Univerzita Komenského v Bratislave
Fakulta matematiky, fyziky a informatiky

Vizualizácia chromozomálnych zmien počas evolúcie

Bakalárska práca

2018
Jaroslava Kokavcová

Univerzita Komenského v Bratislave
Fakulta matematiky, fyziky a informatiky

Vizualizácia chromozomálnych zmien počas evolúcie

Bakalárska práca

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

Bratislava, 2018
Jaroslava Kokavcová

Univerzita Komenského v Bratislave
Fakulta matematiky, fyziky a informatiky

ZADANIE ZÁVEREČNEJ PRÁCE

Meno a priezvisko študenta: Jaroslava Kokavcová
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 chromozomálnych zmien počas evolúcie
Visualization of chromosomal changes during evolution

Anotácia: Cieľom práce je vyvinúť algoritmy na vizualizáciu presunov génov medzi jednotlivými chromozómami počas evolúcie. Práca bude nadväzovať na existujúci vizualizačný nástroj EHdraw, v ktorom sa jednotlivé gény zobrazujú ako čiary rôznych farieb. Hlavným cieľom práce je navrhnúť formulácie problému a optimalizačné algoritmy, ktoré vo výslednej vizualizácii zabránia umelému kríženiu čiar, ktoré nezodpovedá žiadnej evolučnej zmene.

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

Dátum zadania: 19.10.2018

Dátum schválenia: 24.10.2018 doc. RNDr. Daniel Olejár, PhD.
garant študijného programu

.....
študent

.....
vedúci práce

Poďakovanie:

Abstrakt

V práci sa zaoberáme vizualizáciou presunov génov medzi chromozómami počas evolúcie. Práca nadväzuje na existujúci program EHDraw. Hlavným výsledkom práce je formulácia problému poradia chromozómov tak, aby vzniklo čo najmenej umelých kríženi, ktoré nezodpovedajú evolučným zmenám genómu. K tomu je potrebné zadefinovať, ktoré kríženia sú dobré a ktoré sú zlé. Zároveň sme sa venovali tomu, ako je možné tento problém riešiť. Problém transformujeme na známy problém, na ktorý poznáme algoritmus a otestujeme, ako dobre tento algoritmus rieši náš problém.

Kľúčové slová: evolučná história, translokácia, vizualizácia, minimalizácia krížení

Abstract

In thesis we consider visualization of gene transfer between chromosomes during evolution. Thesis follows up existing tool EHDRaw. The main result of thesis is formulation of chromosome order problem so that there is as few spurious crossings which do not correspont to genome change as possible. To accomplish it it is necessary to define good and bad crossings. In thesis we also are looking for solution of our problem. We transform our problem to known problem and we use algorithm for known problem to solve our problem. Then we test whether the algoritmus is good enough for our problem.

Keywords: evolution history, translocation, visualization, crossing minimization

Obsah

Zoznam obrázkov

Zoznam tabuliek

Úvod

Od nepamäti človek skúmal, ako vznikol svet okolo neho. Snažil sa vysvetliť vznik rôznych organizmov pomocou mýtov. Postupne začali vznikať názory, že existujúce druhy sa môžu medzi sebou krížiť a tak vznikajú nové druhy. Veľký prelom nastal v 19. storočí, keď Charles Darwin publikoval svoju evolučnú teóriu, ktorá tvrdí, že všetky druhy vznikli z malého počtu druhov postupnými zmenami genetickej informácie. Ďalší prelom nastal v sedemdesiatich rokoch minulého storočia, keď sa prvýkrát podarilo sekvenovať DNA, ktorá je nositeľkou genetickej informácie. Postupným zlepšovaním technológií sekvenovanie DNA sa stalo rýchlejšie a lacnejšie, a vďaka tomu dnes už poznáme genetickú informáciu veľkého množstva druhov.

Na základe genetickej informácie súčasných druhov ako aj genetickej informácie niektorých vyhynutých druhov môžeme zisťovať, aké zmeny nastali v organizmoch počas evolúcie. Postupnosť týchto zmien nazývame evolučná história. Vďaka spolupráci vedcov z rôznych oblastí, ako napríklad biológia, matematika, informatika, či fyzika, môžeme rekonštruovať evolučné histórie. Zároveň s tým vzniká potreba tieto evolučné histórie vhodne vizualizovať.

Jeden z nástrojov na vizualizáciu evolučných histórií vznikol aj na našej fakulte. Program EHDraw[17] najskôr zobrazoval iba genómy s jedným chromozómom, neskôr bol rozšírený [3], aby umožňoval zobrazovať aj genómy s viacerými chromozómami. Keďže genóm je množina chromozómov, môžeme ich zobrazovať v ľubovoľnom poradí. Preto sa vynorila otázka, aké poradie zvoliť, aby výsledná vizualizácia bola čo najprehľadnejšia.

Cieľom tejto práce je zlepšiť vizualizáciu genómu nástrojom EHDraw, ktorý v súčasnosti minimalizuje celkový počet krížení čiar. No toto kritérium nemusí byť postačujúce, pretože nezohľadňuje, kde sa kríženia nachádzajú. V práci sme sa zamerali na to, aby sme sa zbavili krížení, ktoré nezodpovedajú evolučným zmenám a zostali kríženia iba tam, kde sú vynútené presunmi génov medzi chromozómami alebo inými evolučnými zmenami. Tento problém sme sa najskôr formálne zadefinovali a potom sme hľadali spôsoby riešenia.

Práca pozostáva z 5 kapitol. V prvej kapitole si predstavíme niektoré dôležité pojmy z biológie a bioinformatiky, ktoré súvisia s vizualizáciou evolučných histórií. V druhej kapitole sa pozrieme na aktuálny stav nástroja EHDraw. V tretej kapitole formálne

zadefinujeme problém a pozrieme sa na to, ako ho riešiť pomocou známych algoritmov. V štvrtej kapitole si predstavíme konkrétny algoritmus na riešenie daného problému, ktorý aj implementujeme. Posledná kapitola bude obsahovať experimentálne vyhodnotenie.

Kapitola 1

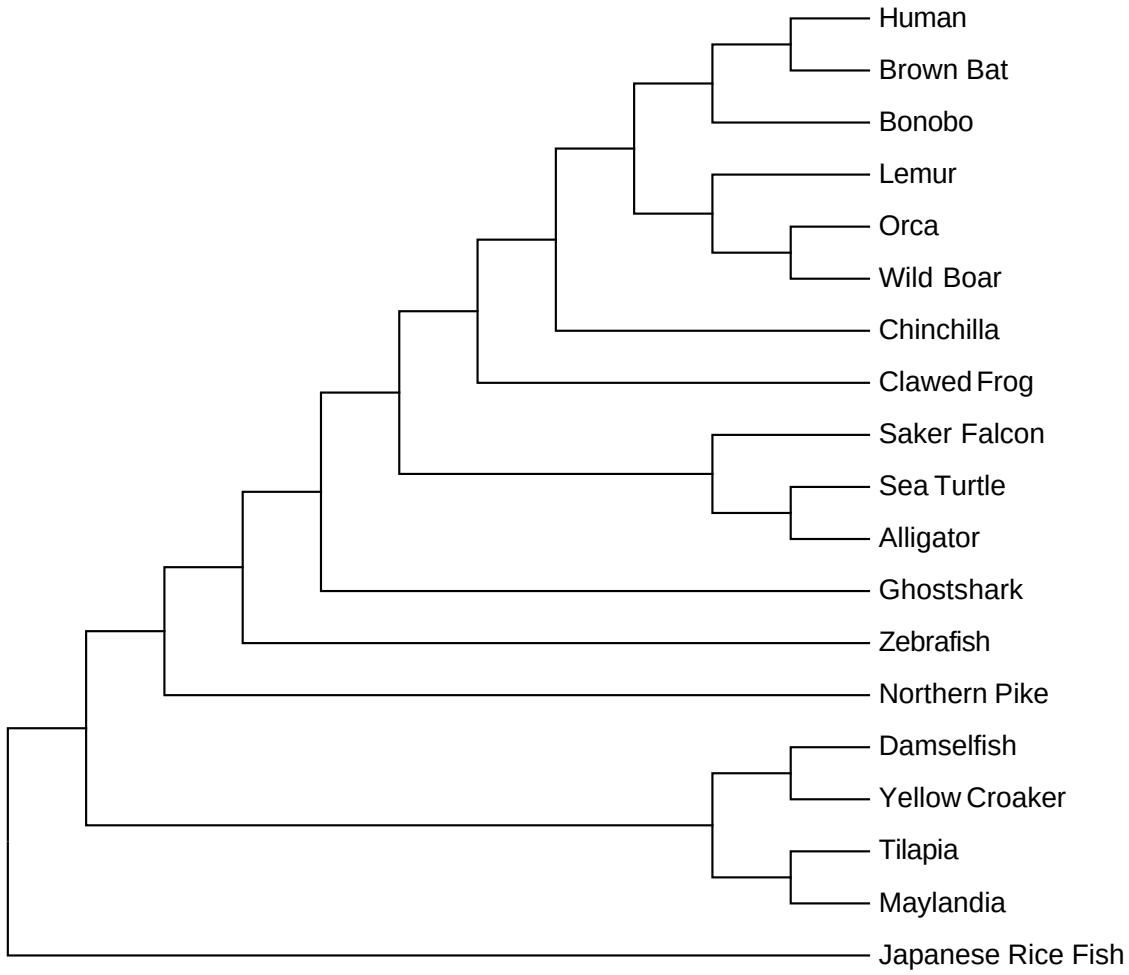
Základné pojmy

V tejto kapitole si predstavíme niektoré základné pojmy z biológie a bioinformatiky, ktoré sa v práci budú vyskytovať. Okrem toho sa pozrieme na súčasný stav problematiky.

