UNIVERZITA KOMENSKÉHO V BRATISLAVE FAKULTA
MATEMATIKY FYZIKY A INFORMATIKY

ANOTÁCIA ZHLUKOV GÉNOV

2011 Milan Mikula

UNIVERZITA KOMENSKÉHO V BRATISLAVE FAKULTA MATEMATIKY FYZIKY A INFORMATIKY

ANOTÁCIA ZHLUKOV GÉNOV

Bakalárska práca

Študijný program: Informatika

Študijný odbor: 921 Informatika

Školiace pracovisko: Katedra Informatiky FMFI

Školiteľ: Mgr. Tomáš Vinař, PhD.

Evidenčné číslo: ac5df09d-b714-4992-a40a-57dac766c78e

Bratislava 2011

Milan Mikula

Univerzita Komenského v Bratislave
Fakulta matematiky, fyziky a informatiky

ZADANIE ZÁVEREČNEJ PRÁCE

Meno a priezvisko študenta: Milan Mikula
Študijný program: informatika (Jednoodborové štúdium, bakalársky I. st., denná forma)
Študijný odbor: 9.2.1. informatika
Typ záverečnej práce: bakalárska
Jazyk záverečnej práce: slovenský

Názov: Anotácia zhlukov génov

Cieľ: Asi 5% ľudského genómu tvoria oblasti, v ktorých sa vyskytuje veľa takmer identických kópií toho istého génu. Cieľom práce bude adaptovať existujúci nástroj Exonerate na zarovnávanie proteínu ku genómu, aby fungoval dobre aj v takýchto obtiažnych regiónoch.

Vedúci: Mgr. Tomáš Vinař, PhD.

Dátum zadania: 27.10.2010

Dátum schválenia: 02.11.2010

doc. RNDr. Daniel Olejár, PhD.
garant študijného programu

.....
študent

.....
vedúci

Abstrakt

V genóme eukaryotických organizmov sa vyskytujú zhluky génov, čo sú sekvencie DNA, v ktorých sa opakujú kúsky sekvencií iba s malými zmenami. Existujúce programy na anotáciu génov, ako napríklad Exonerate, ktorý vytvára anotáciu pomocou proteínových zarovnaní, nefungujú na týchto úsekoch DNA s dobrými výsledkami.

V našej práci používame program Exonerate na vytvorenie predbežnej anotácie obsahujúcej množstvo zmätočných transkriptov, ktorú potom upravujeme novovytvoreným algoritmom.

Problém anotácie sme transformovali na úlohu z teórie grafov, ktorú sme riešili heuristikou. Výsledky algoritmu sme testovali pri použití rôznych vstupných parametrov na zhlukoch génov AMY a PRAME na DNA sekvenciách z makaka a človeka.

Náš algoritmus splnil očakávania a podstatne vylepšil výstup programu Exonerate, ktorý teraz zmysluplne anotuje aj sekvenciu obsahujúcu zhluky génov.

Kľúčové slová: anotácia, zarovnanie, gén, genóm

Abstrakt

In genome of eukaryotes are found clusters of genes, that are DNA segments in which parts of sequences with only minor modifications repeat. Existing programs for genes annotation, such as Exonerate(which creates an annotation using protein alignment) do not return correct results on these segments of DNA.

In our thesis we use the program Exonerate to create preliminary annotation containing many confounding transcripts, which are altered by our algorithm.

We transformed the annotation problem to graph theory problem, which we solve using heuristics. We tested results of our algorithm using different input parameters for the AMY and PRAME gene clusters situated on DNA sequences from macaques and humans.

Our algorithm has fulfilled expectations and significantly improved output of Exonerate, which is now usefull in annotation of sequences containing clusters of genes.

Key words: annotation, alignment, gene, genome

Čestné prehlásenie

Čestne prehlasujem, že túto prácu som vypracoval samostatne, s použitím uvedených zdrojov.

Poďakovanie

Chcel by som sa poďakovať môjmu vedúcemu, Mgr. Tomášovi Vinařovi, PhD, za rady, pomoc, trpezlivosť a čas, ktorý mi venoval. Ďalej mojej rodine a priateľom za povzbudenie a podporu.

Obsah

Zoznam obrázkov

Zoznam tabuliek

Úvod

Najnovšie technológie sekvenovania DNA produkujú stále väčšie množstvo biologických dát – DNA sekvencií najrôznejších organizmov. Nevyhnutným krokom pri skúmaní týchto sekvencií je anotácia, teda určenie polohy a štruktúry génov, ktoré sa v tejto sekvencii nachádzajú.

Obzvlášť náročná je táto úloha v úsekoch, v ktorých počas evolúcie došlo k zduplikovaniu niektorých častí. Takto vznikajú zhluky génov, čo sú úseky, v ktorých sa vyskytujú gény vo viacerých kópiách, ktoré sa navzájom odlišujú iba niekoľkými mutáciami. V tejto práci sa zaoberáme anotáciou týchto zhlukov génov za pomoci zarovnania DNA sekvencie zo známymi proteínmi. Základom našej práce je použitie programu Exonerate, ktorý však na týchto ťažkých sekvenciách nepracuje optimálne. Hlavným prínosom tejto práce je sformulovanie problému anotácie, za pomoci výstupov programu Exonerate, ako dobre definovaného problému na špeciálne zostavenom grafe. Tento problém riešime heuristickým algoritmom, ktorého funkčnosť ukážeme na reálnych dátach zhlukov génov v genómoch človeka a makaka.

V prvej kapitole vysvetlíme kľúčové biologické a informatické pojmy ako DNA, gén (a jeho štruktúra), proteín, proteosynéza, anotácia, zarovnanie. Súčasťou kapitoly je aj popis fungovania programu Exonerate. V druhej kapitole popíšeme čo sú to zhluky génov a aké problémy má program Exonerate na takýchto úsekoch. V nasledujúcich kapitolách budeme pracovať na vlastnom riešení problému. V samostatnej kapitole popíšeme nami zvolenú reprezentáciu zhluku génov grafom. V ďalšej kapitole formalizujeme problém a navrhneme algoritmus riešiaci tento problém. V poslednej kapitole zhodnotíme našu úspešnosť pri riešení problému na skutočných dátach.

Kapitola 1

Anotácia génov pomocou proteínových zarovnaní

1.1 Základné pojmy

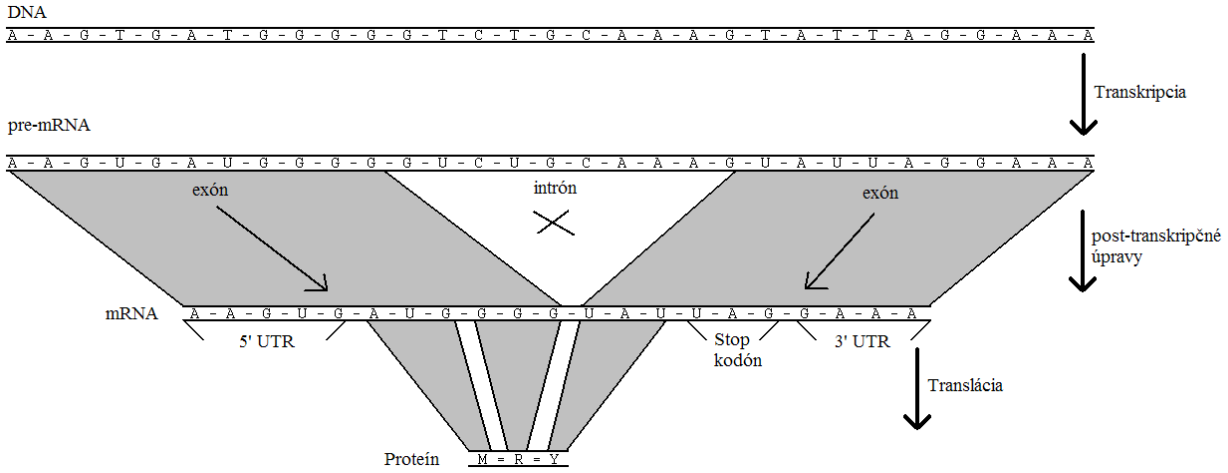
Genetická informácia organizmu je uložená v niekoľkých molekulách DNA (deoxiribonukleová kyselina). DNA sa skladá z reťazca nukleotidov, pričom každý je vlastne jednou zo štyroch dusíkatých báz: adenín, guanín, cytozín a tymín. V DNA je možné aj určiť smer reťazca. Konce reťazca DNA sa nazývajú 5' koniec a 3' koniec, potom smer od 5' konca do 3' konca nazveme kladným a opačný záporným. DNA sa v bunke nachádza vo forme dvoj-závitnice, kde sú dva reťazce DNA k sebe naviazané v opačnom smere a adenín v jednom vlákne sa páruje s tymínom z druhého vlákna a guanín s citozínom.

Pre naše účely sa môžeme na DNA pozerať ako na slovo nad štvor-písmenovou abecedou

$$\Sigma_D = \{A, C, G, T\}$$

RNA (ribonukleová kyselina) je podobná molekula ako DNA, ktorá sa podieľa na spracovaní informácie z DNA. Reťazec RNA má jedno vlákno, inú chemickú štruktúru a namiesto tymínu uracil.

