# Implementácia externých zdrojov dát v hľadaní génov

Bakalárska práca

Peter Kováč

**Univerzita Komenského**

**Fakulta Matematiky, Fyziky a Informatiky**

**Katedra Informatiky**

9.2.1 Informatika

*Vedúca bakalárskej práce:* Mgr. Bronislava Brejová, PhD.

Bratislava, 2010

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

---

Essentially, all models are
wrong, but some are useful.

George Box

Ďakujem mojej vedúcej, Mgr. Bronislave Brejovej, PhD., za cenné rady a skvelý prístup a mojej rodine za sústavnú podporu.

# Abstrakt

Hľadanie (predikcia) génov v DNA sekvenciách je jedným zo základných problémov bioinformatiky. Problém sa často rieši Viterbiho algoritmom na skrytých Markovových modeloch (HMM). Pri hľadaní génov sa využívajú aj externé dáta - EST sekvencie, proteínové sekvencie, zarovnania viacerých DNA sekvencií a iné.

Program ExonHunter okrem HMM obsahuje aj mechanizmus na spracovanie externých dát. Pôvodné implementácie zahrnutia EST a proteínových sekvencií do predikcie nevyhovovali kvôli dlhému času spracovania. Programy zodpovedajúce za spracovanie sme preto vymenili za potenciálne rýchlejšie.

Upravený program sme testovali na DNA sekvenciách mušky *Drosophila melanogaster*. V prípade EST sekvencií sa nám podarilo dosiahnuť zlepšenie času za cenu mierneho zhoršenia presnosti predikcie. V prípade proteínových sekvencií sme zlepšili aj čas aj presnosť. Porovnanie s inými programami ukázalo, že ExonHunter je dobrý hlavne pri predikciách s využitím externých informácií.

**Kľúčové slová:** gén, hľadanie génov, skrytý Markovov model, ExonHunter.

# Predhovor

V posledných rokoch sme svedkami ohromného rozmachu moderných biologických odvetví – genetiky a molekulárnej biológie. Obe disciplíny hľadajú odpovede na široké spektrum otázok, týkajúcich sa zložitých metabolických procesov v živých organizmoch, pôvodu života a evolúcie, umelých genetických zmien v organizmoch alebo medicínskeho výskumu dedičných chorôb.

Pri hľadaní odpovedí na tieto zložité otázky musia biológovia najprv zodpovedať aj jednu úplne základnú: *kde sú v DNA ukryté gény a akú majú funkciu?* Veľkým pomocníkom pri jej zodpovedaní je, možno trochu nečakane, informatika.

Zostavovanie ľudského genómu (kompletnej genetickej informácie človeka) trvalo pätnásť rokov a stálo tri miliardy amerických dolárov. Výsledným produktom je zdigitalizovaná DNA – niekoľko slov s celkovou dĺžkou tri miliardy písmen. Odvtedy sa technológie výrazne zlepšili a dnes je možné získať rovnaké dáta za zlomok času a ceny. Samozrejme, biológovia sa neobmedzujú len na človeka a skúmajú všetko živé – od baktérií a vírusov, cez huby, rastliny, jednoduché bezstavovce, hmyz až po stavovce a cicavce. Laboratóriá po celom svete produkujú ohromné množstvo informácií. Experimentálne metódy analyzovania týchto dát sú však stále relatívne drahé, pomalé a prácne. Matematické modely a počítačové výpočty sú preto vítanou pomocou. Hoci nie sú vždy dokonale presné, pracujú rýchlo a tiež sa neustále zlepšujú.

V tejto práci sa zameriavame na informatickú stránku problému. Našim cieľom je zlepšiť existujúci program na hľadanie génov. Aby sme ako informatici mohli hľadať odpoveď na otázku, kde sú gény a akú majú funkciu, musíme sme okrem matematického a informatického aparátu porozumieť biologickým procesom, súvisiacim s génmi. Preto náš text začíname kapitolou venovanou biológii. V ďalšej kapitole vysvetľujeme základné matematické

modely a ich vzťah k biológii. Tiež tu opisujeme viaceré programy, ktoré sa súčasnosti používajú na hľadanie génov a zarovnanie sekvencií. Až potom sa venujeme vlastnej práci, z väčšej časti spočívajúcej v analýze výsledkov upraveného programu. Aby sme si vedeli urobiť predstavu o kvalite nášho programu, v ďalšej kapitole sme jeho výkon porovnávame s vybranými programami hľadajúcimi gény. V záverečnej kapitole sme naše úsilie sumarizujeme.

Snažili sme sa, aby táto práca neodradila zvedavých informatikov od biológie a aby jej výsledok bol zároveň prínosom pre biológov aktívne pracujúcich s informatickými nástrojmi.

# Obsah

# Zoznam obrázkov

# Zoznam tabuliek

# Kapitola 1

# Základné pojmy z biológie

V nasledujúcom texte uvedieme potrebné biologické pojmy. Čitateľ so základnými znalosťami z genetiky môže túto sekciu preskočiť. Čitateľovi s hlbším záujmom o molekulárnu biológiu a genetiku odporúčame modernú učebnicu, napríklad [Russell, 2010].

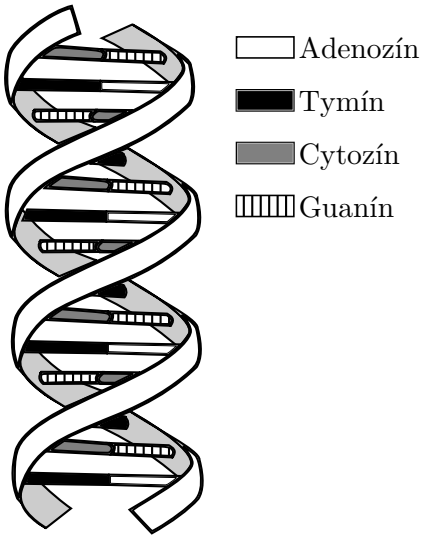
## 1.1 DNA a RNA

Hlavným objektom nášho záujmu je molekula deoxyribonukleovej kyseliny – *DNA*. DNA sa nachádza v bunkách živých organizmov a je nosičom genetickej informácie. Je tvorená dvomi polynukleotidovými vláknami stočenými do dvojzávitnice (obr. 1.1).

*Nukleotid* je molekula pozostávajúca z troch zložiek – fosfátovej skupiny, sacharidu (deoxyribózy) a dusíkatej bázy. V nukleotidoch molekuly DNA sa vyskytujú štyri rôzne dusíkaté bázy: *adenozín*, *cytozín*, *guanín* a *tymín*. Nukleotidy označujeme podľa začiatočného písmena bázy: *A, C, G, T*. Genetická informácia je v DNA zakódovaná v špecifickom poradí nukeotidov. Keďže nukleotidy sa odlišujú len bázami, v ďalších kapitolách nebudeme striktne rozlišovať pojmy nukleotid a báza.