1.1 DNA, gén, chromozóm

DNA *(deoxyribonucleic acid)* je prírodná kyselina, ktorá bola objavená v jadre bunky, z čoho pochádza aj jej názov. Hlavnou úlohou DNA je uchovávanie genetickej informácie, ktorá sa prenáša z generácie na generáciu. Riadi rast, delenie a regeneráciu bunky a tým predurčuje vývoj a vlastnosti celého organizmu. Skladá sa z dvoch komplementárnych vlákien, ktoré tvoria dvojzávitnicu. Každé vlákno je dlhý reťazec nukleotidov (niekoľko miliónov), ktoré pozostávujú zo sacharidu deoxyribózy, fosfátového zvyšku a jednej zo štyroch dusíkatých báz – adenín (A), cytozín (C), guanín (G) a tymín (T). Nukleotid je teda určený jeho bázou a lineárne zapísané písmená (skratky báz) tvoria primárnu štruktúru DNA. Podľa toho, či vlákno je ukončené fosfátom alebo hydroxylovou skupinou rozlišujeme 5' a 3' koniec. Poradie nukleotidov vždy zapisujeme od 5' konca k 3' koncu. Medzi bázami komplementárnych vlákien vznikajú vodíkové väzby, konkrétne A sa páruje s T a C sa páruje s G. Na základe toho, keď poznáme jedno vlákno, vieme k nemu dorobiť komplementárne, a to tak, že každú bázu nahradíme komplementárnou bázou a potom vymeníme poradie napr. pre sekvenciu ACCCGTA dostaneme TACGGGT.

Genetická informácia môže byť uložená vo viacerých molekulách DNA, napr. človek ich má 46. Každú molekulu nazývame **chromozóm**. Poznáme dva typy chromozómov – lineárny a cirkulárny. Na koncoch lineárnych chromozómov sa nachádzajú špeciálne časti nazývané teloméry. Cirkulárne chromozómy môžeme nájsť napríklad u baktérií. Líšia sa tým, že majú oba konce spojené do kruhu, teda nemajú voľné konce. Po "prerezaní"cirkulárneho chromozómu dostávame lineárny chromozóm bez telomér.



Obr. 1.1: Príklad zakoreneného fylogenetického stromu. Na tomto obrázku je znázornený vzťah jedného ľudského génu a jeho ortológov (gény, ktoré majú spoločného predka a oddelili sa speciáciou). Zdroj [16]

Gén je úsek DNA, ktorý kóduje informáciu pre tvorbu produktu, najčastejšie proteínu. V tejto práci budeme zobrazovať poradie génov na chromozóme, ale vo všeobecnosti môžeme našimi metódami namiesto génov zobrazovať aj ľubovoľné úseky DNA.

Genóm je súbor chromozómov. Je to vlastne celá genetická informácia organizmu uložená v DNA. V našej práci budeme genóm chápať ako množinu chromozómov a chromozóm bude určený poradím génov. Gén bude reprezentovaný svojím ID a znamienkom. ID génu je celé číslo, určujúce istý gén alebo úsek DNA. Jeden chromozóm môže obsahovať viac génov s rovnakým ID, ide o kópie s podobnou sekvenciou. Ak sa gén nachádza na opačnom vlákne DNA, budeme to značiť záporným znamienkom.

1.2 Fylogenetický strom

Fylogenetický strom je grafické zobrazenie evolučných vzťahov medzi rôznymi druhmi. Príklad fylogenetického stromu môžeme vidieť na obrázku 1.1. Listy znázorňujú súčasné organizmy. Vnútorné vrcholy zodpovedajú ich vyhynutým predkom. Fylogenetický strom zobrazuje dobre všetky speciácie, teda udalosti, pri ktorých zo spoločného predka vzniknú dva nové druhy. Týmto udalostiam zodpovedajú vnútorné vrcholy. Nevýhodou fylogenetických stromov je, že nezobrazujú genómy jednotlivých druhov a zmeny, ktoré v nich nastali. Niektoré stromy navyše zachytávajú aj horizontálny prenos genetickej informácie, teda presun génov z organizmu, ktorý nie je predkom prijímajúceho organizmu. Jednotlivým hranám sa zvyknú priraďovať dĺžky, ktoré zodpovedajú času medzi dvoma speciáciami.

Na vizualizáciu fylogenetických stromov existuje množstvo programov ako napríklad *phylo.io* [14] alebo *ETE toolkit* [8].

1.3 Evolučná história

Evolučná história je postupnosť udalostí, ktoré sa odohrali na nejakej DNA sekvencii. Môže to byť napríklad mutácia jednej bázy na inú, pridanie alebo vymazanie niektorej bázy. Takéto zmeny môžu ovplyvniť funkciu génu, no v tejto práci nás budú zaujímať udalosti, ktoré menia poradie alebo počet génov na chromozómoch. Preto budeme uvažovať hlavne tieto udalosti:

- Duplikácia — skopírovanie súvislého úseku DNA (jedného alebo viacerých génov) na iné miesto v genóme, napr. keď sa v chromozóme s génmi 1 2 3 4 5 zduplikuje úsek 2 3, ktorý sa umiestni na začiatok, dostaneme 2 3 1 2 3 4 5.
- Inzercia — vloženie nového úseku DNA (nových génov) na niektoré miesto v genóme, napr. keď do chromozómu s génmi 1 2 3 4 5 vložíme gén s ID 6 na štvrtú pozíciu, dostaneme 1 2 3 6 4 5.
- Delécia — odstránenie určitého úseku DNA (génov) z genómu, napr. keď v chromozóme s génmi 1 2 3 4 5 sa vymaže úsek 2 3 4, dostaneme 1 5.
- Inverzia — zmena orientácie nejakého súvislého úseku DNA na opačnú. Pri tomto procese sa otočí poradie génov a zmení sa ich orientácia na opačnú, napr. keď v chromozóme s génmi 1 2 3 4 5 nastane inverzia na úseku 1 2 3, dostaneme -3 -2 -1 4 5.
- Speciácia — evolučný proces, pri ktorom vznikne nový druh. Pri tejto udalosti sa evolučná história rozvetvuje a ďalej bude obsahovať informácie aj o zmenách genómu druhu, ktorý zrovna vznikol.

- fúzia — spojenie dvoch chromozómov do jedného, napr. keď chromozóm s génmi 1 2 3 sa spojí s chromozómom s génmi 4 5 6 7, vznikne nový chromozóm s génmi 1 2 3 4 5 6 7.

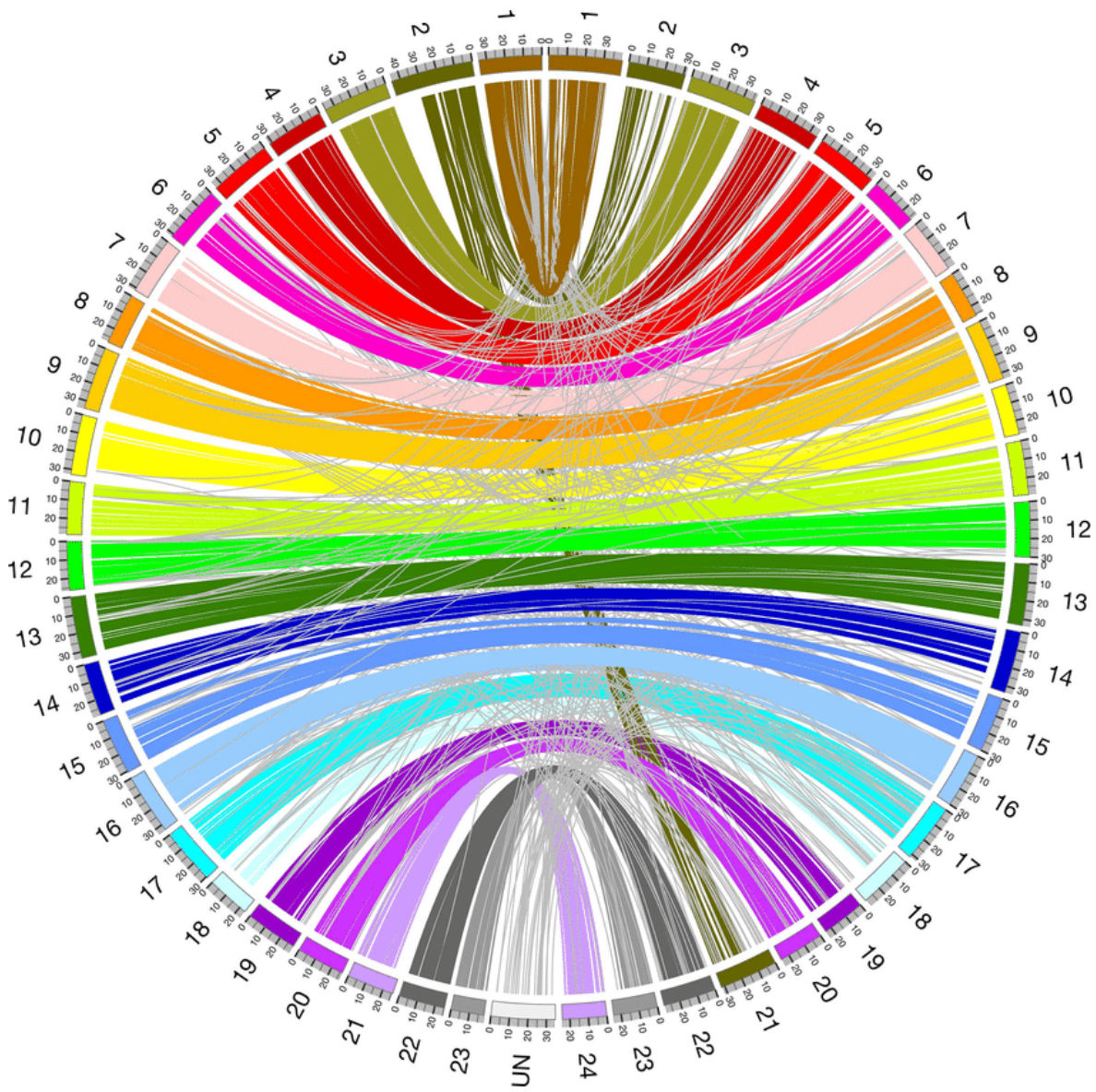
- štiepenie — rozdelenie jedného chromozómu na dva menšie chromozómy, keď chromozóm s génmi 1 2 3 4 5 6 7 rozdelí na chromozómy s génmi 1 2 3 a 4 5 6 7.