Úsek DNA kódujúci biologický produkt, najčastejšie proteín, na nazýva gén. Proteín (bielkovina) je základnou stavebnou a funkčnou jednotkou buniek, a tvorí približne 37% organických látok bunky. Proteín je reťazec aminokyselín, pričom na stavbe proteínu sa



Obrázok 1.1: Zjednodušený príklad syntézy proteínu z génu.

podieľa 20 aminokyselín. Podobne proteín budeme reprezentovať ako slovo nad abecedou

$$\Sigma_P = \{A, C, D, E, F, G, H, I, K, L, M, N, P, Q, R, S, T, V, W, Y\}$$

Medzi úsekmi kódujúcimi proteíny – génmi sa nachádzajú medzi-génové úseky. Proteín sa z génu vytvorí zložitým biochemickým procesom – proteosyntézou, tá pozostáva z dvoch fáz: transkripcie a translácie. Počas transkripcie sa najskôr informácia z DNA prepíše do pre-mRNA. V tejto fáze sa vystrihnú niektoré nekódujúce úseky – *intróny* ktoré sa ďalej nepodieľajú na tvorbe proteínu. Intrón je ohraničený dvojicou sekvencií donor a akceptor. Nevystrihnuté sekvencie sa nazývajú *exóny* a sú kódujúce, s výnimkou prvého exónu ktorý obsahuje nekódujúcu sekvenciu (5' UTR) a posledného ktorý tiež obsahuje nekódujúcu sekvenciu (3' UTR). Výsledná mRNA obsahuje všetky exóny. Pri translácii sa podľa mRNA syntetizuje samotný proteín. Každú aminokyselinu proteínu kóduje trojica báz – kodón. Možných trojíc báz je viac ako aminokyselín, takže nie každá trojica kóduje aminokyselinu. Väčšina aminokyselín je kódovaná viacerými spôsobmi. Taktiež existujú špeciálne stop kodóny, signalizujúce koniec kódujúceho úseku, a štart kodón ktorý sa nachádza na začiatku kódujúceho úseku, ktorý taktiež kóduje aminokyselinu metionín.

1.2 Anotácia a zarovnanie

Medzi kľúčové problémy, ktoré bioinformatika rieši, patrí anotácia a zarovnanie.

Anotácia znamená priradenie biologického významu neznámej DNA sekvencii, čiže odpovedá na otázku, kde sa v DNA nachádzajú gény. Formálne ide o priradenie každej báze vstupnej sekvencie prvok z množiny biologických významov. V tejto množine sa nachádza napríklad gén, medzigénový úsek, exón, intrón a podobne. Keďže napríklad čo je gén a čo nie je sa nedá jednoznačne definovať, musíme si dodefinovať pravdepodobnostný model ktorý nám pomôže definovať úlohu. Ako model sa často používa HMM (skrytý markovov model), podobný konečným automatom, ktorý v každom stave s určitou pravdepodobnosťou prejde do iného stavu, a s určitou pravdepodobnosťou emituje znak zo svojej pracovnej abecedy. Potom úloha najpravdepodobnejšej anotácie sekvencie sa formuluje ako: aká je najpravdepodobnejšia postupnosť stavov, ktorá emitovala túto sekvenciu? Samotná úloha sa ďalej rieši napríklad Viterbiho algoritmom [7], ktorý je založený na dynamickom programovaní. V prípade že poznáme aký proteín je kódovaný v DNA sekvencii a chceme nájsť ako presne, môžeme použiť zarovnanie proteínu ku genómu.

Zarovnanie je spárovanie dvoch (alebo viacerých) sekvencií tak aby k sebe "pasovali". Teda vložíme medzery tak aby bolo veľa rovnakých báz na rovnakom mieste a čo najmenej medzier. Aby sme úlohu formalizovali tak zavedieme skórovanie, ktoré nám udá ako dobre k sebe zarovnané sekvencie "pasujú", a budeme sa teda snažiť nájsť zarovnanie s maximálnym skóre. Napríklad jednoduché skórovanie môže byť za zhodnú dvojicu báz $+1$, za nezhodnú -1, za medzeru -1. Zložitejšie skórovanie je afínne skórovanie medzier: viacej medzier po sebe pravdepodobne spolu súvisí, preto zavedieme penále za začatie medzery o a o predĺženie medzery e, tým pádom môžeme umožniť aj existenciu dlhých medzier v najlepšom zarovnaní. Samozrejme modely skórovania môžu byť ďaleko zložitejšie. Zarovnanie môže byť lokálne alebo globálne. Pri globálnom zarovnaní sa snažíme zarovnať celé vstupné sekvencie k sebe, tak aby výsledné zarovnanie malo maximálne skóre. Pri lokálnom zarovnaní hľadáme zarovnanie podsekvencií vstupných sekvencií s maximálnym skóre.

Problém lokálneho aj globálneho zarovnania sa rieši dynamickým programovaním. V prí-

```
A C T - - G - A C C T G
| | | | | | | | | | |
A T T G T G A A A - T G
```

Obrázok 1.2: Príklad globálneho zarovnania sekvencií ACTGACCTG a ATTGTGAATG. Zarovnanie má skóre 1 podľa skórovania: zhoda +1, nezhoda −1, medzera −1.

pade globálneho zarovnania ide o Needleman-Wunchov algoritmus, v prípade lokálneho zarovnania sa používa jeho modifikácia Smith-Waretmanov algoritmus. Veľkou nevýhodou týchto algoritmov je ich časová a pamäťová zložitosť $O(mn)$, kde m, n sú dĺžky vstupných sekvencií. V praxi je pre použitie týchto algoritmov kvôli nárokom problematické, najmä ak ide o zarovnania dlhých sekvencii. Preto sa používajú heuristické metódy ktoré nezaručujú optimálny výsledok, ale majú podstatne menšie pamäťové a časové nároky. V týchto heuristických metódach je snaha použiť dynamické programovanie na čo najmenšie oblasti. Jeden z programov ktorý používa heuristické metódy je Exonerate.

1.3 Exonerate

V tejto práci nás zaujíma anotácia častí DNA pomocou známych sekvencií proteínov. Proteíny nemusia byť z toho istého organizmu. Na tento účel používame program Exonerate [6].

Exonerate zarovnáva daný proteín k DNA sekvencii. Vstupnú DNA sekvenciu nazveme databáza (target), a vstupný proteín dotaz (query). Cieľom programu je nájsť na ktorých miestach je v DNA sekvencii zakódovaný vstupný proteín. Program určuje aj presné hranice intrónov a exónov, čím vlastne vytvára anotáciu danej sekvencie.

Exonerate podporuje použitie rôznych modelov modelujúcich zarovnanie. Modelov bez možnosti medzier, modelu afinného skórovania medzier a aj zložité modely podobné konečným automatom, ktoré odzrkadľujú biologickú štruktúru génu. Tieto zložitejšie modely v každom svojom stave emitujú jeden stĺpec znakov zo zarovnania. Prechody medzi stavmi a aj emisie majú určenú pravdepodobnosť. Skóre zarovnania je potom pravdepodobnosť s akou toto zarovnanie model vygeneruje. Jeden z takýchto modelov, ktorý Exonerate používa je zobrazený na obrázku 1.3. Keďže proteín môže byť vo vstupnej DNA sekvencii zakódovaný

v oboch smeroch, je model zložený z dvoch častí, jednej pre každý smer. Stavy zhoda, vlož, zmaž modelujú exón ktorý je v dotaze a v databáze veľmi podobný. Stav intrón modeluje veľkú medzeru v databáze (intrón). Stavy donor a akceptor modelujú prechody medzi exónom a intrónom.

Tento model by sme mohli použiť na nájdenie zarovnania, ktoré má najvyššie skóre podľa tohto modelu. Postupovali by sme tak, že by sme dynamickým programovaním vypĺňali celú maticu A rozmerov $m \times n$, kde m a n sú dĺžky databázy a dotazu. Políčko $A[i, j]$ predstavuje najlepšie zarovnanie prvých i a j znakov. Časová aj priestorová zložitosť tohto dynamického programovania je $O(mn)$, čo je pre praktické účely nevyhovujúce. Program Exonerate preto používa heuristický algoritmus, ktorý ale negarantuje nájdenie najlepšieho riešenia. Snahou bude aplikovať časovo náročné dynamické programovanie na čo najmenšie časti matice.