Nukleotidové vlákna sú spojené vodíkovými väzbami medzi bázami, pričom sa spájajú adenozín s tymínom a cytozín s guanínom. Komplementárne bázy sú teda *A-T* a *C-G*. Konce nukleotidových vlákien sú rozdielne a podľa chemických zvyklostí sa označujú ako *5'* a *3'*. Vlákna sú k sebe opačne orientované. Ak teda poznáme poradie nukleotidov na jednom vlákne, poradie na opačnom vlákne vieme zistiť tak, že každú bázu preložíme do jej komple-

Obr. 1.1: Schéma DNA molekuly




mentu a celý reťazec obrátime.

Všetok genetický materiál v bunke sa nazýva *genóm*. V eukaryotických organizmoch (kam patria huby, rastliny, zvieratá, aj človek) je väčšina genómu uložená v bunkovom jadre v *chromozómoch*; menšie časti sa nachádzajú v mitochondriách a u rastlín aj v chloroplastoch. V nasledujúcich kapitolách budeme niekedy pojmom genóm označovať DNA sekvenciu, v ktorej hľadáme gény (hoci nemusí ísť nutne o celý genóm organizmu).

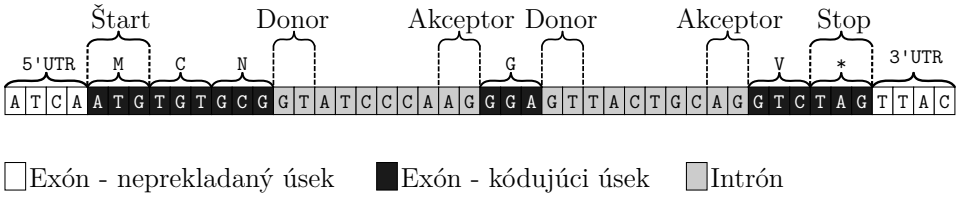
Ribonukleová kyselina – *RNA* – sa podobá na DNA, má však zvyčajne len jedno vlákno. Nukleotidy v RNA obsahujú namiesto deoxyribózy ribózu a namiesto tymínu uracil (U). RNA je rovnako ako DNA nosičom genetickej informácie (v prípade mnohých vírusov dokonca jediným). V rôznych formách sa zúčastňuje tvorby proteínov.

## 1.2 Proteíny

*Proteíny* (bielkoviny) majú v bunke rôzne funkcie. Slúžia ako stavebné látky, hrajú dôležitú úlohu pri metabolizme (enzýmy), môžu regulovať tvorbu ďalších proteínov a niektoré napríklad udržujú vnútrobunkové prostredie ako súčasť bunkovej membrány.

Stavebnými kameňmi proteínov sú *aminokyseliny*. Rozlišujeme 20 ami-

Obr. 1.2: Zjednodušený príklad génu




nokyselín, každá má podľa zvyklosti priradené jedno písmeno abecedy (tab. 1.1). Aminokyseliny sa spájajú sa do rôzne dlhých reťazcov, ktoré sa vplyvom chemických väzieb tvarujú do zložitých štruktúr. Funkcia proteínu často výrazne závisí práve od jeho tvaru.

## 1.3 Gény

Ďalším dôležitým biologickým pojmom je *gén*. Gén je určitý úsek DNA, podľa ktorého sa zložitým molekulárnym aparátom tvorí jeden alebo viacero proteínov (obr. 1.2).

Gény pozostávajú zo striedajúcich sa exónov a intrónov, pričom intrón je vždy medzi dvomi exónmi. Exóny sú úseky kódujúce reťazce aminokyselín. Začiatok prvého a koniec posledného exónu tvoria tzv. *neprekladané úseky* (5'UTR a 3'UTR). Neprekladané úseky obsahujú miesta, na ktoré sa viažu regulačné molekuly a neslúžia na kódovanie aminokyselín. Niekedy môžu obsahovať intróny. Kódujúce časti exónov sa skladajú z trojíc nukleotidov - *kodónov*. Na začiatku kódujúcej časti prvého exónu je *štart kodón* (ATG), na konci kódujúcej časti posledného exónu je *stop kodón* (TAA, TAG alebo TGA). Každý kodón (okrem stop kodónu) kóduje jednu aminokyselinu (tab. 1.1). Tento genetický kód je redundantný, keďže aminokyselín je 20 a rôznych trojíc nukleotidov 64. V určitých prípadoch teda nemusia mutácie v DNA znamenať žiadnu zmenu v kódovanej aminokyseline. Pri hľadaní génov nás najviac zaujímajú kódujúce časti exónov, v ďalších kapitolách preto budeme pre jednoduchosť za exóny považovať len kódujúce časti skutočných exónov a neprekladané úseky budeme označovať samostatne.

Intróny sa počas expresie génu vystrihnú a ďalej neovplyvňujú poradie aminokyselín vo vznikajúcom proteíne. Svoju úlohu hrajú pri alternatívnom zostrihu, keď sa tie isté časti génu môžu v rôznych podmienkach interpretovať buď ako exón, alebo ako intrón.

Tabuľka 1.1: Tabuľka aminokyselín a kodónov

| Aminokyselina | Skratka | Kodóny |
|---|---|---|
| Alanín | A | GCA, GCC, GCG, GCT |
| Cysteín | C | TGC, TGT |
| Kys. asparágová | D | GAC, GAT |
| Kys. glutámová | E | GAA, GAG |
| Fenylalanín | F | TTC, TTT |
| Glycín | G | GGA, GGC, GGG, GGT |
| Hisztidín | H | CAC, CAT |
| Izoleucín | I | ATA, ATC, ATT |
| Lyzín | K | AAA, AAG |
| Leucín | L | CTA, CTC, CTG, CTT, TTA, TTG |
| Metionín | M | ATG (Štart kodón) |
| Asparagín | N | AAC, AAT |
| Prolín | P | CCA, CCC, CCG, CCT |
| Glutamín | Q | CAA, CAG |
| Arginín | R | CGA, CGC, CGG, CGT, AGA, AGG |
| Serín | S | TCA, TCC, TCG, TCT, AGT, AGC |
| Treonín | T | ACA, ACC, ACG, ACT |
| Valín | V | GTA, GTC, GTG, GTT |
| Tryptofán | W | TGG |
| Tyrozín | Y | TAC, TAT |
| | * | TAA, TAG, TGA (Stop kodóny) |

Špecifické miesta v génoch, ktoré sú rozpoznávané bunkovým aparátom, označujeme ako *signály*. Z nášho pohľadu sú najdôležitejšie signály zostrihu: začiatok intrónu – *donor* (zvyčajne dvojica nukleotidov GT) a koniec intrónu – *akceptor* (zvyčajne AG).