- translokácia — presunutie úseku DNA (génov) na nové miesto. Môže sa presunúť v rámci jedného chromozómu, alebo aj medzi rôznymi chromozómami, napr. v chromozóme s génmi 1 2 3 4 5 sa presunie gén 2 medzi gény 4 a 5, teda dostaneme 1 3 4 2 5. Následne sa gény 3 a 4 presunú do stredu chromozómu s génmi 6 7 8 9, teda prvý chromozóm bude mať gény 1 2 5 a druhý 6 7 3 4 8 9.

Vizualizácia evolučných histórií je grafické zobrazenie genómov organizmov a zmien v genómoch v každom čase evolučnej histórie. V našom prípade budeme zobrazovať jednotlivé gény daného genómu a všetky udalosti, ktoré menia poradie alebo počet génov. Gény môžu byť rozdelené do chromozómov. Zobrazujeme fylogenetický strom, ktorý navyše obsahuje informáciu o genóme jednotlivých druhov a zmenách, ktoré v nich nastali.

My sa budeme zaoberať programom *EHDraw* [3], ktorý zobrazuje aj časovú os niektorých zmien v genóme počas evolúcie ako vznik a zánik génov, či presúvanie génov medzi chromozómami. Bližšie tento program popíšeme v nasledujúcej kapitole.

Jeden zo softvérov na vizualizáciu genómov a poradia génov v nich je *Circos* [12]. Na kruhovej osi sa nachádzajú genómy jednotlivých druhov. Čiarami sú spojené rovnaké gény. Príklad výstupu programu Circos môžeme vidieť na obrázku 1.2. Takýto spôsob vizualizácie síce dobre znázorňuje presúvanie génov a iné zmeny v genóme, ale nevidíme v ňom, kedy dané zmeny nastali.



Obr. 1.2: Ukážka výstupu programu *Circos*. Na obrázku vidíme zarovnanie genómu gupky (vľavo) a medaky (vpravo) . Zdroj [2]

Kapitola 2

Nástroj EHDraw

Program na vizualizáciu evolučných histórií, s ktorým budeme v tejto práci pracovať, sa nazýva *EHDraw* (*Evolution History Draw*). Tento program vznikol ako bakalárska práca Dávida Simeunoviča [17] v roku 2016 a bol rozšírený v rámci bakalárskej práce Radoslava Chládeka [3] v roku 2017 obe na Katedre informatiky Fakulty matematiky, fyziky a informatiky Univerzity Komenského v Bratislave. V tejto kapitole sa pozrieme na aktuálny stav programu.

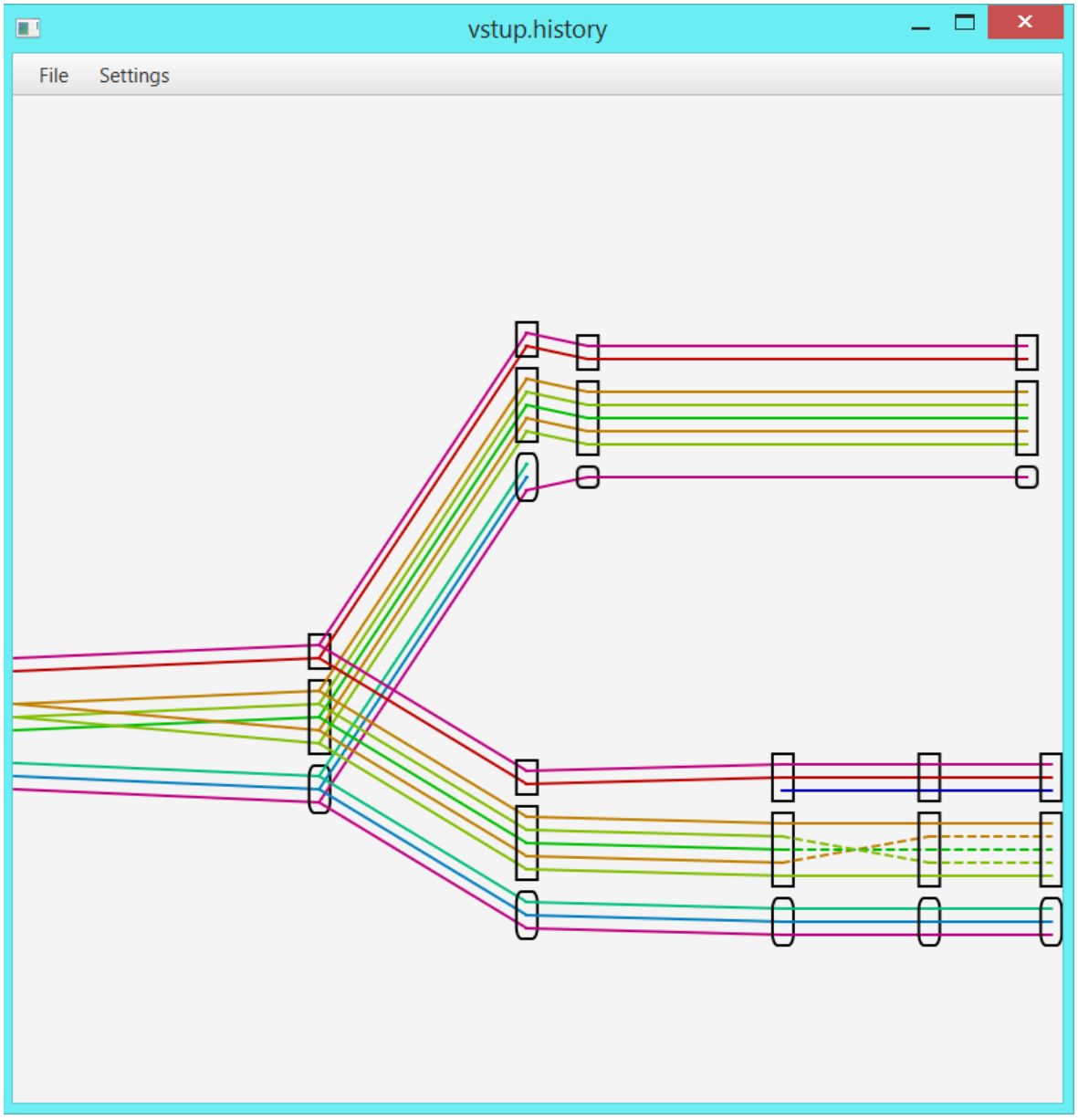
2.1 Výstup

Príklad výstupu programu *EHDraw* je znázornený na obrázku 2.1. Na x-ovej osi sa nachádza čas a na y-môžeme vidieť gény v poradí, v akom sa nachádzajú v genóme. Gény sú znázornené čiarami, pričom rovnaké gény majú rovnakú farbu. Čiarkovaná čiara znamená, že daný gén sa nachádza na opačnom vlákne. Chromozómy v genóme sú oddelené väčšími medzerami. Navyše pri každej udalosti môžeme rozlíšiť, o aký druh chromozómu ide – okolo lineárnych chromozómov je obdĺžnik a okolo cirkulárnych sa nachádza obdĺžnik so zaoblenými rohmi. Genómy rôznych druhov sú oddelené veľkou medzerou.

Na obrázku 2.1 vidíme evolučnú históriu dvoch druhov. Najprv sa zduplikovali gény 3 a 4 a vložili sa medzi piaty a šiesty gén. Následne nastala speciácia – z pôvodného genómu vznikli dve kópie, pre každý druh jedna. V prvom druhu došlo k vymazaniu 8. a 9. génu. V genóme druhého druhu pribudol nový gén na pozíciu 3 a následne sa piaty až siedmy gén pretočili na opačné vlákno.

2.2 Použitie nástroja EHDraw

Vstupom pre nástroj *EHDraw* je evolučná história zapísaná v textovom súbore. Príklad vstupu môžeme vidieť na obrázku 2.2. Podrobný popis formátu vstupného súboru



Obr. 2.1: Ukážka výstupu nástroja EHDraw. Na x-ovej osi je čas, na y-ovej osi sú znázornené gény.

```
predok e1 root 0 root 1 2 $ 3 4 5 $ 6 7 1 @ # -1 -1 -1 -1 -1 -1 -1 -1
predok e2 e1 0.25 dup 1 2 $ 3 4 5 3 4 $ 6 7 1 @ # 0 1 2 3 4 2 3 5 6 7
druh1 e3 e2 0.42 sp  1 2 $ 3 4 5 3 4 $ 6 7 1 @ # 0 1 2 3 4 5 6 7 8 9
druh1 e4 e3 0.47 del 1 2 $ 3 4 5 3 4 $ 1 @ # 0 1 2 3 4 5 6 9
druh1 e5 e4 0.83 leaf 1 2 $ 3 4 5 3 4 $ 1 @ # 0 1 2 3 4 5 6 7
druh2 e6 e2 0.42 sp  1 2 $ 3 4 5 3 4 $ 6 7 1 @ # 0 1 2 3 4 5 6 7 8 9
druh2 e7 e6 0.63 ins  1 2 8 $ 3 4 5 3 4 $ 6 7 1 @ # 0 1 -1 2 3 4 5 6 7 8 9
druh2 e8 e7 0.75 inv 1 2 8 $ 3 -3 -5 -4 4 $ 6 7 1 @ # 0 1 2 3 6 5 4 7 8 9 10
druh2 e9 e8 0.85 leaf 1 2 8 $ 3 -3 -5 -4 4 $ 6 7 1 @ # 0 1 2 3 4 5 6 7 8 9 10
```

Obr. 2.2: Ukážka vstupu pre EHDraw zodpovedajúci výstupu na obrázku 2.1.

môžeme nájsť v bakalárskej práci Radoslava Chládeka [3].