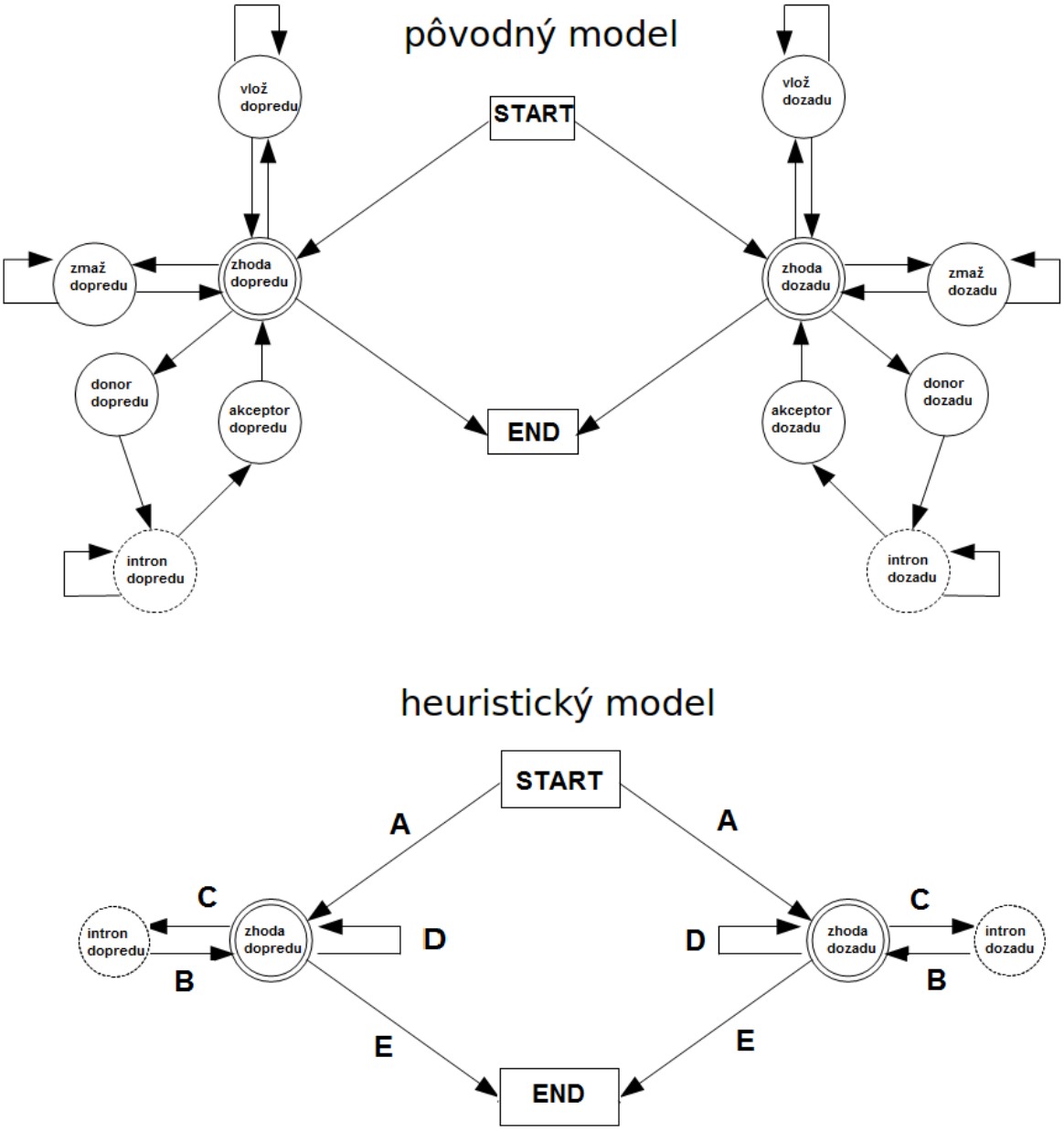
Program funguje v dvoch fázach, v prvej fáze sa určia jadrá zobrazenia (HSP) čo sú zarovnania krátkej dĺžky s vysokým skóre, v druhej fázy sa HSP spoja pomocou dynamického programovania. Samotný algoritmus dynamického programovania sa odvodí od skórovacieho modelu.

Pri tvorbe HSP sa najskôr vytvorí zoznam k-tic z dotazu ktoré sa porovnávajú s k-ticamy z databázy, a tie zarovnania ktoré vyhovejú sa po oboch stranách rozširujú, kým stúpa skóre.

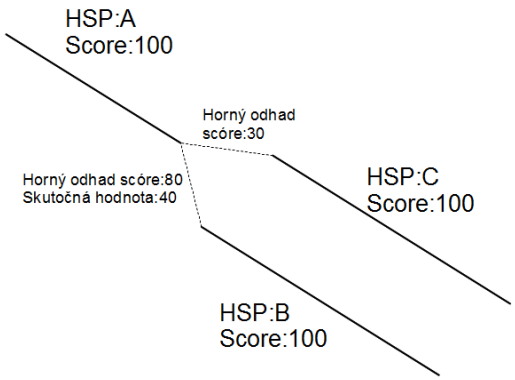
Zaujímavejšou časťou je spájanie HSP do výsledného zarovnania. Ako prvé sa určia úseky (SAR) na ktoré sa použije dynamické programovanie. SAR sa vytvorí na začiatku a konci každého HSP, taktiež v prípade blízkych HSP sa na ich hranici vytvorí spoločný SAR. Vzdialené HSP medzi ktorými je veľká medzera, potenciálne reprezentujúcej intrón, sa taktiež prepoja. Aby sa dynamické programovanie sa neaplikovalo na všetky SAR, aplikuje sa iba na najperspektívnejšie SAR, na tento účel sa použije BSDP (Bounded Spare Dynamic Programing). BSDP funguje na princípe niekoľkých prioritných frontov. Pre každý SAR sa vypočíta maximálne možné skóre. Pre každý HSP sa zostaví prioritný front obsahujúci všetky SAR a ich maximá, ktoré v ňom končia. A taktiež sa zostaví globálny prioritný front obsahujúci maximá s ostatných frontov. V priebehu algoritmu sa vždy nájde najperspektívnejší SAR na ktorý sa aplikuje dynamické programovanie a pôvodné maximálne možné skóre sa nahradí

skutočným. Akonáhle maximálna hodnota vo fronte je už skutočná, t.j. všetky maximálne možné skóre sú menšie, je použité zarovnanie prislúchajúce tejto hodnote. Algoritmus končí akonáhle celkové zarovnanie neobsahuje žiadnu nevypočítanú hodnotu. Výsledok BSDP je rovnaký ako keby sa dynamické programovanie použilo na všetky oblasti.

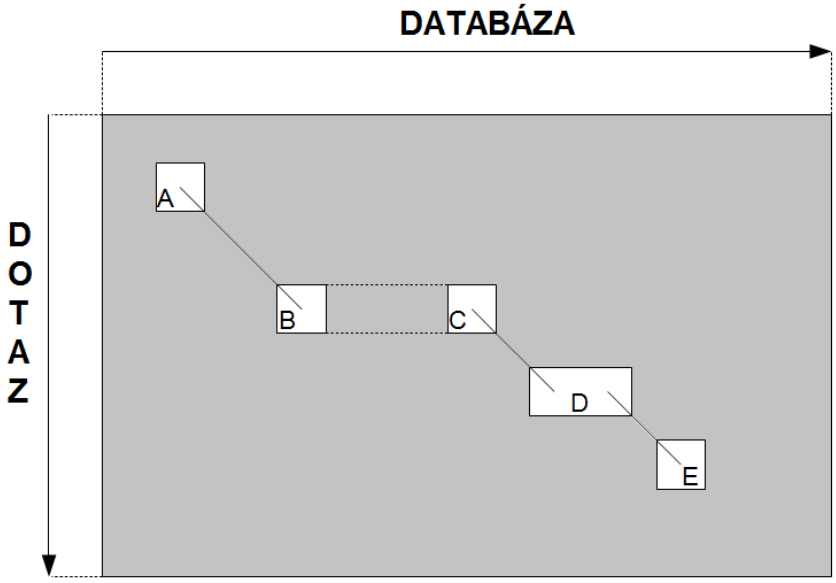
Program Exonerate automaticky generuje rôzne algoritmy dynamického programovania pre rôzne typy SAR. Aby sme mohli toto zabezpečiť musíme pôvodný model zarovnania rozdeliť na časti zodpovedajúce jednotlivým SAR. V pôvodnom modeli sa určia portal-stavy ktoré reprezentujú zarovnanie v rámci HSP, a span-stavy ktoré reprezentujú dlhé medzery znamenajúce intrón. S pôvodného modelu zarovnania sa vytvorí heuristický model v ktorom sú iba začiatočný stav, koncový stav, portal-stavy a span-stavy. V tomto heuristickom modeli každý prechod medzi stavmi zodpovedá jednému z odvodených modelov tak, aby vystihoval pôvodný model s ktorého bol vytvorený. Každý odvodený model, teda prechod medzi portal a span stavmi, potom popisuje jeden typ SAR. Tento odvodený model sa použije pri výpočte zarovnania SAR dynamickým programovaním. Jeden z možných modelov zarovnania a taktiež jeho prevod na heuristický model popíšeme na obrázku 1.3.



Obrázok 1.3: Pôvodný model obsahuje stavy na modelovanie exónov, intrónov, a prechodov medzi nimi.
Model zarovania obsahuje portal-stav zhoda, ktorý modeluje zhodné úseky exónu, čo sú vlastne HSP. Span-stav intrón modeluje dlhé medzery medzi HSP. každý z prechodov v odvodenom heuristickom modeli modeluje typ SAR ktorý spája tieto oblasti. Označenie prechodov korešponduje s označením SAR na obrázku 1.5



Obrázok 1.4: Príklad fungovania BSDP. Máme dva SAR A→B a A→C a prioritný front pre HSP A. Keď sa vypočíta skutočná hodnota skóre pre SAR A→B, ktorá je vyššia ako horný odhad pre A→C, tak skóre pre SAR A→C sa vôbec nebude počítať.