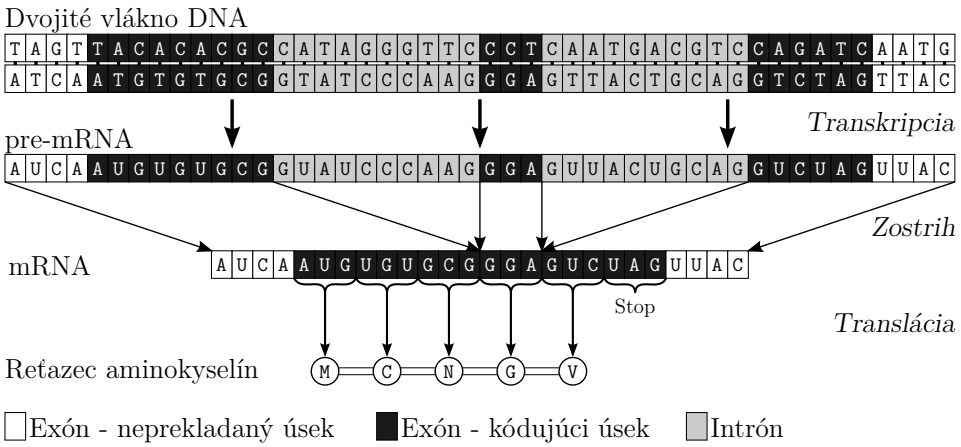
### 1.3.1 Expresia génov kódujúcich proteíny

Proces tvorby proteínov z génov kódujúcich proteíny má niekoľko fáz (obr. 1.3):

1. *Transkripcia.* Na špeciálne miesto v DNA molekule (promótor) sa naviaže enzým RNA polymeráza. Rozpletie dvojzávitnicu a v smere od 5' do 3' prekladá jedno vlákno do tzv. *pre-mRNA* molekuly. Na konci génu sa zastaví a odviaže sa od DNA.

2. *Zostrih.* Z pre-mRNA molekuly sa vystrihnú intróny a zvyšné exóny sa spoja do jednej makromolekuly *mRNA* (mediátorová RNA). Pri niektorých génoch je možný *alternatívny zostrih*. To znamená, že zostrih

Obr. 1.3: Expresia génu kódujúceho proteín




môže prebehnúť niekoľkými spôsobmi, takže podľa jedného génu sa môže vytvoriť viacero rôznych proteínov.

3. *Translácia.* Na štart kodón v mRNA sa naviaže ribozóm a s pomocou molekúl tRNA (transferová RNA) postupne prekladá trojice nukleotidov do reťazca aminokyselín. Preklad sa skončí na stop kodóne. Vzniknutý reťazec aminokyselín sa v niektorých prípadoch ešte môže upraviť a vytvorí finálny proteín.

# Kapitola 2

# Základné pojmy z bioinformatiky

V tejto kapitole uvedieme niektoré základné pojmy z bioinformatiky, týkajúce sa zarovnania sekvencií a hľadania génov. Pre hlbší pohľad do problematiky čitateľovi odporúčame učebnicu [Durbin et al., 1998].

Pod DNA sekvenciou v bioinformatike rozumieme slovo nad abecedou

$$\Sigma_D = \{A, C, G, T\}. \tag{2.1}$$

Podobne môžeme v zmysle predchádzajúcej kapitoly definovať RNA sekvenciu ako slovo nad abecedou

$$\Sigma_R = \{A, C, G, U\} \tag{2.2}$$

a proteínovú sekvenciu ako slovo nad abecedou

$$\Sigma_P = \{A, C, D, E, F, G, H, I, K, L, M, N, P, Q, R, S, T, V, W, Y\}. \tag{2.3}$$

## 2.1 Dáta v bioinformatike

Sekvencie používané v bioinformatike majú svoj pôvod v skutočných bunkách živých organizmov. Z nich sa získavajú sekvenovaním – zložitými biochemickými reakciami sa biologické molekuly postupne upravia do podoby, v ktorej ich dokáže digitalizovať sekvenovací prístroj. Počas tohoto procesu dochádza k chybám, a to biologickým (kontaminované vzorky) ako aj technickým

(sekvenovací prístroj zle zaznamená bázu). Navyše, sekvenované biologické molekuly sa počas fázy prípravy rozsekajú na relatívne krátke úseky a spájajú sa až v zdigitalizovanej podobe počítačmi, pričom opäť môže dôjsť k chybám. Pre ilustráciu, ľudský genóm má 3 miliardy báz a na 10 000 báz pripadá priemerne menej ako jedna chyba [Dolgin, 2009].

Vďaka svojej prístupnosti a užitočnosti sa pri predikcií génov veľmi často využívajú EST sekvencie (*expressed sequence tag*). Proces vzniku EST sekvencií opisujú napríklad [Nagaraj et al., 2006]. Po ukončení expresie génov v bunke zostávajú molekuly mRNA (obr. 1.3). Tieto molekuly už prešli fázami transkripcie a zostrihu, takže sú tvorené len zreťazenými exónmi a z oboch strán zakončené neprekladanými úsekmi. mRNA molekuly sa nedajú priamo sekvenovať, preto sa z nich enzýmami vyrobia dvojvláknové cDNA (komplementárna DNA) sekvencie. cDNA sekvencie sa sekvenujú z oboch koncov, čím vznikajú 5'EST a 3'EST sekvencie. Ich dĺžka sa pohybuje medzi niekoľkými desiatkami až stovkami báz.

Zdigitalizované sekvencie sa zhromažďujú v rozsiahlych databázach, z ktorých mnohé sú verejne prístupné. Medzi najvýznamnejšie databázy patria GenBank [NCBI, a], zhromažďujúca všetky verejne publikované sekvencie (verzia z 15. apríla 2010 obsahuje vyše 100 miliónov anotovaných DNA sekvencií, z toho 65 miliónov EST sekvencií); RefSeq [NCBI, c], obsahujúca referenčné sekvencie celkovo z desiatky tisíc organizmov (verzia z 9. mája 2010 obsahuje 2 milióny DNA sekvencií, 2 milióny RNA sekvencií a 10 miliónov proteínových sekvencií); a NCBI Protein [NCBI, b], obsahujúca proteínové sekvencie (verzia z 3. júna obsahuje 33 miliónov proteínových sekvencií).