Po spustení programu vidíme prázdnu plochu. Všetky ovládacie prvky nájdeme v hornom menu. Evolučnú históriu načítame vybratím príslušného súboru (*File* $\longrightarrow$ *Open*). Po zobrazení môžeme výstup upravovať pomocou nástrojov v nastaveniach. Môžeme meniť šírku a dĺžku grafickej plochy, hrúbku čiar a veľkosť medzier medzi nimi, veľkosti medzier medzi chromozómami a jednotlivými vetvami (*Settings* $\longrightarrow$ *General Settings*). Taktiež je možné upravovať každý gén zvlášť výberom konkrétneho génu v nastaveniach (*Settings* $\longrightarrow$ *Gene Settings*) alebo kliknutím naň pravým tlačidlom myši. Vieme nastaviť hrúbku čiary génu, jeho farbu alebo vypnúť zobrazovanie daného génu. Okrem toho, výstup je možné priblížiť, vzdialiť a posúvať. V prípade potreby môžeme výstup exportovať do SVG formátu (*File* $\longrightarrow$ *Save Image*) a ďalej upravovať v editore vektorovej grafiky. Pridané funkcie, ako napríklad možnosť približovať a vzďaľovať výstup, oproti verzií od Radoslava Chládeka [3] vznikli vrámci ročníkového projektu autorky tejto práce [11].

2.3 Optimalizácia krížení

Okrem vyššie spomenutých funkcionalít, program EHDraw umožňuje zvýšiť čitateľnosť výstupu minimalizovaním počtu krížiacich sa čiar a taktiež počtu čiarkovaných čiar. Využíva na to heuristiky používané na minimalizáciu krížení v grafe. Pritom rieši problém, ako zoradiť chromozómy, aby sa predišlo kríženiu čiar, kde prerezať cirkulárne chromozómy alebo či je lepšie nechať pôvodnú orientáciu chromozómu alebo ho otočiť. Viac o tejto minimalizácií sa možno dočítať v práci [3].

Kapitola 3

Definícia problému

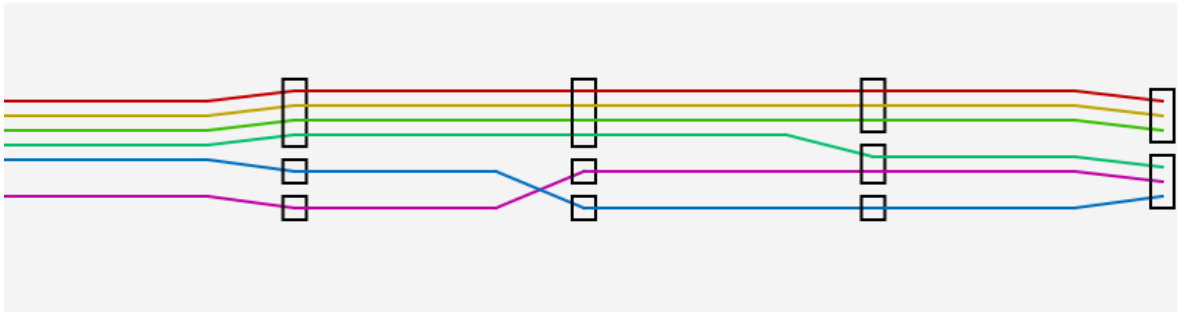
Naším cieľom je vykresliť evolučnú históriu čo najprehľadnejšie. Ako je napísané v predchádzajúcej kapitole v časti 2.3, nástroj EHDraw v snahe zlepšiť čitateľnosť výstupu minimalizuje počet kríženi medzi čiarami. No v tejto práci nás bude zaujímať nielen to, koľko kríženi má výsledný obrázok, ale aj to, kde sa nachádzajú tieto kríženia, pretože na miestach, kde vidíme na obrázku kríženia, očakávame, že sa diali zmeny v chromozómoch. Preto by sme chceli, aby sa kríženia nachádzali na miestach, kde sa skutočne udiali zmeny v chromozómoch. V tejto kapitole sa pozrieme na možnú formuláciu tohto problému. Najskôr budeme uvažovať evolučné histórie bez speciacií, a neskôr sa pozrieme na to, ako sa náš problém zmení, keď povolíme speciácie.

3.1 Dobré a zlé kríženia

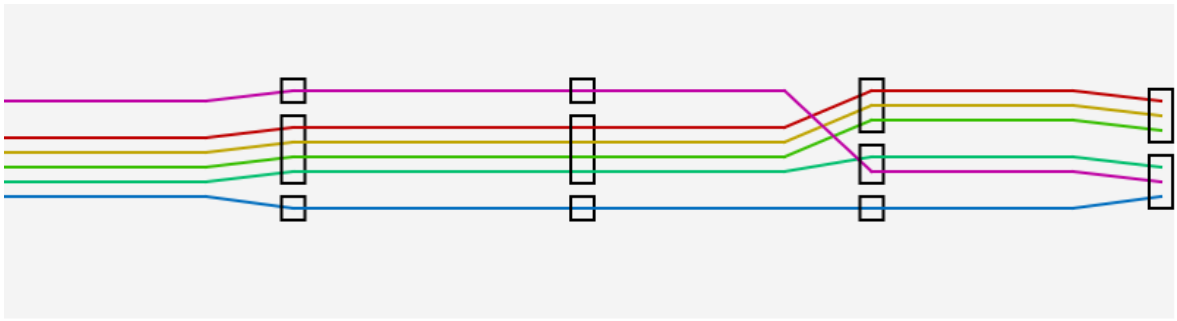
Na obrázku 3.1 vidíme dve rôzne vizualizácie tej istej evolučnej histórie. Na obrázku 3.1a je iba jedno kríženie, ale nachádza medzi dvoma udalosťami, kde nenastala žiadna zmena. Takéto kríženie čiar je umelé a budeme sa snažiť ich odstrániť. Naopak na obrázku 3.1b sú až 4 kríženia, ale nachádzajú sa v mieste, kde sa dejú presuny génov medzi chromozómami. Takéto kríženia nám budú menej prekážať.

Chceli by sme dosiahnuť, aby kríženia sa nachádzali iba na zaujímavých miestach, kde sa niečo deje. Najprv musíme zadefinovať, čo sú zaujímavé miesta. Program EHDraw zobrazuje rôzne udalosti. Napríklad vznik nových génov alebo ich zánik sa dajú jednoducho zobraziť bez toho, aby vo výslednom nakreslení spôsobili kríženia čiar. Na druhú stranu, premiestňovanie génov znamená zmenu ich vzájomnej polohy, a teda aj možný vznik kríženi. Preto pre nás budú zaujímavé tie udalosti, kde dochádza k premiestňovaniu génov, obzvlášť medzi rôznymi chromozómami. Za zaujímavé miesta budeme považovať tie, kde medzi dvoma po sebe nasledujúcimi udalosťami nastal presun génov medzi chromozómami.

Intuitívne potom za dobré kríženia budeme považovať tie, ktoré sa nachádzajú na



(a) Minimálny počet krížení



(b) Kríženia na zaujímavých miestach

Obr. 3.1: Dve vizualizácie tej istej evolučnej histórie. Na obrázku 3.1a je iba jedno kríženie, ale nachádza sa na mieste, kde nenastala žiadna zmena. Na obrázku 3.1b sú 4 kríženia, ale nachádzajú tam, kde dochádza k presunom génov medzi chromozómami.

zaujímavých miestach. Naopak kríženia, ktoré sa nachádzajú na miestach. kde nenastal žiadny presun génov sú neprirodzené, budeme ich považovať za zlé a budeme as snažiť ich minimalizovať.

3.2 Evolučná história ako vrstvový graf

Najskôr si uvedieme pár základných pojmov z teórie grafov. Graf G pozostáva z množiny vrcholov $V(G)$ a množiny hrán $E(G)$. Vrcholy zodpovedajú objektom a hrany znázorňujú vzťahy medzi nimi. Hrany môžu mať smer, vtedy hovoríme o orientovanom grafe. V ohodnotenom grafe má navyše každá hrana pridelenú hodnotu.

Vrstvový graf (*layered graph*) je taký orientovaný acyklický graf G, v ktorom sú vrcholy rozdelené do vrstiev $L_1, L_2, \ldots L_h$ takých, že platí: $V(G) = L_1 \cup L_2 \cup \ldots \cup L_h$, $L_i \cap L_j = \emptyset$ pre $i \neq j$ a pre každú hranu (u, v) platí $u \in L_i, v \in L_j$ pre $i < j$ $i, j \in \{1 \ldots h\}$. V **riadnom vrstvovom grafe** (*proper layered graph*) navyše platí to, že hrany spájajú iba vrcholy zo susedných vrstiev L_i a L_{i+1}. Takýto graf sa používa vtedy, keď chceme zobraziť nejakú hierarchickú štruktúru ako napríklad diagram tried, graf volaní funkcií alebo fylogenetický strom.

Teraz sa vyskúšame na celý problém pozrieť formálnejšie. Danej evolučnej histó-

rii priradíme graf. Za vrstvy budeme považovať genóm po nastaní určitej evolučnej udalosti. Susedné vrstvy sú vo vzťahu predok-potomok. Jednotlivé gény genómu budú predstavovať vrcholy. Hranou spojíme tie gény zo susedných vrstiev, ktoré predstavujú v predkovi aj potomkovi ten istý gén. Gény každej vrstvy sú navyše rozdelené do chromozómov s pevne daným poradím génov. Tento graf označíme G_H. Všetky hrany v ňom smerujú od predka k potomkovi. Graf G_H pripomína riadny vrstvový graf s násobnými hranami.