Obrázok 1.5: Obrázok ukazuje rôzne druhy SAR, na ktoré sa bude aplikovať dynamické programovanie. A je SAR na začiatku HSP, D spája dva blízke HSP, B a C sú prepojené, lebo táto medzera môže predstavovať intrón. E sa nachádza na konci HSP

Kapitola 2

Problémy pri anotácii zhlukov génov

2.1 Zhluky génov

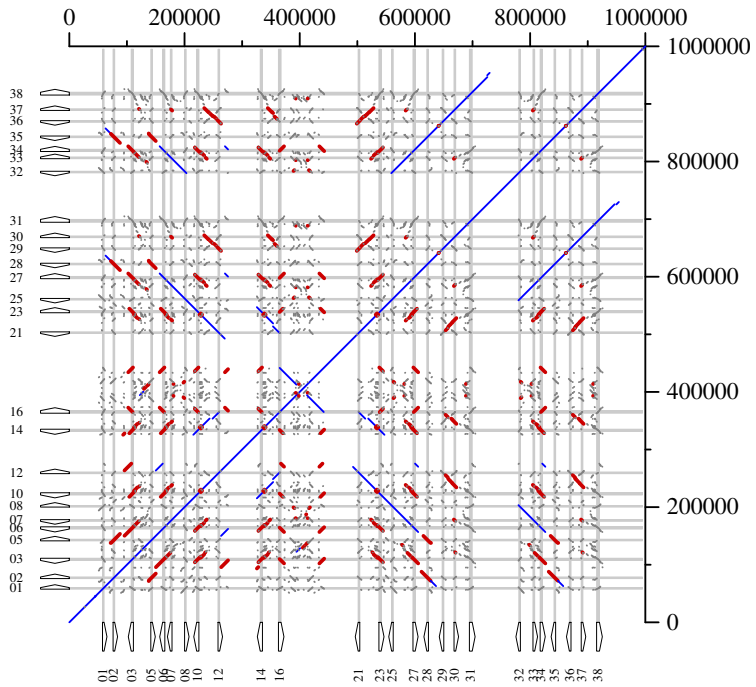
V genóme organizmov sa nachádzajú takzvané zhluky génov[1] . Sú to miesta, kde došlo ku skopírovaniu úseku DNA a duplikácii génov. Samotné kopírovanie je chybou pri niektorých biologických procesoch. V prípade, že organizmus s touto chybou zostane životaschopný, môže sa chyba zachovať. Zhluky génov obsahujú viacero kópií génu, pričom tieto kópie nemusia byť totožné, lebo v priebehu evolúcie nastávajú mutácie jednotlivých génov oddelene. Zhluky génov tak obsahujú gény kódujúce podobné – homologické proteíny s podobnou funkciou a štruktúrou. Taktiež môžu obsahovať pseudo-gény, teda už biologicky nefunkčné gény.

Zhluky génov tvoria približne 5% ľudskej genetickej informácie. Zaujímavé je, že sa objavujú častejšie u vývojovo vyšších organizmov. Tieto zhluky sa tiež používajú pri výskume, napríklad evolúcie príbuzných druhov.

2.2 Aplikácia programu Exonerate na zhluky génov

Ak chceme použiť program Exonerate na zarovnávanie sekvencií ku anotácii zhluku génov, zarovnáme viaceré homologické proteíny ku vstupnej sekvencii. Avšak výstupné zarovnania obsahujú viaceré nadbytočné zarovnania. Ďalším problémom je, že pre získanie zarovnania všetkých podobných génov musíme znížiť citlivosť programu Exonerate, čo má za násle-

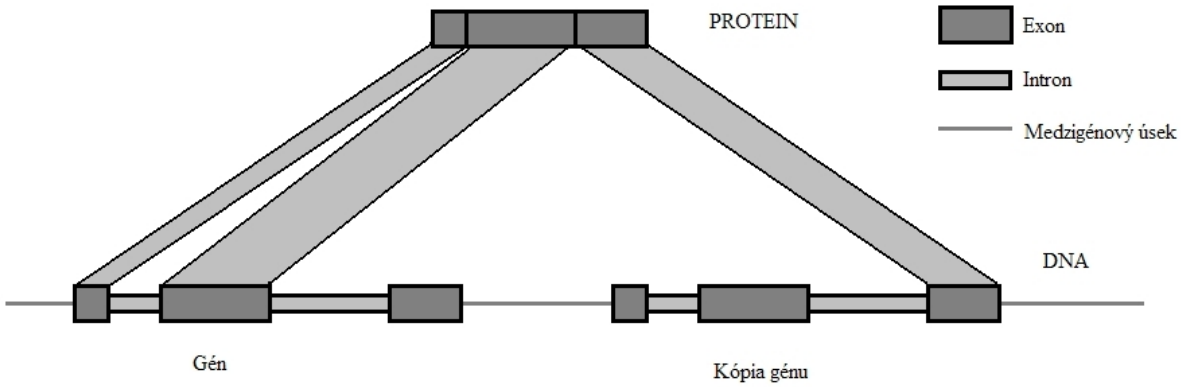
[1] tiež génové klastre (z anglického gene clusters)



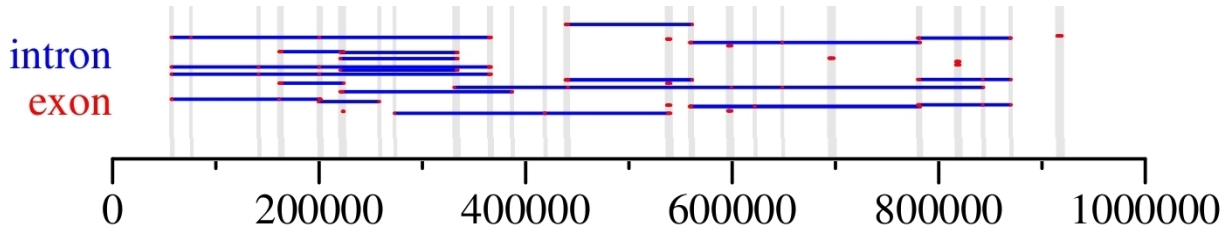
Obrázok 2.1: Na obrázku je zhluk génov PRAME. Modrá farba značí zhodu sekvencií väčšiu ako 98%. Červená farba zhodu väčšiu ako 92%. Sivá zhodu väčšiu ako 73%.

dok vznik ďalších nadbytočných zarovnaní. Jednu z možných príčin vzniku nadbytočných zarovnaní popisuje obrázok 2.2.

Výstup z programu Exonerate obsahuje aj množstvo kópií exónov alebo aj prekrývajúce sa exóny, pričom tieto exóny pochádzajú zo zarovnaní iných proteínov k tej istej sekvencii.



Obrázok 2.2: Na tomto príklade vidíme jednu z možností vzniku nadbytočného zarovnania. V tomto prípade sa celý medzigénový úsek spolu s časťami génu interpretuje ako intrón.



Obrázok 2.3: Na obrázku je ľudský zhluk génov PRAME zarovnaný s proteínmi pomocou programu Exonerate. Vidíme množstvo evidentne nesprávnych zarovnaní. Obrázok bol vygenerovaný programom Mikroskop[1]

Kapitola 3

Reprezentácia anotácie pomocou grafu

V predchádzajúcej kapitole sme videli, že aplikácia programu Exonerate na sekvencie obsahujúce zhluky génov nemá dobré výsledky. V tejto kapitole popíšeme reprezentáciu množiny zarovnaní pomocou grafu, ktorá nám umožní definovať úlohu anotácie zhluku génov, ako úlohu na tomto grafe. Táto reprezentácia sa podobá na takzvané splicing grafy[5]. Exóny budú v našom grafe reprezentované vrcholmi, a intróny hranami orientovanými v smere zarovnania.

Každý exón je reprezentovaný štruktúrou

$$e = (target, from, to)$$

kde *target* je DNA sekvencia z ktorej pochádza, *from* a *to* sú súradnice začiatku a konca exónu. Avšak pôvodná množina exónov K' obsahuje exóny, ktoré sa prekrývajú. Preto ako prvé rozdelíme jednotlivé exóny na časti tak, aby nová množina exónov K obsahujúca tieto časti, neobsahovala čiastočne prekrývajúce sa exóny, aby sa každý exón z množiny K' dal zložiť z niekoľkých exónov z možiny K a aby množina K bola minimálna. Pri operácii rozdelenia exónu vytvoríme ε-novú hranu, ktorá novovytvorené exóny spojí. Množinu ε-nových hrán označíme N_ε.

Taktiež zadefinujeme podmnožinu $K_s \subseteq K$ exónov kódujúcich začiatok proteínu a podmnožinu $K_t \subseteq K$ exónov kódujúcich koniec proteínu.

Intrón reprezentujeme pomocou štruktúry

$$i = (target, from, to, e_1, e_2)$$

kde *target*, *from* , *to* sú zadefinované rovnako, ako pri exóne, a $e_1, e_2 \in K$ sú exóny, ktoré spája daný intrón v pôvodnom zarovnaní. Množinu všetkých intrónov nazveme N.