## 2.2 Zarovnanie sekvencií

Jedným z najzákladnejších problémov v bioinformatike je zarovnanie dvoch sekvencií (reťazcov). Pod zarovnaním rozumieme spárovanie báz z jednej sekvencie s bázami z druhej sekvencie. Snažíme sa spárovať čo najviac podobných alebo veľmi rovnakých úsekov v oboch sekvenciách. Na miesta, kde sa sekvencie nezhodujú, môžeme vložiť medzery. Tie reprezentujú vkladanie (inzercia) alebo odstraňovanie (delécia) báz z jednej alebo druhej sekvencie. Spárovanie nerovnakých báz reprezentuje mutáciu v jednej zo sekvencií. Kvalita zarovnania sa vyhodnocuje na základe nejakej skórovacej schémy. Veľmi jednoduchá schéma je napríklad +1 za každú zhodnú dvojicu báz a -1 za

Obr. 2.1: Zarovnanie sekvencií AGATCAACTG a AATCGTG.

(a) Globálne zarovnanie (skóre 2)

```
AGATCAACTG
| |||   ||
A-ATC--GTG
```

(b) Lokálne zarovnanie (skóre 3)

```
AA-C-TG
||  | ||
AATCGTG
```

Skórovacia schéma: zhoda +1, nezhoda alebo medzera -1.

nerovnaké bázy alebo medzeru v jednej zo sekvencií. Skórovacia schéma je zvyčajne matica typu $|\Sigma| \times |\Sigma|$ obsahujúca skóre zarovnania dvojíc písmen (označovaná aj ako substitučná matica) a záporné číslo $g$ určujúce penalizáciu za zarovnanie bázy s medzerou. Zvyčajne je pri DNA sekvenciách hodnota zarovnania rovnaká bez ohľadu na písmeno, pričom hodnota zarovnania rovnakých písmen je kladná, hodnota zarovnania nerovnakých písmen záporná a hodnota zarovnania s medzerou záporná a menšia alebo rovná hodnote zarovnania nerovnakých písmen.

Dva základné typy zarovnaní sú globálne zarovnanie (obr. 2.1a) a lokálne zarovnanie (obr. 2.1b). Globálne zarovnanie znamená nájdenie zarovnania kompletných vstupných sekvencií s najvyšším skóre, lokálne zarovnanie znamená zarovnanie nejakých podreťazcov vstupných sekvencií s najvyšším skóre. Jedna zo sekvencií sa označuje ako dotaz (query), druhá ako databáza (target).

Pri hľadaní génov s použitím externých dát sa lokálne zarovnanie sekvencií využíva veľmi často. Typickým príkladom je lokálne zarovnanie EST sekvencií alebo proteínových sekvencií ku genómu. V našej práci sme intenzívne používali programy hľadajúce lokálne zarovnania, preto v nasledujúcom texte popíšeme algoritmus na nájdenie lokálneho zarovnania s najvyšším skóre a heuristické prístupy k lokálnemu zarovnaniu.

### 2.2.1 Lokálne zarovnanie

Algoritmus riešiaci problém lokálneho zarovnania je *Smithov-Watermanov algoritmus* [Smith and Waterman, 1981]. Pre dané sekvencie $P = p_1 p_2 \dots p_n$, $Q = q_1, q_2, \dots q_m$ a danú skórovaciu schému $S$ nájde zarovnanie podreťazcov $p_i \dots p_j, q_k \dots q_l$ s najvyšším skóre.

Smithov-Watermanov algoritmus je modifikáciou algoritmu pre hľadanie globálneho zarovnania s najvyšším skóre [Needleman and Wunsch, 1970] a je rovnako, ako Needlemanov-Wunschov algoritmus, založený na dynamickom

programovaní. Algoritmus pracuje tak, že postupne vypĺňa maticu $A$ typu $(n+1) \times (m+1)$. Prvok $A[i,j]$ obsahuje skóre takého zarovnania podreťazcov $p_1 \ldots p_i$, $q_1 \ldots q_j$, ktoré končí dvojicou $p_i$ a $q_j$ a má najvyššie skóre, alebo prázdneho zarovnania (skóre 0). Matica $A$ sa inicializuje prázdnym zarovnaním

$$A[i,0] = 0 \qquad \forall i \in \langle 0,n \rangle \tag{2.4}$$

$$A[0,j] = 0 \qquad \forall j \in \langle 0,m \rangle \tag{2.5}$$

a zvyšné hodnoty sa vypočítajú podľa rekurentného vzťahu

$$A[i,j] = \max \left\{ \begin{array}{l} 0, \\ A[i-1,j-1] + S[p_i,q_j], \\ A[i-1,j] + g, \\ A[i,j-1] + g \end{array} \right\}, \ \forall i \in \langle 1,n \rangle, j \in \langle 1,m \rangle. \tag{2.6}$$

Zároveň sa pre účely zrekonštruovania hľadaného zarovnania do $A[i,j]$ uloží ukazovateľ na tú z trojice buniek $A[i-1,j-1]$, $A[i-1,j]$, $A[i,j-1]$, z ktorej sa hodnota $A[i,j]$ vypočítala, alebo nulový ukazovateľ, ak je hodnota bunky $A[i,j]$ rovná 0.

Pri zostavovaní zarovnania s najvyšším skóre sa v matici $A$ nájde prvok s najvyšším skóre a sledovaním ukazovateľov až po prvý nulový ukazovateľ sa zarovnanie spätne konštruuje podľa hodnoty ukazovateľa: *a*) ak ukazuje na bunku $A[i-1,j-1]$, v zarovnaní je dvojica $p_i$-$q_j$; *b*) ak ukazuje na bunku $A[i-1,j]$, v zarovnaní je $p_i$ s medzerou; *c*) ak ukazuje na bunku $A[i,j-1]$, v zarovnaní je medzera s $q_j$.

Takýto základný model sa dá rozšíriť napríklad o afínne skórovanie medzier – medzery na susedných pozíciách v zarovnaní pravdepodobne nevznikli nezávisle, ale vložením alebo vymazaním viacerých písmen naraz. Prvú (otváraciu) bázu zarovnanú s medzerou penalizujeme nejakou hodnotou $o$, ďalšie bázy rozširujúce medzeru hodnotou $e$, pričom $e > o$. Skóre medzery dĺžky $l_0$ je teda $g = o + e(l_0 - 1)$, čo je viac ako skóre $k > 1$ medzier v súčte rovnakej dĺžky $l_1 + \ldots + l_k = l_0$.

Obr. 2.2: Smithov-Watermanov algoritmus na sekvenciách AGATCAACTG a AATCGTG

(a) Matica zarovnania

| | | A | G | A | T | C | A | A | C | T | G |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| A | 0 | 1 | 0 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 0 |
| A | 0 | 1 | 0 | 1 | 0 | 0 | 1 | 2 | 1 | 0 | 0 |
| T | 0 | 0 | 0 | 0 | 2 | 0 | 0 | 1 | 1 | 2 | 1 |
| C | 0 | 0 | 0 | 0 | 1 | 3 | 2 | 1 | 2 | 1 | 1 |
| G | 0 | 0 | 1 | 0 | 0 | 2 | 2 | 1 | 1 | 1 | 2 |
| T | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 0 | 2 | 1 |
| G | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 3 |

(b) Nájdené zarovnania (skóre 3)

```
AA-C-TG
||  |  ||
AATCGTG

ATG
|||
ATG
```

Skórovacia schéma: zhoda +1, nezhoda alebo medzera -1.