3.3 Definícia problému

V predchádzajúcej časti sme zadefinovali graf evolučnej histórie G_H. Najskôr intuitívne zadefinujeme dobré a zlé kríženia na tomto grafe, a potom aj formálne.

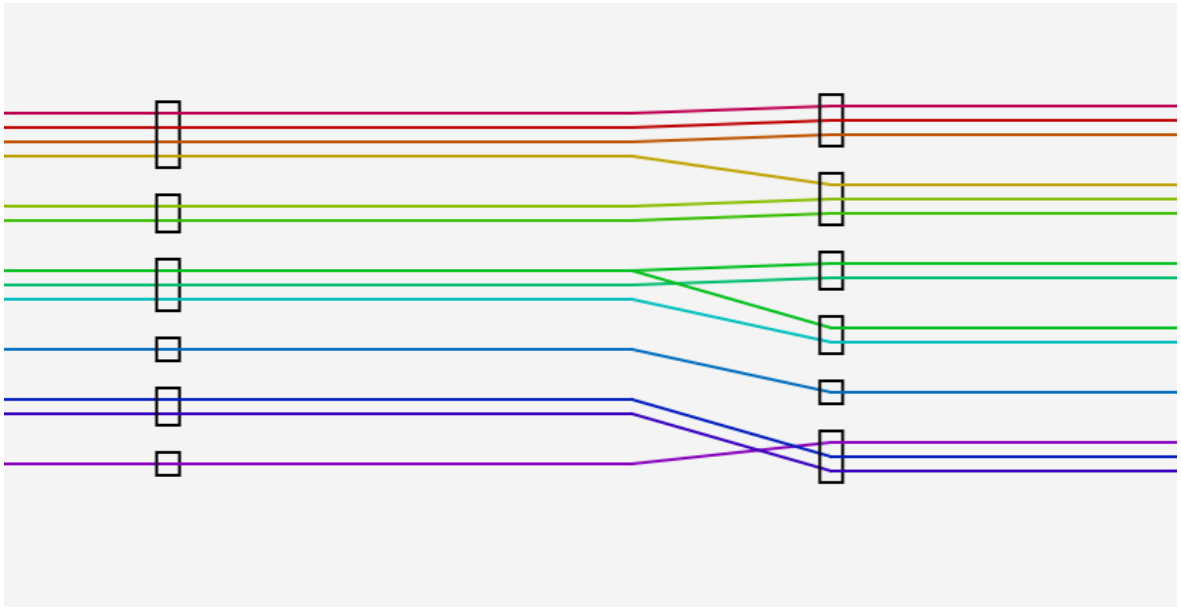
Presuny génov môžeme vidieť na obrázku 3.2. Keď sa pozrieme na dve susedné udalosti, chromozómy, medzi ktorými sa presúvajú gény, vytvárajú jeden komponent súvislosti (za predpokladu, že chromozómy predstavujú vrcholy, medzi ktorými sú násobné hrany génov). Preto chromozómové komponenty medzi susednými udalosťami budú zodpovedať zaujímavým miestam a kríženia v rámci komponentu nám nebudú prekážať. Budeme sa snažiť minimalizovať kríženia medzi komponentami. Dobré kríženia sú tie, ktoré vznikajú vrámci jedného komponentu medzi dvoma susednými udalosťami. Ak nastáva kríženie čiar medzi rôznymi komponentami, ide o zlé kríženia.

Označme podgraf grafu G_H obsahujúci všetky vrcholy a hrany z vrstiev L_i a L_{i+1} ako G_{L_i}, v ktorom navyše máme vrcholy zabalené do chromozómov. Formálne zadefinujeme dobré a zlé kríženia a problém minimálneho počtu zlých krížení:

Def. 1. Majme graf evolučnej histórie G_H s vrstvami $L_1 \dots L_h\}$. Dobré kríženia sú tie, ktoré vznikli medzi vrstvami L_i a L_{i+1} pre $i \in \{1, \dots h-1\}$ a krížiace sa hrany sa nachádzajú v jednom chromozómovom komponente grafu G_{L_i}. Kríženia medzi hranami, ktoré nie sú v jednom komponente súvislosti v grafe G_{L_i}, nazveme zlé kríženia.

Def. 2. Majme graf evolučnej histórie G_H. Pre každú vrstvu $L_i,\ i \in \{1, \dots h\}$, chceme nájsť takú permutáciu chromozómov, aby platilo, že počet krížení medzi hranami, ktoré nie sú v jednom chromozómovom komponente súvislosti v grafe G_{L_i}, bol minimálny. Tento problém nazveme problém minimálneho počtu zlých krížení na grafe evolučnej histórie.

Nevýhodou tejto definície je, že uvažuje iba poradie chromozómov. Nedovoľuje chromozómy otáčať, ani v prípade cirkulárnych chromozómov zmeniť miesto prerezania, a tým znížiť celkový počet krížení.



Obr. 3.2: Ukážka presunov génov medzi chromozómami.

3.4 Graf chromozómov

Na zjednodušenie riešenia prerobíme pôvodný graf nasledujúcim spôsobom. Podobne ako v grafe G_H, aj tu bude vrstva zodpovedať genómu po odohraní niektorej z udalostí. Chromozóm v čase každej udalosti bude predstavovať vrchol. Hranou spojíme tie chromozómy zo susedných vrstiev, ktoré majú aspoň jeden spoločný gén. Týmto hranám dáme váhy podľa počtu spoločných génov. Dostaneme nový graf, ktorý nazveme graf chromozómov a označíme ho G_C. Graf G_C pripomína riadny vrstvový graf s ohodnotenými hranami.

Aj pre tento graf zadefinujeme dobré a zlé kríženia veľmi podobne ako pre G_H:

Def. 3. Majme graf chromozómov G_C s vrstvami $L_1 \dots L_h\}$. Dobré kríženia sú tie, ktoré vznikli medzi vrstvami L_i a L_{i+1} pre $i \in \{1, \dots h-1\}$ a krížiace sa hrany sa nachádzajú v jednom komponente grafu G_{L_i}. Kríženia medzi hranami, ktoré nie sú v jednom komponente súvislosti v grafe G_{L_i}, nazveme zlé kríženia.

Taktiež zadefinujeme problém minimálneho počtu zlých krížení:

Def. 4. Majme graf komponentov G_H. Pre každú vrstvu L_i, $i \in \{1, \dots h\}$, chceme nájsť takú permutáciu chromozómov, aby platilo, že suma súčinov hodnôt krížiacich sa hrán, ktoré nie sú v jednom komponente súvislosti v grafe G_{L_i}, bola minimálna. Tento problém nazveme problém minimálneho počtu zlých krížení na grafe chromozómov.

Ľahko vidno, že úpravou grafu evolučnej histórie na graf chromozómov sa zníži počet dobrých krížení. Graf G_C nezobrazuje presuny génov v rámci jedného chromozómu, a teda ani kríženia, ktoré tým vznikli. Taktiež zanedbáva kríženia pri presune génu do

iného chromozómu s hranami chromozómov, medzi ktorými sa gén presúval. Dôležitejšie však je, či mení počet zlých krížení.

Lema 1. Počet zlých krížení v grafe evolučnej histórie G_H a súčinov krížiacich sa hrán v grafe chromozómov G_C k nemu prislúchajúcemu je rovnaký.

Dôkaz. Predpokladajme, že v grafe G_H existuje dvojica hrán uv a $u'v'$ taká, že vytvára zlé kríženie. ukážeme, že aj v grafe G_C musí existovať dvojica hrán prislúchajúca hranám uv a $u'v'$, ktoré sa krížia.

Z toho, že hrany uv a $u'v'$ tvoria zlé kríženie, vieme, že ich koncové gény sú v rôznych chromozómoch, označme ich c_1, c_2, resp. c'_1, c'_2. Bez ujmy na všeobecnosti môžeme predpokladať, že u sa nachádza skôr ako u' a v sa nachádza neskôr ako v' v grafe G_H. Potom platí, že aj c_1 sa nachádza skôr ako c'_1 a c_2 sa nachádza neskôr ako c'_2 v grafe G_H, pretože pozície génov jedného chromozómu a druhého nemôžu byť premiešané. Pri transformácií grafu G_H na graf G_C sa nemenia relatívne pozície chromozómov, takže aj v grafe G_C platí, že c_1 sa nachádza skôr ako c'_1 a c_2 sa nachádza neskôr ako c'_2. Keďže chromozómy c_1 a c_2, resp. c'_1 a c'_2 sú v grafe G_C spojené hranou, aj v tomto grafe vznikne kríženie. Pri transformácií sa nemení ani to, ktoré chromozómy sú spolu v jednom komponente. Takže hrany c_1c_2 a $c'_1c'_2$ tvoria zlé kríženie aj v grafe G_C. □

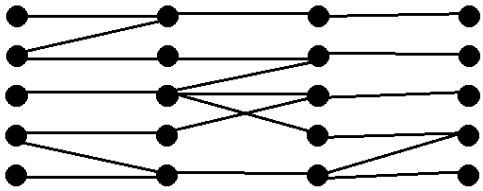
3.5 Graf komponentov

V predchádzajúcej časti sme zostrojili z evolučnej histórie graf chromozómov G_C tak, že sme z chromozómov spravili vrcholy a z génov hrany. Teraz z neho vytvoríme graf komponentov G_K, ktorý využijeme pri riešení problému minimálneho počtu zlých krížení. Najskôr pri vytváraní grafu G_K zanedbáme hodnoty hrán v grafe G_C, ktoré neskôr zohľadníme. Graf komponentov G_K zostrojíme nasledovne:

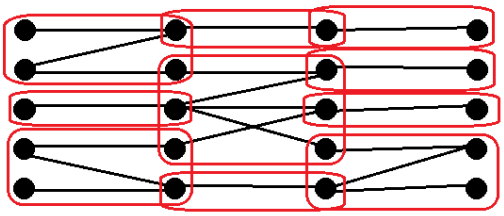
- Pozrieme sa na vrstvy L_i a L_{i+1} a nájdeme v nich všetky komponenty. Každý komponent medzi dvoma susednými vrstvami bude zodpovedať vrcholu. Relatívne pozície vrcholov v rámci jednej vrstvy budú zodpovedať relatívnemu poradie prvých vrcholov z komponentu v grafe G_C.