Teda môžeme zadefinovať ohodnotený orientovaný graf $G = (V, E, h)$, kde $V = K \cup \{s, t\}$

$$E = \{(e_1, e_2) \in V \times V | \exists t, fr, to : (t, fr, to, e_1, e_2) \in N\} \cup N_\varepsilon \cup (\{s\} \times K_s) \cup (K_t \times \{t\})$$

a $h : E \to \mathbb{R}_0^+$ je ohodnotenie hrán. V tomto grafe je hranou reprezentovaný každý intrón alebo ε-nova hrana, a navyše hrany vedú aj z počiatočného vrcholu s do všetkých exónov začínajúcich zarovnanie, a podobne zo všetkých exónov končiacich zarovnanie vedie hrana do koncového vrcholu t. Použité ohodnotenie hrán h sa skladá z čiastkových ohodnotení vrcholov(exónov) h_v a hrán(intrónov) h_e. Teda

$$(\forall i = (tar, f, t, e_1, e_2) \in N : h(e_1, e_2) = \alpha_i h_e(i) + \alpha_e h_v(e_2)$$

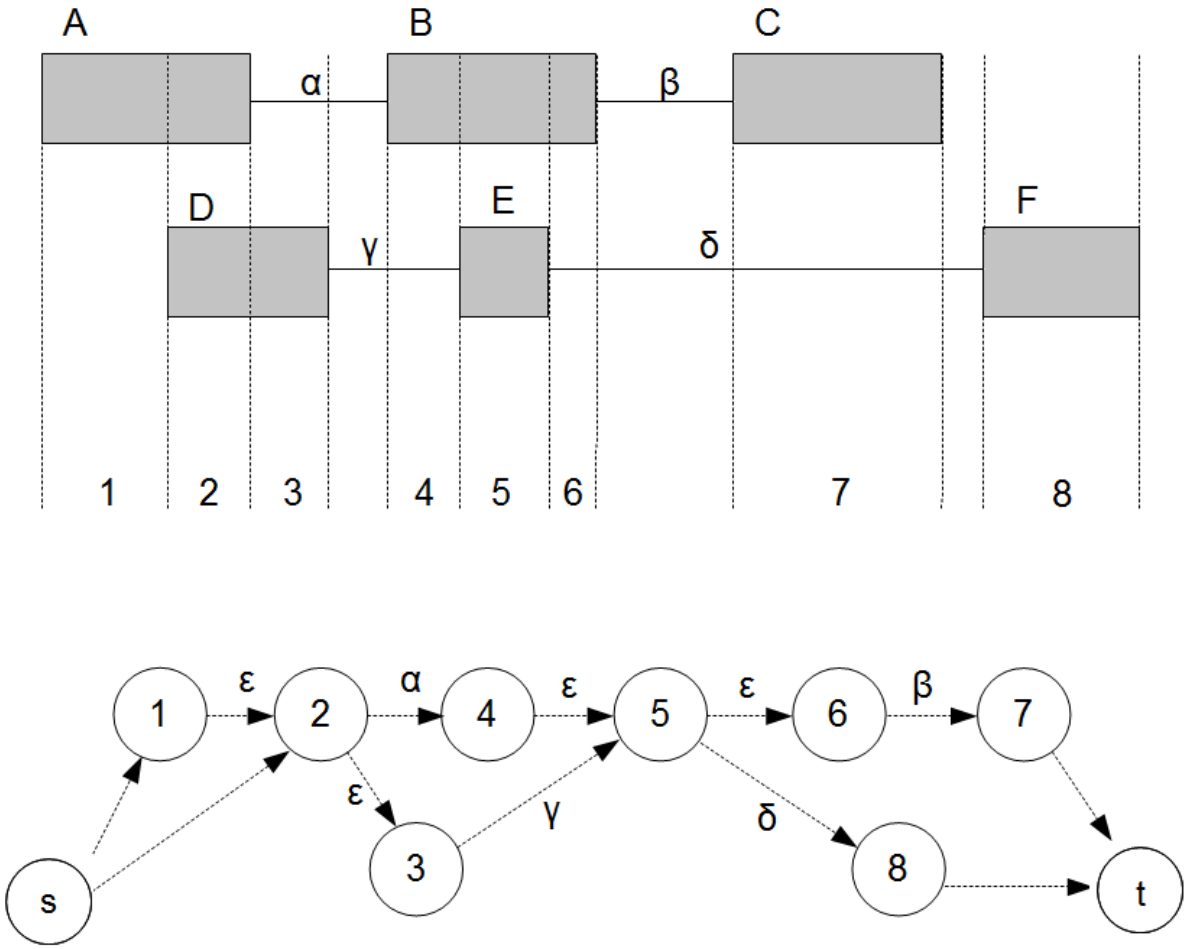
$$\forall (e_1, e_2) \in N_\varepsilon : h(e_1, e_2) = \alpha_e h_v(e_2)$$

$$\forall e \in N : h(s, e) = \alpha_e hv(e) \wedge h(e, t) = 0$$

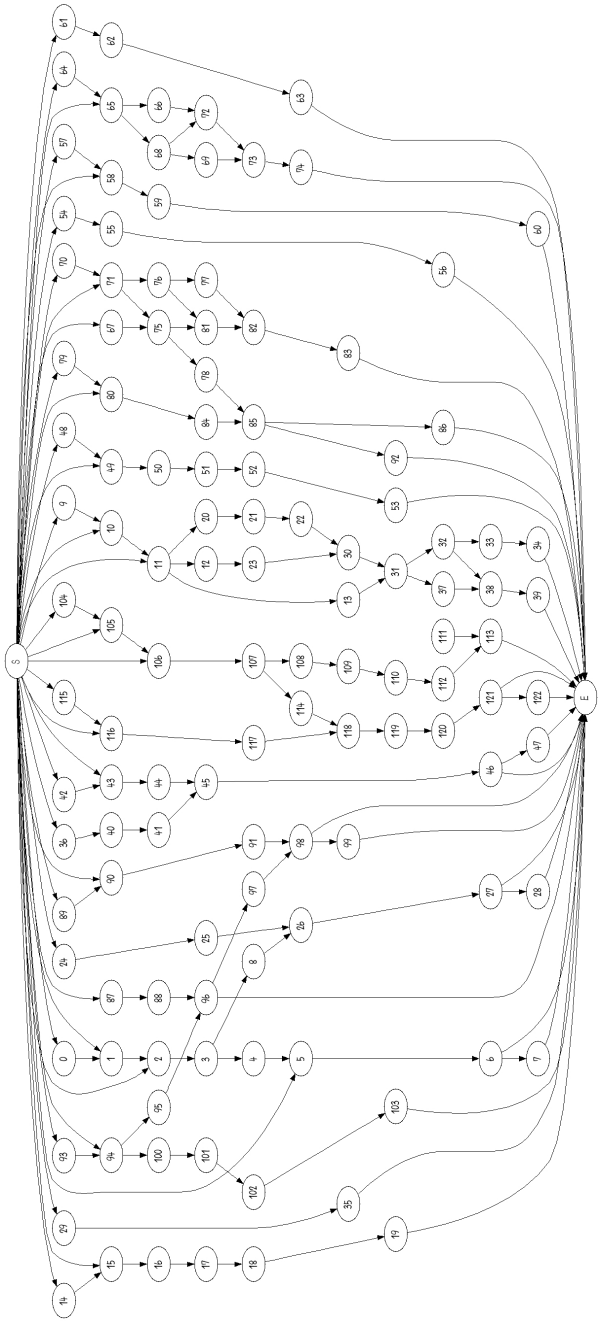
kde α_e, α_i sú váhovacie koeficienty. Docielili sme, že cena každej s-t cesty je lineárnou kombináciou cien hrán a vrcholov na nej ležiacich.

Všimnime si, že každému vstupnému zarovnaniu zodpovedá v našom grafe jedna s-t cesta, avšak graf obsahuje aj ďalšie cesty. ktoré vznikli ako kombinácia vstupných zarovnaní. Obrázok 3.1 popisuje na jednoduchom príklade tvorbu grafu z množiny zarovnaní. Obrázok 3.2 ukazuje výsledný graf zodpovedajúci zhluku génov PRAME.

V ďalšej kapitole sa budeme zaoberať tým, ako tento graf použijeme na anotáciu zhlukov génov.



Obrázok 3.1: Na obrázku je jednoduchý príklad tvorby grafu z množiny zarovnaní. Pôvodné exóny (prvky množiny N') sú označené písmenami A až F. Číslami 1 až 8 označíme neprekrývajúce sa časti exónov (prvky množiny N'). Písmenami α až δ označíme intróny. V grafe spojíme hranou tie časti exónov, ktoré boli spojené intrónom, alebo patrili jednému exónu. Na začiatok a koniec oboch zarovnaní pridáme ďalší vrchol.



Obrázok 3.2: Na obrázku je grafová reprezentácia zhluku génov Prame. Obrázok bol vygenerovaný programom Graphviz[3].