### 2.2.2 Použitie heuristík v lokálnom zarovnaní

Časová aj pamäťová zložitosť algoritmu Smith-Waterman je $O(nm)$. Pre praktické použitie na úrovni chromozómov, alebo celých genómov, je čas potrebný na výpočet príliš veľký. V praxi sa používajú rôzne heuristiky, ktoré síce nevypočítajú zaručene optimálne zarovnania, no dokážu nájsť dostatočne dobré riešenia dostatočne rýchlo.

Princíp mnohých heuristických metód spočíva v rýchlom vyhľadaní *jadier zarovnaní* (high-scoring segment pairs, HSPs, krátke zarovnania sekvencií rovnakej dĺžky bez medzier), ich vhodnom spojení do väčších celkov a doladení výsledného zarovnania. Časovo „drahé“ dynamické programovanie sa aplikuje len na vybrané relatívne krátke úseky. V nasledujúcom texte rozumieme pod číslom diagonály $d$ hodnotu $d = i - j$, kde $i$ označuje začiatok zarovnania v databázi a $j$ začiatok zarovnania v dotaze.

### 2.2.3 Programy

#### BLAST

BLAST (Basic Local Alignment Search Tool) [Altschul et al., 1990], [Altschul et al., 1997] je vďaka svojej rýchlosti veľmi obľúbený nástroj pri zarovnávaní veľmi dlhých sekvencií. Existuje niekoľko verzií základného programu: na zarovnanie dvoch DNA sekvencií; dvoch proteínových sekvencií; proteínovej sekvencie k DNA sekvencii a ďalšie. Algoritmus funguje v niekoľkých fázach.

(a) *Zostavenie indexu.* Najprv sa vytvorí zoznam všetkých $k$-tic z dotazu, ktoré sa zarovnajú s nejakou $k$-ticou z databázy so skóre aspoň $T$. V prípade dvoch DNA sekvencií sa zhoda hodnotí +5, nezhoda -4; proteínové sekvencie sú ohodnotené podľa matice BLOSUM62 [Heinkoff and Heinkoff, 1992].

(b) *Vytvorenie jadier zarovnaní.* Ak sú dve vybrané $k$-tice v pomyselnej matici lokálneho zarovnania na jednej diagonále vo vzdialenosti menej ako $A$, spoja sa do jedného úseku a tento úsek sa rozširuje na obe strany, pokiaľ skóre úseku rastie. Vzniknutý úsek je jadro zarovnania.

(c) *Rozšírenie zarovnania o medzery.* Jadrá zarovnaní so skóre aspoň $S_g$ sa spoja medzerami, pričom sa použije dynamické programovanie a zovšeobecnené afínne skórovanie, umožňujúce medzery v oboch zarovnávaných sekvenciách.

(d) *Odfiltrovanie náhodných zarovnaní.* Napokon sú ako výstup vrátené tie zarovnania, pre ktoré je očakávaný počet zarovnaní s rovnakým skóre v náhodných sekvenciách menší ako nejaké zvolené $E$.

**Sim4**

Sim4 [Florea et al., 1998] je program určený na správne zarovnanie EST sekvencií ku genómu, ktoré berie ohľad na výskyt intrónov a sekvenovacích chýb.

(a) *Nájdenie jadier zarovnaní.* Program najprv deteguje presné zhody dĺžky 12 a rozširuje ich oboma smermi so skóre 1 za zhodu bázových párov a -5 za nezhodu. Rozširovanie sa, podobne ako pri programe BLAST, končí vtedy, keď už rozšírenie segmentu neprispieva k zvýšeniu skóre.

(b) *Zreťazenie jadier do dlhších segmentov.* V druhom kroku sa dynamickým programovaním vyberie najlepšie zreťazenie jadier vzhľadom na podmienky *i*) počiatočné pozície jadier v exprimovanej sekvencii sú vo vzostupnom poradí; a *ii*) susedné jadrá sú na takmer rovnakých diagonálach, alebo sa ich diagnonály odlišujú dostatočne na to, aby mohli reprezentovať intrón.

(c) *Vytýčenie hraníc exónov.* Susedné „jadrá exónov“ (niekoľko jadier zarovnaní na blízkej diagonále) sa buď spoja, alebo rozšíria tak, aby úseky

zodpovedajúce intrónom obsahovali signály zostrihu ($GT \ldots AG$, respektíve obrátený komplement $CT \ldots AC$). To vo všeobecnosti nemusí byť možné, v takom prípade sa na danom úseku hľadajú jadrá s kratšou dĺžkou a zopakujú sa kroky *b*) a *c*).

(d) *Určenie konečného zarovnania.* Presné zarovnania exónov sa doladia programom Sim3 [Chao et al., 1997], určeným na zarovnanie veľmi podobných sekvencií.

**BLAT**

BLAT (The BLAST Like Alignment Tool) [Kent, 2002] vznikol počas projektu sekvenovania ľudského genómu, kedy bolo potrebné pravidelne zarovnávať veľký počet EST sekvencií a kúskov myšacieho genómu k ľudskému genómu. BLAT dokáže zarovnať DNA, RNA alebo proteínové sekvencie a tiež zarovnať DNA alebo RNA sekvencie k proteínovej sekvencii prekladom do proteínovej sekvencie.

(a) *Počiatočná fáza budovania jadier zarovnaní.* Na rozdiel od BLAST-u, BLAT buduje index neprekrývajúcich sa $k$-tic z databázy. Medzi týmito $k$-ticami hľadá prekrývajúce sa $k$-tice z dotazu. BLAST uvažuje aj prekrývajúce sa $k$-tice a index buduje pre dotaz, ktorý je väčšinou kratší. Prístup programu BLAT je časovo aj pamäťovo efektívnejší.

(b) *Určenie jadier zarovnaní a homologických úsekov.* „Pseudojadrom“ zarovnania je pri proteínoch jedna $k$-tica; pri DNA sekvenciách dve $k$-tice, medzi ktorými je medzera menšia ako 2. Blízke pseudojadrá sa spoja do jadier zarovnaní. Spojením blízkych jadier sa vytvoria homologické úseky.

(c) *Rozšírenie homologických úsekov.* Vzniknuté homologické úseky sú základm pre fázu dolaďovania zarovnaní, ktorá je pre DNA sekvencie iná ako pre proteínové sekvencie.