- Vrcholy, ktoré vznikli z komponentov medzi vrstvami L_{i-1}, L_i a medzi L_i, L_{i+1} spojíme hranou vtedy, keď v grafe G_C vo vrstve L_i majú aspoň jeden spoločný vrchol.

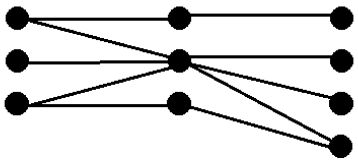
Spôsob, ako vytvoriť graf G_K, môžeme vidieť na obrázku 3.3. Novovzniknutý graf G_K je riadny vrstvový graf. Navyše má tú vlastnosť, že pokiaľ ho vieme nakresliť bez kríženia hrán tak, aby sme zachovali vrstvy, potom pôvodný graf G_C vieme nakresliť bez krížení medzi komponentmi. Túto vlastnosť si ukážeme neskôr. V prípade, že sa



(a) Pôvodný graf s jednoduchými hranami



(b) Komponenty medzi dvoma vrstvami sú zvýraznené červenou



(c) Nový graf G'

Obr. 3.3: Na obrázku je znázornené ako z pôvodného grafu G spravíme graf G'

nedá graf G_K nakresliť bez krížení, chceli by sme zistiť, koľko krížení medzi komponentmi sa vytvorí v pôvodnom grafe a tento počet minimalizovať. Na dosiahnutie toho pridáme hranám hodnoty. V pôvodnom grafe G_C sme jednotlivým hranám pridelili hodnoty podľa počtu génov. V upravenom grafe G_K bude hodnota hrany zodpovedať súčtu všetkých hrán vchádzajúcich do vrcholov, ktoré majú dva susedné komponenty spoločné. Spôsob, ako pridať grafu pri transformácií hodnoty, je znázornený na obrázku 3.4

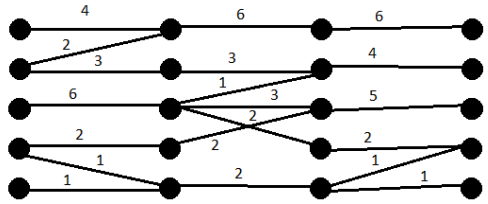
Tento upravený graf opäť spĺňa podmienky riadneho vrstvového grafu s ohodnotenými hranami. V ňom chceme minimalizovať súčin hodnôt krížiacich sa hrán. Problém minimalizácie krížení je v informatike známy, preto môžeme na jeho riešenie použiť už známy algoritmus. Bližšie si ho ukážeme v nasledujúcej kapitole.

Keďže graf komponentov spája komponenty súvislosti susedných vrstiev grafu G_C do jedného vrchola, ľahko vidno, že stratíme informáciu o dobrých kríženiach, ktoré sa v tomto komponente nachádzali.

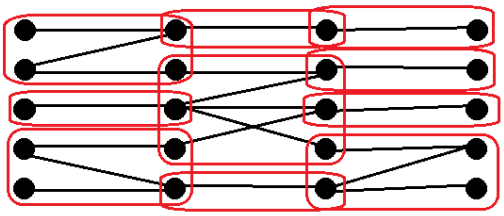
Napriek tomu, že pri transformácií grafu chromozómov na graf komponentov sa môžu niektoré kríženia stratiť, tvrdíme, že z grafu komponentov G_K vieme spätne spraviť graf chromozómov, ktorý má rovnaký počet krížení ako graf G_K.

3.6 Rozšírenie pre histórie so speciáciami

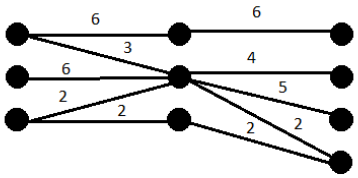
Doteraz sme uvažovali evolučné histórie bez speciácií. Teraz sa pozrieme na to, ako sa zmení náš problém, keď povolíme aj speciácie. Pri speciácií z genómu jedného organizmu vzniknú genómy dvoch nových organizmov. To sa prejaví na grafe evolučnej



(a) Pôvodný graf s ohodnotenými hranami



(b) Komponenty medzi dvoma vrstvami sú zvýraznené červenou



(c) Nový graf G'

Obr. 3.4: Na obrázku je znázornené ako z pôvodného grafu G spravíme graf G' aj s ohodnotenými hranami

histórie tak, že daná vrstva sa rozvetví na dve vrstvy, takže vrstva predok má dvoch potomkov. Potom histórie v oboch vrstvách pokračujú ďalej nezávisle na sebe. Teda vrstvy budeme určovať pre genóm každého organizmu zvlášť. Pokiaľ má genóm aspoň dva gény, pri speciácií dôjde ku kríženiu hrán. Tieto kríženia sú prirodzené, preto ich budeme považovať za dobré. Navyše spĺňajú definíciu 1 dobrých krížení. Preto povoliť speciácie nám nespôsobí žiadne problémy.

Kapitola 4

Algoritmus a implementácia

V tejto kapitole sa pozrieme na to, ako minimalizovať zlé kríženia definované v 4. Pokúsime sa nájsť konkrétny algoritmus na riešenie daného problému a implementujeme ho.

Sugiyamova metóda [18] je známou metódou na vizualizáciu orientovaných grafov. Jej hlavným cieľom je zvýšiť čitateľnosť grafu minimalizovaním počtu krížení. Samotný algoritmus má niekoľko krokov, pre nás však bude podstatná len tá časť, kde sa optimalizuje počet krížení preusporiadaním vrcholov každej vrstvy.

4.1 Minimalizácia krížení

Jeden z problémov, ktorými sa zaoberá teória grafov, je určenie priesečníkového čísla grafu, teda nájsť minimálny počet krížení hrán zo všetkých jeho nakreslení. Tento problém má veľký význam pri vizualizácií grafov, pretože kríženia znižujú čitateľnosť grafu. Našim cieľom je minimalizovať kríženia v riadnom vrstvovom grafe, kde môžeme iba meniť poradie vrcholov v danej vrstve. Tento problém je NP-ťažký [6], a teda zatiaľ nepoznáme žiaden efektívny algoritmus na jeho riešenie. Preto sa pozrieme na heuristiky, ktoré sa používajú.

4.1.1 Počítanie krížení

Najprv sa pozrieme na to, ako počítať kríženia. Naivný algoritmus, ktorý porovnáva všetky hrany medzi dvoma vrstvami, beží v čase $O(|E|^2)$. Hrana vu sa kríži s hranou $v'u'$ práve vtedy, keď na jednej vrstve sa vrchol v nachádza skôr než vrchol v' a na druhej susednej vrstve sa nachádza vrchol u neskôr než vrchol u'. Relatívne pozície vrcholov na jednej vrstve dostaneme porovnaním y-ových (pre sprava doľava) súradníc. Takýto algoritmus na počítanie krížení je postačujúci, pretože v najhoršom prípade môže vzniknúť $\Omega|E|^2$ krížení. Existujú aj ďalšie algoritmy [9, 13, 15], ktoré majú menšiu zložitosť pre málo krížení, my sa však nimi nebudeme bližšie zaoberať.

4.1.2 Jednostranná minimalizácia križení

Namiesto riešenia problému na celom grafe sa najskôr pozrieme na ľahšiu verziu pre dve susedné vrstvy. Navyše, budeme predpokladať, že jedna vrstva už má určené poradie, a my môžeme meniť iba poradie vrcholov v druhej vrstve. Riešenie tohto problému sa využíva v heuristikách na minimalizáciu križení v celom vrstvovom grafe. Naším cieľom je pre daný bipartitný (dvojvrstvový) graf G s vrstvami L_1, L_2 a daným poradím vrcholov vo vrstve L_1 nájsť permutáciu vrcholov vrstvy L_2 takú, aby výsledný počet križení bol minimálny. Už tento problém je NP-ťažký [4], preto sa na riešenie používajú heuristiky. My si ukážeme dve z nich, o ďalších sa dá dočítať v literatúre [19].

Jednou z najpoužívanejších heuristík je **barycentrová heuristika**. Novú pozíciu vrchola v z vrstvy L_1 dostaneme tak, že sa pozrieme na všetky y-ové súradnice susedov vrchola v a jeho nová pozícia sa vypočíta ako aritmetický priemer týchto súradníc. Existujú aj obmeny, kde vypočítaný priemer neslúži priamo na umiestnenie vrchola ale použije sa ako skóre na zoradenie všetkých vrcholov danej vrstvy. Po zbehnutí tejto heuristiky je výsledný počet križení najviac $O(\sqrt{|V|})$ krát horší ako optimum [4].

Ďalšou metódou je **mediánová heuristika**. Podobne ako pri predchádzajúcej heuristike, aj tu sa na určenie pozície vrchola v z vrstvy L_2 budeme pozerať na súradnice jeho susedov. V tomto prípade nová pozícia v bude medián súradníc jeho susedov. Je dokázané, že výsledný počet križení po zbehnutí tejto heuristiky je najviac 3-krát horší ako optimum [4].

Výhodou oboch uvedených heuristík je vlastnosť, že pokiaľ existuje poradie vrcholov vrstvy L_2 také, že výsledný graf neobsahuje kríženia, nájdu takéto poradie [19]. Výpočet pozície jedného vrchola trvá $O(|N(v)|)$, kde $N(v)$ je množina všetkých susedov vrchola v. Z toho vidno, že časová zložitosť pre všetky vrcholy je $O(|E|)$. Následné usporiadanie vrcholov podľa vypočítaného skóre potrebuje čas $O(|L_1| \log |L_1|)$. Napriek horším garantovaným vlastnostiam barycentrovej heuristiky, v praxi sa ukazuje, že dáva lepšie výsledky ako mediánová heuristika [10]. Napríklad pre riedke grafy barycentrová heuristika nájde zvyčajne len o 3% horší výsledok ako optimum.

