuyasu Yamada. Counting edge crossings in a 2-layered drawing. *Inf. Process. Lett.*, 91(5):221–225.

[10] Oscar Robinson, David Dylus, and Christophe Dessimoz. Phylo. io: interactive viewing and comparison of large phylogenetic trees on the web. *Molecular biology and evolution*, page msw080, 2016.

[11] Georg Sander. *Graph layout through the VCG tool*, pages 194–205. Springer Berlin Heidelberg, 1995.

[12] Dávid Simeunovič. Vizualizácia evolučných histórií. Bakalárska práca, Univerzita Komenského v Bratislave, 2016.

[13] Kozo Sugiyama, Shojiro Tagawa, and Mitsuhiko Toda. Methods for visual understanding of hierarchical system structures. *IEEE Transactions on Systems, Man, and Cybernetics*, 11(2):109–125, 1981.

[14] Roberto Tamassia. *Handbook of graph drawing and visualization*, chapter 13. CRC press, 2013.

[15] Matus Valach, Zoltan Farkas, Dominika Fricova, Jakub Kovac, Brona Brejova, Tomas Vinar, Ilona Pfeiffer, Judit Kucsera, Lubomir Tomaska, B Franz Lang, et al. Evolution of linear chromosomes and multipartite genomes in yeast mitochondria. *Nucleic acids research*, page gkq1345, 2011.

Elektronická príloha

Súčasťou práce je elektronická príloha obsahujúca zdrojové súbory rozšíreného nástroja. Táto príloha obsahuje aj *.jar* súbory programov *PIVO2* a *EHConverter*. Obrázky vo formáte *SVG* zobrazujúce reálne a dve generované evolučné histórie vizualizované našim nástrojom sú v priečinku *obrázky*.