Kapitola 4

Anotácia pomocou najkratších ciest

4.1 Formulácia problému

Predtým, ako formálne definujeme problém najlepšej anotácie, definujeme najskôr neprekrývajúce sa s-t cesty v grafe G. Dve s-t cesty u a v sú v grafe G *navzájom neprekrývajúce sa*, ak majú rôzny parameter *target*, alebo ak v jednej z ciest je parameter *to* exónu zodpovedajúci vrcholu pred vrcholom t menší ako parameter *from* exónu zodpovedajúcemu prvému vrcholu po vrchole s. Ako reprezentáciu výslednej anotácie hľadáme graf H, ktorý spĺňa nasledovné podmienky:

1. H obsahuje neprekrývajúce sa s-t cesty grafu G, ktoré majú spoločné vrcholy iba s a t,

2. po pridaní ľubovoľnej s-t cesty grafu G je porušená podmienka (1),

3. H má minimálny súčet ohodnotenia všetkých svojich hrán.

Aby nami nájdený graf zodpovedal biologickým očakávaniam, je potrebné zvoliť ohodnotenia hrán a vrcholov. Ohodnotenie hrán h_e sme definovali ako dĺžku intrónu, a ohodnotenie vrcholov h_v ako pomerné skóre pripadajúce na daný exón získané z programu Exonerate. Tieto ohodnotenia sa dajú váhovať pomocou váhovacích konštánt α_e a α_i. Keďže lepšie zarovnania majú v programe Exonerate väčšie skóre váhovací koeficient pre vrcholy α_e by mal byť ≤ 0. Váhovací koeficient pre hrany α_i by mal byť > 0, aby sme uprednostňovali kratšie

zarovnania. Používali sme aj alternatívne ohodnotenie hrán definované ako logaritmus dĺžky intrónu, ktoré nám umožní ľahšie identifikovať príliš dlhé intróny.

4.2 Najlacnejšia cesta v grafe

V priebehu riešenia problému budeme riešiť čiastkovú úlohu, ktorou je nájdenie najlacnejšej cesty v ohodnotenom acyklicky orientovanom grafe. Táto úloha sa dá riešiť v čase $\Theta(|V| + |E|)$ [2]. V takomto grafe existuje topologické usporiadanie, teda usporiadanie vrcholov grafu tak, aby hrany viedli iba jedným smerom. V našom prípade nemusíme toto usporiadanie vytvárať algoritmicky, pretože máme k dispozícii prirodzené usporiadanie v závislosti od súradníc a orientácie exónov.

Algoritmus prechádza v jednom smere poľom A, v ktorom hodnota $A[v]$ pre vrchol $v \in V$ predstavuje hodnotu zatiaľ najlacnejšej nájdenej cesty z vrcholu v_0 do vrcholu v. V ďalšom poli $B[v]$ si pamätáme, z ktorého vrcholu vedie najlacnejšia cesta do vrcholu v. Pri reprezentácii grafu pomocou množiny susedov, prechádza každým vrcholom a každou hranou maximálne raz.

Pseudokód použitého algoritmu:

(1) $\forall v \in V : A[v] \leftarrow \infty$
(2) $A[v_0] \leftarrow 0$
(3) **for** $\forall$ vrcholi v v topologickom usporiadaní **do**
(4) **for** $\forall$ hrany (v, u) vedúce z vrcholu v **do**
(5) **if** $A[v] + h(v, u) < A[u]$ **then begin**
(6) $A[u] \leftarrow A[v] + h(v, u)$
(7) $B[u] \leftarrow v$
(8) **end**

4.3 Riešenie problému pomocou greedy heuristiky

V rámci riešenia tejto práce sa nám nepodarilo nájsť efektívny exaktný algoritmus na riešenie problému nájdenia grafu H s požadovanými vlastnosťami. Rozhodli sme sa teda riešiť úlohu pomocou greedy heuristiky, ktorá síce negarantuje splnenie bodu (3), ale rieši problém v polynomiálnej časovej a priestorovej zložitosti.

Samotný greedy algoritmus spočíva v opakovanom hľadaní najlacnejšej s-t cesty v grafe G, po nájdení cesty túto cestu z grafu G odoberieme (vrcholy s a t ponecháme) a pridáme cestu do výsledného grafu H. Tým zabránime viacnásobnému použitiu jedného vrcholu (exónu). Taktiež z pôvodného grafu odoberieme vrcholy a hrany, ktoré sa s vybranou s-t cestou prekrývajú. Týmto sme zabezpečili platnosť podmienky (1). V prípade, že už žiadne s-t cesty v G neexistujú, algoritmus skončí a H obsahuje niekoľko neprekrývajúcich sa disjunktných ciest spojených vrcholmi s a t.

Výsledný algoritmus sme implementovali v programovacom jazyku C. V prvej fáze prevedieme vstupnú množinu zarovnaní uloženú v GTF 2.2 súbore, ktorú vytvoril program Exonerate, do jej grafovej reprezentácie. V druhej fáze aplikujeme na tento graf náš heuristický algoritmus.

V ďalšej kapitole ukážeme aplikáciu nášho algoritmu na reálne dáta.

Kapitola 5

Aplikácia na skutočné dáta

V tejto kapitole ukážeme aplikáciu nášho algoritmu na reálne dáta, konkrétne na zhluky génov AMY a PRAME v genómoch človeka a mamaka. Algoritmus sme najskôr aplikovali s váhovacími konštantami $\alpha_e = -1$ (váha skóre exónu) $\alpha_i = 1$ (váha skóre intrónu). Na zhluku génov PRAME v ľudskom genóme tiež ukážeme, ako sa menili výsledky so zmenou pomerov váh intrónov a exónov.

V prípade zhluku génov PRAME sme použili DNA sekvenciu z ľudského genómu o dĺžke 1020015 báz a sekvenciu z genómu makaka o dĺžke 204000 báz. Ako proteíny sme požili 4 gény z ľudského genómu tak, ako sú oanotované v databáze UCSC genóme browser [4]. Pri zhluku génov AMY sme použili ľudskú DNA sekvenciu o dĺžke 220643 báz a sekvenciu z genómu makaka o dĺžke 170013 báz. Používali sme 3 rôzne ľudské proteíny, ktoré sa nachádzajú v UCSC genóme browser.

Počty exónov a zarovnaní pred a po aplikácii algoritmu popisuje tabuľka 5.1. Graficky znázornené vstupy a výstupy nájdeme na obrázkoch 5.1, 5.2, 5.3 a 5.4. Obrázky boli vytvorené programom Mikroskop [1]. Pre každý zhluk a každú DNA sekvenciu máme jeden obrázok. Z obrázkov je zrejmé, že boli vybrané vždy najkratšie zarovnania. Väčšina výstupných zarovnaní je zhodných s niektorým zo vstupných zarovnaní, no nájdu sa aj také zarovnania, ktoré v pôvodnej množine zarovnaní neboli. Jeden z takýchto prípadov sa nachádza na obrázku 5.3. Prvé výstupné zarovnanie sa nenachádza medzi vstupnými zarovnaniami. Jedno z vstupných zarovnaní zdieľa s týmto novým zarovnaním prvý a posledný exón, avšak nové zarovnanie používa aj exóny z iného zarovnania, teda dochádza ku kombinácii viacerých

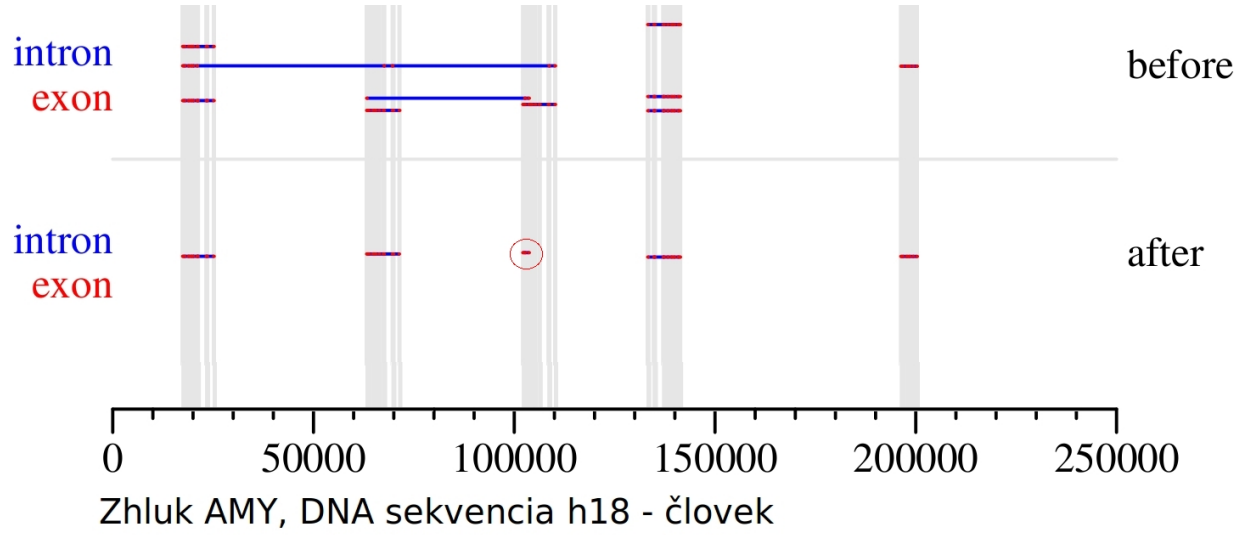
Tabuľka 5.1: Tabuľka sumarizuje výsledky nášho algoritmu pre váhovacie konštanty $\alpha_e = -1$ a $\alpha_i = 1$. Menší počet odstránených zarovnaní v zhluku AMY spôsobilo, že neobsahuje toľko prekrývajúcich sa zarovnaní ako PRAME.