4.1.3 Viac-vrstvová minimalizácia križení

Teraz sa pokúsime problém rozšíriť na celý graf. Našou úlohou je pre daný vrstvový graf G s vrstvami $L_1, \ldots L_h$ nájsť také permutácie vrcholov $\pi_1, \ldots \pi_h$, aby počet križení v grafe bol minimálny. Riešiť problém naraz pre celý graf je ťažké. Preto si ukážeme heuristiku na riešenie tohto problému.

Najbežnejšia metóda je **prechod po vrstvách** *(layer-by-layer sweep)*. Algoritmus vyzerá tak, že pre vrstvy L_i a L_{i+1} pomocou jednostrannej minimalizácie nájdeme preusporiadanie vrcholov vo vrstve L_{i+1} pre pevné poradie vrcholov vrstvy L_i, pričom i postupne narastá, až kým neprejdeme celý graf. Následne budeme prechádzať graf od

poslednej vrstvy po prvú, pričom opäť riešime jednostranný problém pre vrstvy L_i a L_{i+1}, kde ale pevné poradie vrcholov má vrstva L_{i+1}. Takto prechádzame graf zvyšujúc a znižujúc i dovtedy, kým sa už nedá počet krížení vylepšiť.

4.2 Minimalizácia krížení v ohodnotenom grafe

Pôvodný algoritmus heuristikami minimalizuje kríženia v riadnom vrstvovom grafe. My chceme minimalizovať kríženia v grafe komponentov, ktorý síce spĺňa definíciu riadneho vrstvového grafu, ale navyše jeho hrany sú ohodnotené. Preto pôvodný algoritmus trochu upravíme.

Prvú úpravu spravíme pri počítaní krížení. Ako sme spomenuli v predchádzajúcej kapitole, pri počítaní počtu krížení budeme sčítavať súčin hodnôt krížiacich sa hrán.

Ďalšia úprava je potrebná pri použití barycentrovej heuristiky. V nej sa pre každý vrchol v počíta skóre ako aritmetický priemer súradníc susedov vrchola v. V ohodnotenom grafe namiesto jednoduchého aritmetického priemeru použijeme vážený priemer, teda skóre s_v pre vrchol v vypočítame pomocou vzorca:

$$s_v = \frac{y_1 w_1 + y_2 w_2 + \ldots + y_n w_n}{w_1 + w_2 + \ldots + w_n}$$

Kde y_i je súradnica i-teho suseda u_i vrchola v a w_i je hodnota hrany $u_i v$.

Podobným spôsobom by sa dala upraviť aj mediánová heuristika, ktorá by namiesto počítania mediánu počítala vážený medián.

4.3 Implementácia

Nástroj EHDraw je implementovaný v programovacom jazyku Java. Pôvodne program bol tvorený nasledujúcimi triedami:

- *EHDraw* — hlavná trieda, v ktorej je grafická aplikácia
- *EvolutionTree* — trieda, ktorá reprezentuje evolučnú históriu a obsahuje metódy na jej vykreslenie a optimalizácie. Nachádza sa v nej pomocná trieda *EvolutionNode*, ktorá reprezentuje jednu udalosť evolučnej histórie
- *Chromosome* — trieda predstavujúca jeden chromozóm po niektorej udalosti
- *DrawFactory* — abstraktná trieda, ktorá slúži na vykresľovanie evolučnej histórie. Jej implementácie sú FXDrawFactory, ktorá slúži na vykresľovanie do grafickej aplikácie JavaFX, a SVGDrawFactory, pomocou ktorej môžeme evolučnú históriu exportovať do súboru formátu SVG
- *Settings* — trieda, v ktorej sa nachádzajú globálne nastavenia

- *GeneMeta* — trieda, ktorá obsahuje nastavenia jedného génu
- *StatisticsGenerator, HistoryRandomizer, EvolutionGenerator, EHConverter* — triedy, ktoré slúžia na tvorbu testov a generovanie evolučných histórií.

Aby sme zachovali možnosť minimalizovať všetky kríženia, zvolili sme návrhový vzor stratégia. Pridali sme abstraktnú triedu *MinimizatioStrategy* a jej dve implementácie — *AllCrossingMinimizationStrategy* a *SpuriousCrossingMinimizationStrategy*. Do prvej spomenutej triedy sme presunuli algoritmus na minimalizáciu krížení, ktorý bol implementovaný Radoslavom Chládekom [3]. Náš algoritmus bol implementovaný v druhej triede. Algoritmus na minimalizáciu krížení si môžeme zvoliť v nastaveniach (*Settings* $\longrightarrow$ *Minimize*).

Kapitola 5

Experimentálne vyhodnotenie

V predchádzajúcej časti sme uviedli algoritmus, pomocou ktorého minimalizujeme počet zlých krížení. Chceli by sme ho aspoň čiastočne analyzovať. Preto sa v tejto kapitole pozrieme na to, ako sa náš algoritmus správa na generovaných aj reálnych dátach.

5.1 Pomocné triedy a použitý softvér

Na tvorbu testov boli použité nasledujúce programy:

- *StatisticsGenerator* — je trieda nástroja *EHDraw*. Bola upravená tak, aby vytvárala súbory potrebné pre naše testy, ktoré neskôr popíšeme a aby automaticky ich automaticky spúšťala.
- *HistoryRandomizer* — je trieda nástroja *EHDraw* vytvorená v rámci predchádzajúcej práce [3], náhodne permutuje a otáča chromozómy v každej vrstve. Tým vznikajú histórie s veľkým počtom krížení.
- *EvolutionGenerator* — je pomocná trieda programu *PIVO2* [7]. Trieda slúži na generovanie evolučných histórií vo formáte špecifickom pre PIVO. Ako parametre berie počet listov evolučnej histórie, počet génov v genóme, počet DCJ operácií a počet chromozómov. DCJ (*double cut and join*) operácie prerežú genóm na menšie časti, ktoré sa následne spoja v inom poradí. Tým simuluje zmeny, ktoré nastávajú počas evolúcie.
- *EHConverter* [1] — je program, ktorý konvertuje rôzne formáty evolučných histórií do formátu vhodného pre vstup nástroja *EHDraw*.

5.2 Generované dáta

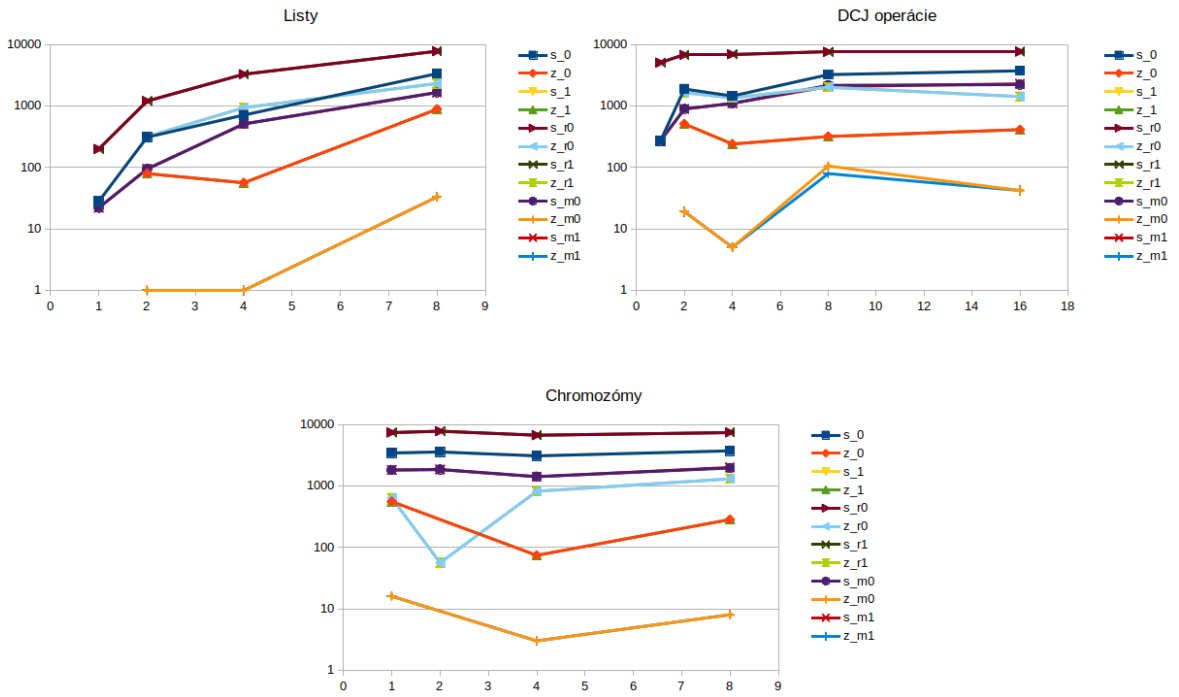
Na evolučných históriách vygenerovaných programom *EvolutionGenerator* sme porovnávali nasledujúce hodnoty:

- s_0 — skóre všetkých krížení vygenerovanej histórie pred optimalizáciou
- z_0 — skóre zlých krížení pred optimalizáciou
- $s_{r,0}$ — skóre všetkých krížení po znáhodnení triedou *HistoryRandomizer*
- $z_{r,0}$ — skóre zlých krížení po znáhodnení
- $s_{m,0}$ — skóre všetkých krížení po optimalizácií všetkých krížení triedou *AllCrossingsMinimizationStrategy*
- $z_{m,0}$ — skóre zlých krížení po optimalizácií všetkých krížení
- s_1 — skóre všetkých krížení vygenerovanej histórie po optimalizácií zlých krížení
- z_1 — skóre zlých krížení po optimalizácií zlých krížení
- $s_{r,1}$ — skóre všetkých krížení po znáhodnení a optimalizácií zlých krížení
- $z_{r,1}$ — skóre zlých krížení po znáhodnení a optimalizácií zlých krížení
- $s_{m,1}$ — skóre všetkých krížení po optimalizácií všetkých krížení a zlých krížení
- $z_{m,1}$ — skóre zlých krížení po optimalizácií všetkých krížení a zlých krížení

Skóre všetkých krížení obsahuje aj informáciu o čiarkovaných čiarach. Počítanie tohto skóre je rovnaké ako v predchádzajúcej práci [3]. Skóre zlých krížení je počítané na grafe komponentov. Keďže nepoznáme najlepšie skóre, budeme sa pozerať na vyššie uvedené hodnoty a tie porovnávať. Chceme zistiť, či algoritmus funguje rovnako dobre na všetkých vstupoch alebo je závislý na počiatočnom usporiadaní chromozómov. Preto budeme porovnávať pôvodný vstup, znáhodnený vstup, ktorý má veľa krížení, a vstup s minimalizovanými kríženiami. Zaujíma nás nielen počet zlých krížení, ale aj počet všetkých krížení.