   *i*) *DNA sekvencie.* V prvej fáze sa homologické úseky rozšíria len o zarovnania nepovoľujúce nezhody a medzery. Väčšie medzery medzi homologickými úsekmi sa algoritmus snaží rekurzívne vyplniť hľadaním čoraz kratších zhodných úsekov, až pokým sa nenájdu žiadne ďalšie zhodné úseky alebo je už medzera kratšia ako 5 báz.

Takto vytvorené segmenty sa ďalej rozširujú aj o zarovnania s jednou alebo dvomi nezhodami alebo medzerami. Na rozdiel od BLASTu sa väčšie medzery umiestnia s ohľadom na signály zostrihu, aby mohli reprezentovať intrón. Autori uvádzajú, že táto stratégia je vhodná predovšetkým pre sekvencie, ktoré sú si podobné aspoň na 95%, čo znamená, že pri zarovnávaní sekvencií z rôznych druhov nemusí priniesť dobré výsledky.

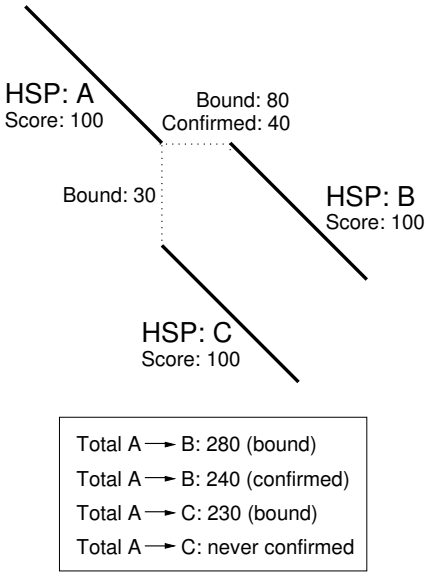
*ii*) *Proteínové sekvencie.* Jadrá zarovnaní sa rozšíria na obe strany, tak aby dosiahli maximálne skóre; skóre za zhodu je +2, za nezhodu -1. Zostrojí sa graf s ohodnotenými hranami, ktorého vrcholmi sú rozšírené jadrá. Medzi ľubovoľnou dvojicou vrcholov $u, v$ je hrana z $u$ do $v$ vtedy, keď úsek zodpovedajúci vrcholu $u$ začína v databázi aj v dotaze skôr ako úsek vrcholu $v$. Hodnota takejto hrany je skóre jadra $v$ mínus penalizácia závislá od dĺžky medzery medzi jadrami. Možný prekryv jadier $u$ a $v$ sa vyrieši zostrojením nového vrcholu zodpovedajúcemu nejakej pozícií $i$ tak, aby sa maximalizoval súčet skóre od začiatku $u$ po $i$ a od $i$ po koniec $v$; cena hrany z $u$ do $i$ je skóre celého $u$ mínus skóre úseku od začiatku $u$ po $i$, analogicky sa vypočíta cena hrany z $i$ do $v$. Na takto zostrojenom grafe sa vypočíta najdrahšia cesta, reprezentujúca zarovnanie.

(d) *Zostrojenie výsledného zarovnania.* Homologické úseky sa spoja obdobným algoritmom, akým sa spájajú jadrá proteínových zarovnaní. Penalizácia za medzeru v zarovnaní DNA sekvencií je konštanta + logaritmus dĺžky medzery. Zostávajúce medzery pri zarovnaní mRNA k DNA alebo DNA k DNA sa program pokúsi vyplniť opakovaným spustením zarovnávacieho algoritmu.

### Exonerate

Program Exonerate [Slater and Birney, 2005] je komplexný nástroj na zarovnávanie sekvencií. Na výpočet zarovnania vie použiť viacero modelov. Medzi nimi nechýba štandardný Smithov-Watermanov algoritmus, s možným afínnym skórovaním medzier a heuristické modely na zarovnanie dlhších sekvencí.

(a) *Nájdenie jadier zarovnaní.* Jadrá zarovnaní sa nájdu rovnako ako v programe BLAST.




Obr. 2.3: Zjednodušený príklad BSDP v programe Exonerate. Najvyšší horný odhad skóre nejakého úseku je 280, preto sa v prioritnom fronte overuje prvý. Vypočítaná hodnota 240 je viac ako horný odhad skóre úseku A-C, takže na tomto úseku sa DP nepočíta, čím sa ušetrí čas. Pri zarovnaní skutočných sekvencí sa zvyčajne týmto spôsobom preskočí viacero úsekov, takže úspora je väčšia. [Slater and Birney, 2005]

(b) *Určenie oblastí na spresnenie zarovnaní.* Pre každé jadro sa určia dve oblasti, na ktoré sa neskôr aplikuje dynamické programovanie. Sú to:

*(i)* začiatok jadra;

*(ii)* koniec jadra;

*(iii)* v prípade malej medzery medzi dvomi jadrami sa určí jedna spoločná oblasť pre koniec jedného jadra a začiatok druhého jadra;

*(iv)* v prípade veľkej medzery medzi dvomi jadrami (potenciálne znamenajúcej intrón) sa koncová oblasť prvého a začiatočná oblasť druhého jadra stanú závislými a výpočet druhej oblasti bude závisieť od výsledku výpočtu na prvej oblasti.

Konkrétny algoritmus dynamického programovania sa určí na základe typu zarovnávaných sekvencií a typu oblasti, na ktorú bude aplikovaný.

(c) *Odstránenie neperspektívnych oblastí.* Keďže aplikovaním dynamického programovania na každú z vybraných oblastí by sa oproti Smithovmu-Watermanovmu algoritmu veľmi nezlepšil čas výpočtu, niektoré neperspektívne oblasti sa musia odstrániť. Pre každú oblasť sa vypočíta odhad maximálneho skóre zarovnania. Na základe tohoto odhadu sa ohraničeným riedkym dynamickým programovaním (bounded sparse dynamic programming, BSDP) nepočítajú zarovnania v neperspektívnych oblastiach.

BSDP je implementované niekoľkými prioritnými frontami. Pre každé jadro sa zostrojí front obsahujúci záznam pre každé čiastkové zarovnanie končiace v danom jadre zoradený od najvyššieho skóre (na začiatku sú namiesto exaktne vypočítaných hodnôt len predpočítané maximálne odhady). Navyše sa zostrojí globálny prioritný front pozostávajúci z maximálnych skóre ostatných frontov. Postupne sa dynamickým programovaním overujú odhady úsekov nachádzajúcich sa v globálnom fronte (počnúc najvyšším odhadom) a nahradzujú sa skutočne vypočítanými hodnotami (obr. 2.3). Ak sú vypočítané hodnoty nižšie, do globálneho frontu sa vyberie nový zástupca daného jadra. Potvrdením všetkých odhadov v globálnom fronte sa určí výsledné zarovnanie.