Zhluk génov	počet exónov pred	#exónov po	#zarovnaní pred	# zarovnaní po
PRAME človek	84	33	32	7
PRAME mamak	39	26	16	5
PRAME (spolu)	123	59	48	12
AMY človek	48	41	10	5
AMY makak	30	20	7	3
AMY (spolu)	78	61	17	8

zarovnaní. V tomto prípade je možné, že ide o gén, ktorý vznikol kombináciou viacerých neúplných kópií. Ďalší zaujímavý prípad vidíme na obrázku 5.1. Zaujíma nás konkrétne tretie výsledné zarovnanie, kde došlo ku kombinácii dvoch rôznych zarovnaní. V tomto prípade nejde pravdepodobne o kompletný funkčný proteín, ale asi došlo ku pseudogenizácii daného génu, ktorú mohla spôsobiť duplikácia na nevhodnom mieste. Obrázky 5.2 a 5.4 ukazujú anotácie, ktoré sú zaujímavé tým, že vznikli pomocou proteínov z iného organizmu (v oboch prípadoch ide o DNA sekvencie makaka a ľudské proteíny.)

Obrázok 5.5 a tabuľka 5.2 popisuje vplyv váhovacích konštánt α_e (váha skóre exónu) α_i (váha skóre intrónu) na výstup nášho algoritmu (výstupnú množinu zarovnaní). Vplyv váhovania sme testovali na ľudskom zhluku génov PRAME, keďže ide o sekvenciu s najzložitejšou štruktúrou, s ktorou sme pracovali.

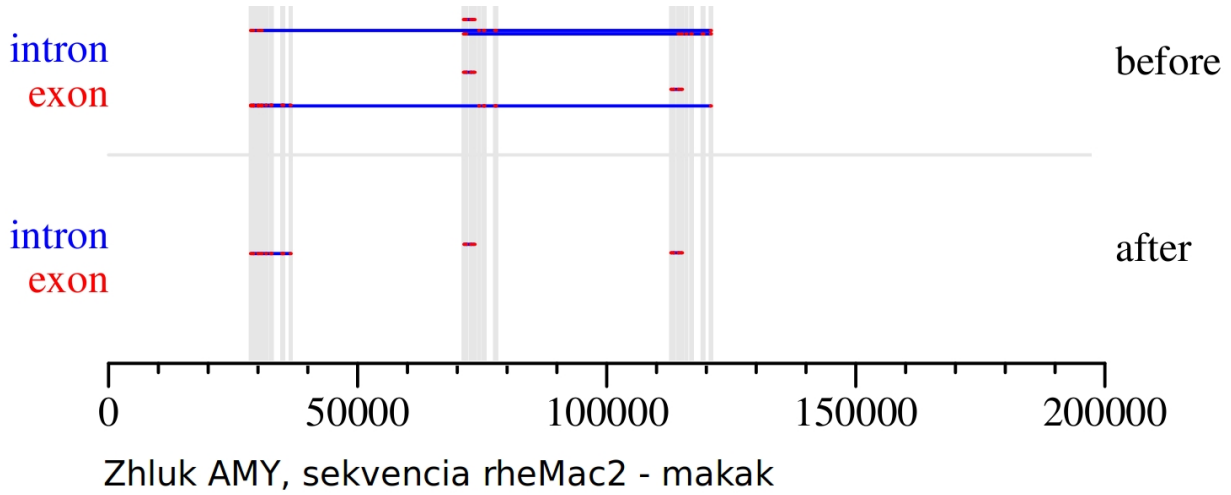
Pri použití skórovania, kde výsledné skóre exónu a intrónu bolo aspoň rádovo podobné, dochádzalo len k minimálnym zmenám a výstup algoritmu bol približne rovnaký ako na obrázku 5.3. Výraznejšie zmeny nastali až v prípade, že výsledné skóre intrónu bolo extrémne podhodnotené. Pri testovaní vplyvu váhovania sme našli dve rôzne zarovnania na ktoré mala zmena váhovania vplyv. Prvé zarovnanie má tri rôzne varianty A, B, C. Ďalším meniacim sa úsekom bolo druhé zarovnanie, ktoré má dve varianty 0 a 2. V prípade extrémne podhodnoteného skóre intrónu dochádzalo k väčším zmenám (množiny zarovnaní D, E), tieto nás však nebudú zaujímať. Výsledné množiny zarovnaní sú pre rôzne váhovania graficky znázornené na obrázku 5.5.



Obrázok 5.1: Na obrázku je výstup nášho programu pre zhluk AMY na DNA sekvencii hg18, ktorá pochádza od človeka. Na obrázku každá čiara reprezentuje zarovnanie proteínu k DNA sekvencii. $\alpha_e = -1$ a $\alpha_i = 1$

Varianty prvého zarovnania A, B, C menia iba vnútornú štruktúru zarovnania, začiatočný a koncový exón zostáva nezmenený. Výsledné zarovnanie je pospájané z viacerých častí, pričom každá časť patrí inému zarovnaniu. Preto, ako sa mení váhovanie, tak sa mení skóre pre jednotlivé časti, ktoré sa iným spôsobom skombinujú do výsledného zarovnania. Všetky tri varianty A, B, C sú pre nás vyhovujúce a nevieme z dostupných informácií zistiť, ktorý je biologicky správnejší.

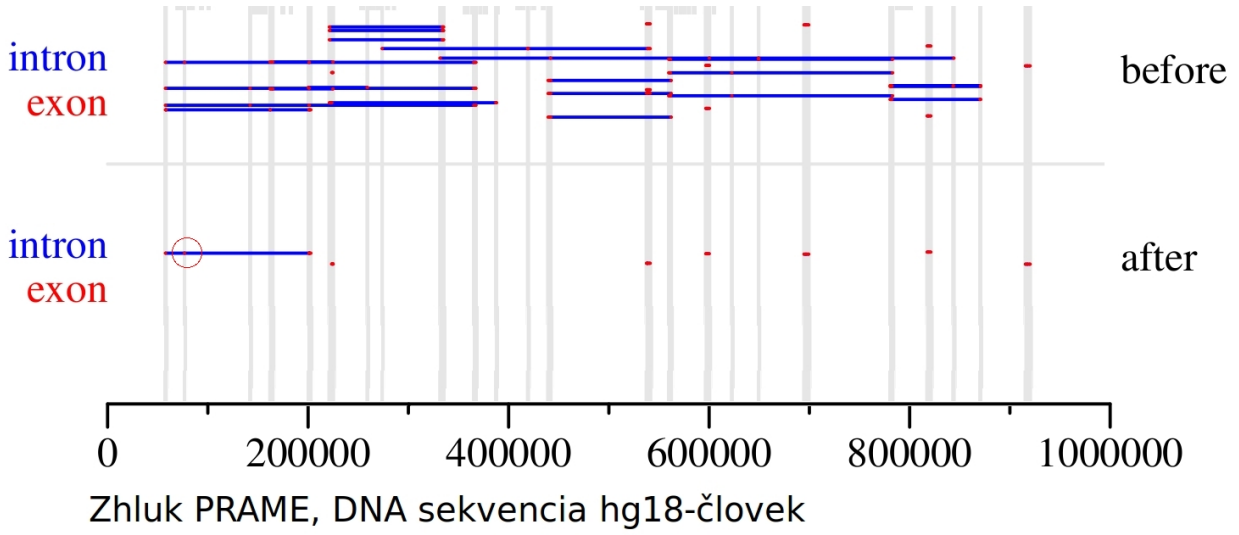
V prípade druhého zarovnania nastáva variant 0 alebo 2. Variant druhého výstupného zarovnania 0 zodpovedá jednému zo vstupných zarovnaní. Jeho variant 2 nastáva pri nižších váhach intrónu, keď sa variant 0 rozšíri o ďalšie dva exóny. Toto je spôsobené prekrytím koncového exónu variantu 0 s exónom iného zarovnania X, variant 2 potom obsahuje celý variant 0 a navyše aj časť tohto zarovnania X. Pri vyšších váhach intrónu nie je výhodné rozširovať variant 0, preto sa tu variant 2 nevyskytuje.