Dané hodnoty sme skúmali pre rôzne hodnoty parametrov počtu listov, DCJ operácií a chromozómov. Jeden parameter sa menil a ostatné sme držali na predvolených hodnotách. Ako predvolenú hodnotu sme použili pre počet listov 8, pre počet génov 32, pre počet DCJ operácii 8 a pre počet chromozómov 8. Výsledné grafy môžeme vidieť na obrázku 5.1. Na x-ovej osi je hodnota skúmaného parametra a na y-ovej osi sú zobrazené hodnoty skóre. Os y je zobrazená v logaritmickej mierke.

Z grafov možno vidieť, že náš algoritmus nič neoptimalizuje. Dôvodom bude pravdepodobne to, že veľa krížení zanedbáva. Taktiež problém môže byť v tom, že na optimalizáciu krížení v grafe komponentov používame heuristiky, ktoré nemusia nájsť najmenej krížení. Výsledok, že algoritmus nebude veľmi dobrý v minimalizácií všetkých krížení bol očakávaný, pretože veľa z nich zanedbáva. Napriek tomu sme si mysleli, že



Obr. 5.1: Experimentálne vyhodnotenie generovaných dát

Tabuľka 5.1: Tabuľka skúmaných skóre na reálnych dátach

	s_0	z_0	s_1	z_1	$s_{r,0}$	$z_{r,0}$	$s_{r,1}$	$z_{r,1}$	$s_{m,0}$	$z_{m,0}$	$s_{m,1}$	$z_{m,1}$
Mammal1	681	91	681	91	6583	4323	6583	4323	626	9	628	9
Mammal2	384	0	384	0	4660	40	4660	40	384	0	384	0
Mammal3	342	0	342	0	3326	0	3326	0	342	0	342	0

bude lepší v minimalizácií zlých krížení. Pôvodný algoritmus, ktorý minimalizuje počet všetkých krížení, je lepší aj minimalizácií zlých krížení.

Na grafe pre DCJ operácie môžeme vidieť, že algoritmus dokáže znížit počet zlých krížení a mierne aj všetkých krížení. V tomto prípade sa znížil počet krížení na už predtým optimalizovanom vstupe.

5.3 Reálne dáta

Okrem generovaných dát sme otestovali náš algoritmus aj na reálnych dátach. Znovu nás budú zaujímať skóre uvedené v predchádzajúcej časti.

Dáta pochádzajú z článku od Foote a spol. [5]. Boli upravené programom PIVO 2. Predstavujú vybrané gény človeka, ktoré sa nachádzajú aj u iných cicavcov.

Prvá história skúma 31 génov človeka, ktoré sa nachádzajú v prvom chromozóme. Vyskytuje sa v nej 7 druhov — človek, myš, lama, kosatka, krava, pes a mrož. Druhá

história skúma 30 génov z druhého ľudského chromozómu u tých istých druhov. Tretia história obsahuje iných 30 génov z druhého chromozómu u 5 druhov — človek, myš, kosatka, krava a pes.

Z dát vidno, že náš algoritmus ani na reálnych dátach nezlepšil počet všetkých krížení. Taktiež na žiadnom vstupe nezlepšil ani počet zlých krížení. V prípade tretej histórie sa ani po znáhodnení v nej nevyskytli žiadne zlé kríženia, ktoré by bolo treba optimalizovať. Naopak môžeme si všimnúť, že v prípade prvej histórie zlé kríženia sú lepšie minimalizované pôvodným algoritmom ako tým naším.

Záver

Výsledkom tejto práce je formálna definícia dobrých a zlých krížení v evolučnej histórií a problému minimalizácie zlých krížení. Najprv sme zadefinovali evolučnú históriu ako riadny vrstvový graf a v ňom sme zaviedli definície dobrých a zlých krížení. Potom sme zostrojili graf chromozómov, v ktorom sme opäť zadefinovali dobré a zlé kríženia. Ukázali sme, že hoci v grafe chromozómov je celkový počet krížení menši ako v pôvodnom grafe histórie, zlé kríženia sa zachovávajú. Nakoniec sme zadefinovali graf komponentov, čím sme pretransformovali pôvodný problém na známy problém minimalizácie počtu krížení v grafe. V literatúre sme našli heuristiku, ktorá rieši tento problém, a implementovali sme ju. Žiaľ, ukázalo sa, že tento spôsob nie je vyhovujúci na riešenie nášho problému. Vo väčšine prípadov sa nepodarilo nič optimalizovať. Možných dôvodov je viacero, napríklad, že počas transformácie problému sme stratili dôležité informácie kríženiach a preto nebolo možne minimalizovať celkový počet krížení. Taktiež problém môže byť v tom, že heuristika na nájdenie minimálneho počtu krížení v grafe komponentov nemusí nájsť najlepšie možné riešenie, a teda nepomôže optimalizovať zlé kríženia. Ako ďalšie pokračovanie v práci by mohlo byť vyskúšanie presných metód na minimalizáciu krížení v grafe komponentov, a tým zníženiu počtu zlých krížení. Taktiež sa dá zamyslieť nad inými definíciami problému, ktoré by boli dobre riešiteľné a vedeli by znižovať počet zlých krížení.

Literatúra

[1] Ročníkový projekt: Skripty na konverziu formátov evolučných histórií. `http://davinci.fmph.uniba.sk/~chladek8/rocnikovy_chladek`.

[2] Axel Künstner a kol. The genome of the trinidadian guppy, poecilia reticulata, and variation in the guanapo population. `https://www.researchgate.net/figure/Whole-genome-alignment-between-Guppy-and-Medaka-CIRCOS-plot-of-syntenic-rela fig3_312181649`. Accesed Jul 15, 2019.

[3] Radoslav Chládek. Vizualizácia evolučných histórií genómov s viacerými chromozómami. Bakalárska práca, Univerzita Komenského v Bratislave. 2017.

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

[5] Andrew D Foote, Yue Liu, Gregg WC Thomas, Tomáš Vinař, Jessica Alföldi, Jixin Deng, Shannon Dugan, Cornelis E van Elk, Margaret E Hunter, Vandita Joshi, et al. Convergent evolution of the genomes of marine mammals. *Nature genetics*, 47(3):272, 2015.

[6] Michael R Garey and David S Johnson. Crossing number is np-complete. *SIAM Journal on Algebraic Discrete Methods*, 4(3):312–316, 1983.

[7] Albert Herencsár. An improved algorithm for ancestral gene order reconstruction. Diplomová práca, Univerzita Komenského v Bratislave. 2014.

[8] Jaime Huerta-Cepas, Joaquín Dopazo, and Toni Gabaldón. Ete: a python environment for tree exploration. *BMC Bioinformatics*, 11(1):24, Jan 2010.

[9] Michael Jünger. Simple and efficient bilayer cross counting. *Graph Algorithms and Applications 5*, 5:179, 2006.

[10] Michael Jünger and Petra Mutzel. 2-layer straightline crossing minimization: Performance of exact and heuristic algorithms. In *Graph Algorithms And Applications I*, pages 3–27. World Scientific, 2002.

[11] Jaroslava Kokavcová. Ročníkový projekt: Vizualizácia evolučných histórií genómov. 2017-2018, `http://www.st.fmph.uniba.sk/~kokavcova20/rp.html`.

[12] Martin I Krzywinski, Jacqueline E Schein, Inanc Birol, Joseph Connors, Randy Gascoyne, Doug Horsman, Steven J Jones, and Marco A Marra. Circos: An information aesthetic for comparative genomics. *Genome Research*, 2009.

[13] Hiroshi Nagamochi and Nobuyasu Yamada. Counting edge crossings in a 2-layered drawing. *Information processing letters*, 91(5):221–225, 2004.

[14] Oscar Robinson, David Dylus, and Christophe Dessimoz. Phylo.io : Interactive viewing and comparison of large phylogenetic trees on the web. *Molecular Biology and Evolution*, 33(8):2163–2166, 2016.

[15] Georg Sander. Graph layout through the vcg tool. In *International Symposium on Graph Drawing*, pages 194–205. Springer, 1994.

[16] Schrisfield. Phylogenetic tree. `https://en.wikipedia.org/wiki/File:Phylogenetic_Tree.pdf`. Accesed Mar 22, 2019.

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

[18] K. Sugiyama, S. Tagawa, and M. Toda. Methods for visual understanding of hierarchical system structures. *IEEE Transactions on Systems, Man, and Cybernetics*, 11(2):109–125, Feb 1981.

[19] Roberto Tamassia. *Handbook of Graph Drawing and Visualization (Discrete Mathematics and Its Applications)*. Chapman & Hall/CRC, 2007.