Tento algoritmus, rovnako ako $A^*$ prehľadávanie zachováva prípustnosť, čiže výsledok výpočtu je rovnaký ako výsledok výpočtu na všetkých oblastiach.

(d) *Modely.* Overovanie horných odhadov čiastkových zarovnaní sa počíta algoritmom závislým od typu zarovnávaných sekvencií a od typu oblasti. Všetky algoritmy sú odvodené od ich modelov. Modely sú reprezentované konečnými automatmi a sú odvodené od základného modelu. Základným modelom je jednoduchá substitučná matica umožňujúca len zarovnanie bez medzier. Postupným rozširovaním jej modelu možno získať modely pre algoritmy na globálne a lokálne zarovnania, s afínnym skórovaním medzier, s možnosťou určenia intrónov alebo model zarovnávajúci proteín k DNA. Program Exonerate obsahuje mechanizmus automatického generovania týchto modelov a automatického generovania zodpovedajúcich algoritmov.

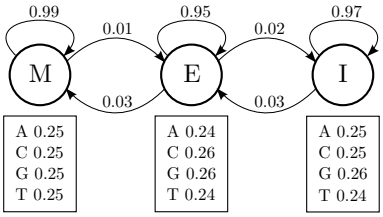
## 2.3 Hľadanie génov

Aby sme mohli formálne uviesť problém hľadania génov, definujme si množinu $G$ biologických pojmov. Vhodnou množinou pri hľadaní génov je napríklad

$$\begin{aligned} G \;=\; \{&\textit{exón, intrón, 5'UTR, 3'UTR, štart kodón,}\\ &\textit{stop kodón, donor, akceptor, medzigénový úsek}\}. \end{aligned} \tag{2.7}$$

V nasledujúcom texte bude slovo sekvencia znamenať DNA sekvenciu,

Obr. 2.4: Jednoduchý HMM na anotáciu DNA sekvencií. Stavy: M – medzigénový úsek, E – exón, I – intrón.




pokiaľ nebude uvedené inak. Pre ľubovoľnú sekvenciu $S, |S| = n$ definujeme jej anotáciu $A_S$ ako prvok množiny $G^n$. Inými slovami, každú pozíciu v sekvencii $S$ označíme nejakým biologickým pojmom – prvkom $G$.

Pod hľadaním (predikciou) génov v DNA sekvencií $S$ rozumieme priradenie takej anotácie $A$, ktorá rozumne popisuje biologickú štruktúru danej sekvencie.

Na rozlíšenie anotácií, ktoré dobre charakterizujú dané DNA sekvencie, od biologicky nezmyselných anotácií, potrebujeme skórovací systém. Ten by mal priradiť vysoké skóre z pohľadu biologicky rozumným anotáciám (napríklad sekvencia s génom podobným tomu na obrázku 1.2) a nízke skóre biologicky nezmyselným anotáciám (napríklad striedajúce sa intróny a medzigénové úseky, alebo stop kodón v strede exónu).

### 2.3.1 Skryté Markovove modely

Biologicky relevantné anotácie sa často modelujú skrytými Markovovými modelmi. **Skrytý Markovov model** (Hidden Markov model, HMM) $H$ je pätica $(K, \Sigma, \pi, a, e)$, kde $K$ je konečná množina stavov, $\Sigma$ je vstupná abeceda, $\pi$ je pravdepodobnostná miera na množine $K$, $a$ je množina $a = \{a_q \mid q \in K\}$ pravdepodobnostných mier na množine $K$ a $e$ je množina $e = \{e_q \mid q \in K\}$ pravdepodobnostných mier na množine $\Sigma$. Pre daný stav $q \in K$ je $\pi(q)$ pravdepodobnosť, že výpočet na HMM začne v stave $q$. Pre dané stavy $q_1, q_2 \in K$ je $a_{q_1}(q_2)$ pravdepodobnosť, že výpočet na HMM prejde zo stavu $q_1$ do stavu $q_2$. Pre daný stav $q \in K$ a daný symbol $x \in \Sigma$ je $e_q(x)$ pravdepodobnosť, že model v stave $q$ vygeneruje symbol $x$. Anotačnou funkciou $f : K \to G$ priradíme každému stavu nejaký biologický význam. Anotáciu $A$ cesty $A' = A'_1 \dots A'_n$ získame aplikovaním anotačnej funkcie $A = f(A'_1) \dots f(A'_n)$.

Pozorný čitateľ si istotne všimol podobnosť HMM s konečným automatom. Na rozdiel od konečného automatu môže mať HMM viacero počiatočných stavov, symboly generuje v stavoch, nie pri prechodoch (pričom v každom stave môže vygenerovať potenciálne akýkoľvek symbol z abecedy) a nemá akceptačné stavy. Podrobnejšie sa skryté Markovove modely preberajú napríklad v knihe [Durbin et al., 1998].

V danom modeli $H = (K, \Sigma, \pi, a, e)$ vieme pre danú sekvenciu $S = s_1 s_2 \dots s_n$ a nejakú cestu stavmi $A' \in K^n, A' = A'_1, A'_2, \dots, A'_n)$ vypočítať pravdepodobnosť výskytu (vygenerovania) $S$ a $A'$ ako

$$P(S, A') = \pi(a_1) e_{a_1}(s_1) \prod_{i=2}^{n} a_{a_{i-1}}(a_i) e_{a_i}(s_i). \tag{2.8}$$

Napríklad model z obrázku 2.4 vygeneruje sekvenciu $CATG$ a jej anotáciu $MEEI$ s pravdepodobnosťou (predpokladajme, že vždy začne v medzigénovom úseku):

$$P(CATG, MEEI) = 0.25 \cdot 0.01 \cdot 0.24 \cdot 0.95 \cdot 0.24 \cdot 0.02 \cdot 0.26. \tag{2.9}$$

Pravdepodobnosť nezmyselnej anotácie $MIMI$ je nula, keďže v danom modeli neexistuje prechod zo stavu $M$ do stavu $I$ ani zo stavu $I$ do stavu $M$; inými slovami $a_M(I) = a_I(M) = 0$, takže v súčine je nulový člen.

Pre danú sekvenciu $S$ vieme, aké písmená má model vygenerovať, ale cesta, ktorou sa k emitovanej sekvencii dopracujeme je *skrytá*. Možných ciest (a teda možných anotácií) je často veľmi veľa. Ak veríme, že náš model dobre modeluje biologickú realitu, mali by sme vybrať takú cestu, ktorej prechodom vygenerujeme vstupnú sekvenciu s najväčšou pravdepodobnosťou.