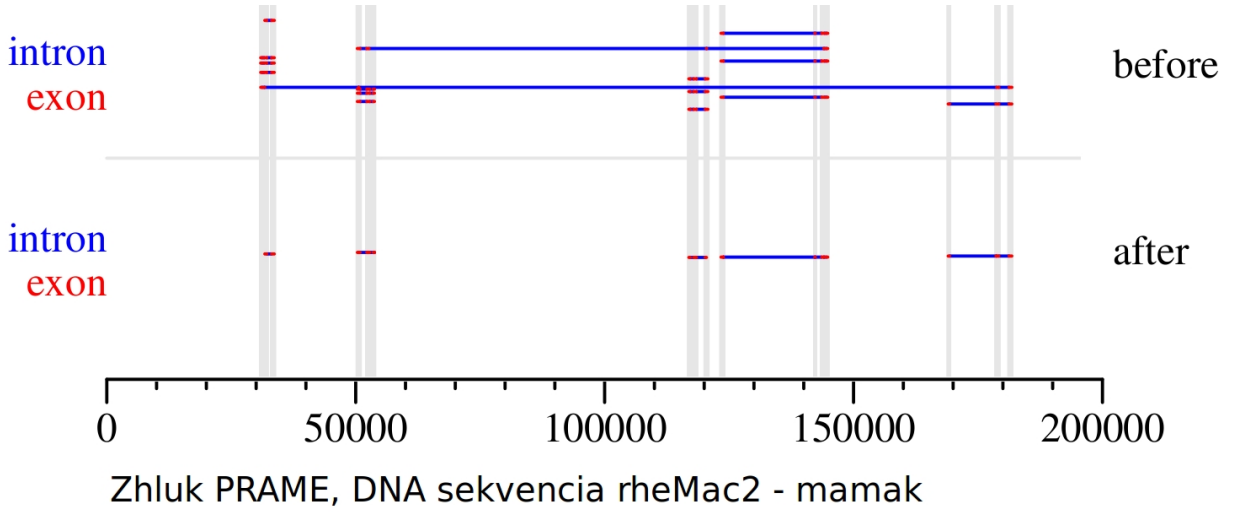
Obrázok 5.2: Na obrázku je výstup nášho programu pre zhluk AMY na DNA sekvencii rheMac2, ktorá pochádza od makaka. Na obrázku každá čiara reprezentuje zarovnanie proteínu k DNA sekvencii. $\alpha_e = -1$ a $\alpha_i = 1$

Tabuľka 5.2: Tabuĺka popisuje, ktoré množiny zarovnaní sme vytvorili pomocou nášho algoritmu s použitím rôznych váhovacích konštánt α_e, α_i. Množiny zarovnaní sú popísané na obrázku 5.5.

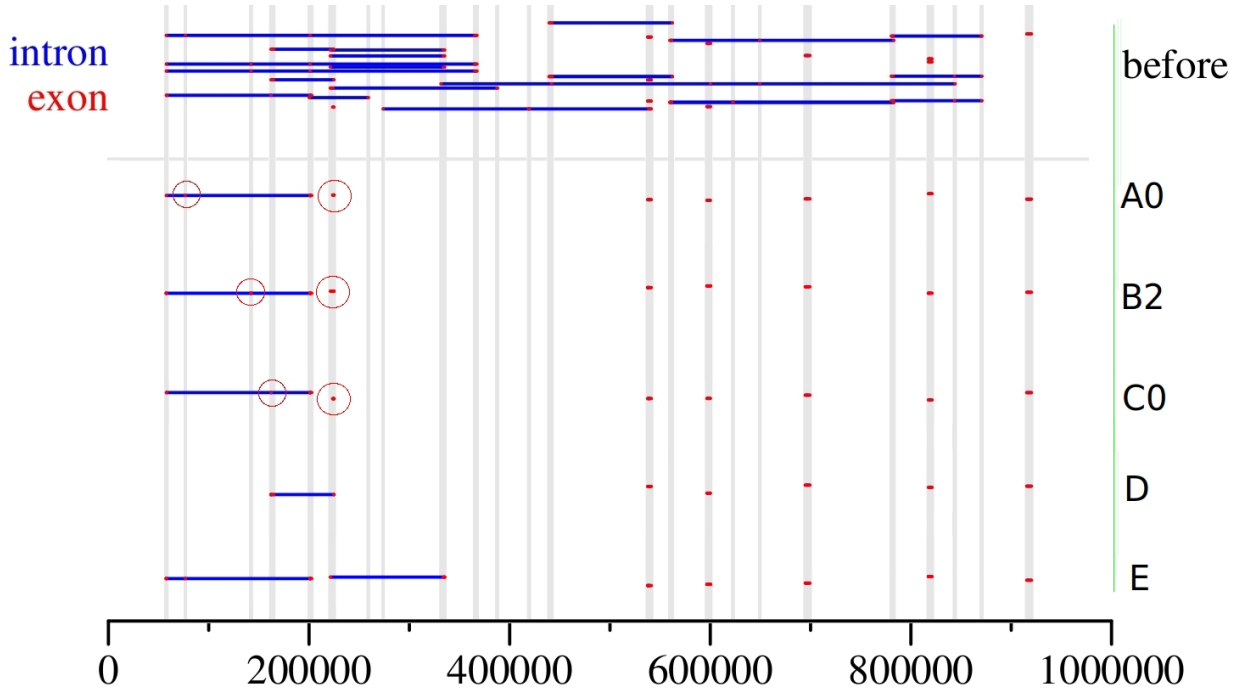
α_e	α_i	Množina zarovnaní
-1	4.4 až 10^6	C0
-1	4.3 až 0.4	A0
-1	0.3 až 0.02	A2
-1	0.01 až 0.001	B2
-1	0.0009 až 0.0004	A2
-1	0.0003 až 10^{-9}	D
-1	0	E



Obrázok 5.3: Na obrázku je výstup nášho programu pre zhluk PRAME na DNA sekvencii hg18, ktorá pochádza od človeka. Na obrázku každá čiara reprezentuje zarovnanie proteínu k DNA sekvencii. $\alpha_e = -1$ a $\alpha_i = 1$



Obrázok 5.4: Na obrázku je výstup nášho programu pre zhluk PRAME na DNA sekvencii rheMac2, ktorá pochádza od makaka. Na obrázku každá čiara reprezentuje zarovnanie proteínu k DNA sekvenciíi. $\alpha_e = -1$ a $\alpha_i = 1$



Obrázok 5.5: Na obrázku sú výstupné množiny zarovnaní pre rôzne váhovania. Názvy množín zodpovedajú názvom v tabuľke 5.2. Pri príliš malej váhovacej konštante intrónu α_i v prípade množín B a D, vidíme že boli uprednostnené dlhé zarovnania. Zaujímavé je rozdielne použitie exónov a intrónov v prvom zarovnaní z množín A a A'

Záver

Cieľom tejto práce bolo anotovať sekvenciu predstavujúcu zhluk génov pomocou zarovnania so známymi proteínmi. Na tento účel sme použili program Exonerate, ktorý však na týchto miestach produkuje množstvo nadbytočných transkriptov.

Množinu zarovnaní získanú z programu Exonerate sme reprezentovali pomocou špeciálne zostaveného grafu, na ktorom sme riešili úlohu získania anotácie heuristickým algoritmom. Po aplikácii nášho algoritmu sme získali sadu neprekrývajúcich sa transkriptov, ktorá vyhovuje našim očakávaniam. Vo väčšine prípadov nami nájdené transkripty boli prítomné aj vo výstupe z programu Exonerate, avšak v niektorých prípadoch sme vytvorili aj nový transkript, ktorý je kombináciou viacerých vstupných zarovnaní. Na ľudskej sekvencii zhluku PRAME sme ukázali ako použitie rôznych vstupných parametrov, vplýva na kvalitu výstupu získaného našim programom.

Opravili sme výstup z programu Exonerate aplikovaného na zhluky génov, čím sme tieto sekvencie úspešne oanotovali. Očakávame, že sa náš algoritmus bude možné použiť aj v praxi na automatizáciu anotácie génových zhlukov, ktorá v súčasnosti prebieha obvykle manuálne.

Literatúra

[1] B. Brejova, T. Vinar, Y. Chen, S. Wang, G. Zhao, D. G. Brown, M. Li, and Y. Zhou. Finding genes in Schistosoma japonicum: annotating novel genomes with help of extrinsic evidence. *Nucleic Acids Res*, 37(7):e52, 2009.

[2] T. H. Cormen, C. E. Leiserson, and R. L. Rivers. *Introduction to Algorithms*. The MIT Press, 1990.

[3] J. Ellson, E. Gansner, L. Koutsofios, and G. North, S.and Woodhull. Graphviz—open source graph drawing tools. In *Graph Drawing*, pages 594–597. Springer, 2002.

[4] P. A. Fujita, B. Rhead, A. S. Zweig, A. S. Hinrichs, D. Karolchik, M. S. Cline, M. Goldman, G. P. Barber, H. Clawson, A. Coelho, M. Diekhans, T. R. Dreszer, B. M. Giardine, R. A. Harte, J. Hillman-Jackson, F. Hsu, V. Kirkup, R. M. Kuhn, K. Learned, C. H. Li, L. R. Meyer, A. Pohl, B. J. Raney, K. R. Rosenbloom, K. E. Smith, D. Haussler, and W. J. Kent. The ucsc genome browser database: update 2011. *Nucleic Acids Res*, 39(Database issue):D876–882, 2011.

[5] S. Heber, M. Alekseyev, S. H. Sze, H. Tang, and P. A. Pevzner. Splicing graphs and EST assembly problem. *Bioinformatics*, 18 Suppl 1:S181–188, 2002.

[6] G. Slater and E. Birney. Automated generation of heuristics for biological sequence comparison. *BMC Bioinformatics*, 6(1):31, 2005.

[7] M. Zvelebil and J. O. Baum. *Understanding Bioinformatics*. Garland Science, Aug. 2007.