Formálne teda definujeme problém hľadania génov nasledovne: pre danú sekvenciu $S \in \Sigma^*, |S| = n$, nájdi najpravdepodobnejšiu cestu stavmi $A'_{max} \in K^n$ takú, že

$$A'_{max} = \arg \max_{A' \in K^n} P(A' \mid S). \tag{2.10}$$

### Viterbiho algoritmus

Algoritmus efektívne riešiaci problém (2.10) sa nazýva Viterbiho algoritmus [Rabiner, 1989]. Aj tento algoritmus je založený na dynamickom programovaní. Majme danú sekvenciu $S = s_1 \dots s_n$ a model $H = (K, \Sigma_D, \pi, a, e)$. Viterbiho algoritmus vypĺňa maticu $V$ typu $|K| \times n$. Prvok $V[q, s_i]$ obsahuje

pravdepodobnosť najpravdepodobnejšej cesty modelom $H$ emitujúcej sekvenciu $s_1 \dots s_i$ a končiacej v stave $q$. Inicializácia matice $V$ sa vykoná prenásobením počiatočných pravdepodobností stavov a príslušných emisných pravdepodobností prvého písmena sekvencie:

$$V[q,1] = \pi(q)e_q(s_1) \qquad \forall q \in K. \tag{2.11}$$

Ďalšie bunky matice sa počítajú z predchádzajúcich, pričom sa, podobne ako v prípade Smithovho-Watermanovho algoritmu, ukladá ukazovateľ na predchádzajúcu bunku v doteraz najpravdepodobnejšej ceste:

$$V[q,i] = e_q(s_i) \max_{k \in K} \{V[k, s_{i-1}]a_k(q)\}, \quad \forall q \in K, i \in \langle 2, n\rangle, \tag{2.12}$$

$$p[q,i] = \arg\max_{k \in K} \{V[k, s_{i-1}]a_k(q)\}, \qquad q \in K, i \in \langle 2, n\rangle. \tag{2.13}$$

Pravdepodobnosť najpravdepodobnejšej cesty $A' = A'_1 \dots A'_n$ a sekvencie $S$ je rovná

$$P(S, A') = \max_{q \in K}\{V[q,n]\}. \tag{2.14}$$

Cesta sa zostaví spätným prechodom po ukazovateľoch:

$$A'_n = \arg\max_{q \in K}\{V[q,n]\}, \tag{2.15}$$

$$A'_i = p[q, i+1], \quad \forall i \in \langle 1, n-1\rangle. \tag{2.16}$$

Časová zložitosť Viterbiho algoritmu je $O(n|K|^2)$, pamäťová zložitosť je $O(n|K|)$.

### 2.3.2 Zostavenie modelu

Architektúra modelu je kľúčová pri správnom určovaní génov a ich štruktúry. Aby HMM pre hľadanie génov vedel spoľahlivo rozlíšiť medzigénové úseky, intróny a exóny, musí obsahovať stavy pre signály zostrihu (akceptor, donor), pamätať si informáciu o pozícii v kodóne, rozoznať štart a stop kodóny a pod. Reálne používané modely majú množstvo stavov, napríklad skrytý Markovov model programu Augustus ich má 51 [Stanke and Waack, 2003b].

So zvyšujúcim sa počtom stavov úmerne rastie počet parametrov modelu – pravdepodobnostných mier $a_q$, $e_q$. Ručné nastavenie všetkých parametrov by s najväčšou pravdepodobnosťou neviedlo k správnemu fungovaniu mo-

delu. Modely sa preto musia trénovať na už oanotovaných dátach. To môže byť problém pri organizmoch, ktorých genómy ešte nie sú v dostatočnej dobre preskúmané a sú evolučne veľmi vzdialené od organizmov s dobre oanotovanými genómami. Navyše, genómy vzdialených organizmov sa veľmi líšia. Napríklad drozofila má kratšie gény s menším počtom exónov a intrónov ako človek. V podstate nie je možné určiť parametre modelu tak, aby dokázal správne predikovať gény vo viacerých organizmoch súčasne. Parametre modelu sa preto musia natrénovať pre každý organizmus zvlášť.

### 2.3.3 Modifikácie HMM

Skryté Markovove modely používané na riešenie problémov praxe sa často v niektorých aspektoch odlišujú od základnej podoby, akú sme predstavili v tejto kapitole. Medzi obľúbené modifikácie patria stavy vyšších rádov – rád stavu $k$ znamená, že emisné pravdepodobnosti závisia od predchádzajúcich $k$ emitovaných symbolov.

Ďalšou častou úpravou je možnosť jedným prechodom daným stavom emitovať viacero symbolov z abecedy, pričom ich presný počet nemusí byť konštantný a môže kopírovať nejaké pravdepodobnostné rozdelenie. Takýto model sa nazýva skrytý polo-Markovov model (hidden semi-Markov model, HSMM).

Pravdepodobnosť, že klasický HMM zotrvá v danom stave $q$ čas $d$ (predpokladajme, že pravdepodobnosť prechodu zo stavu $q$ späť do $q$ je nenulová) je rovná $P_q(d) = a_q(q)^{d-1}(1 - a_q(q))$. Náhodná premenná $D_q$, znamenajúca dobu, počas ktorej daný HMM zotrvá v stave $q$, má geometrické rozdelenie. Z toho vyplýva, že aj dĺžky sekvencií emitovaných počas doby $D$ majú geometrické rozdelenie. Dĺžka mnohých prvkov v skutočných DNA sekvenciách však nemá geometrické rozdelenie.

HSMM dokáže modelovať ľubovoľné pravdepodobnostné rozdelenie dĺžok emitovaných sekvencií. Dané rozdelenie dĺžok sekvencií emitovaných daným stavom $q$ sa v HSMM modeluje tak, že najprv sa v závislosti od daného rozdelenia určí doba $d$, ktorú model zotrvá v stave $q$ a potom sa emituje $d$ písmen z abecedy. Spätný prechod do stavu $q$ nie je možný. Podrobnejšie sa teória HSMM rozoberá v [Rabiner, 1989] (časť IV. D).

Nevýhodou je, že časová zložitosť Viterbiho algoritmu pri prechode upraveným stavom je priamo úmerná najväčšej možnej dĺžke. To môže byť nepraktické pri modelovaní dlhých úsekov, napríklad intrónov.

### 2.3.4 Externé dáta

Hľadanie génov modelmi využívajúcimi len DNA sekvenciu sa v literatúre označuje ako *ab-initio* predikcia. Výhodou *ab-initio* programov je ich všeobecné použitie na v podstate akýkoľvek organizmus. Okrem DNA sekvencií sa pri hľadaní génov využívajú aj tzv. *externé dáta*. Sú to v podstate akékoľvek rozumné informácie o skúmanej sekvencii. Prístupy využívajúce externé dáta dokážu byť presnejšie, avšak nedajú sa použiť na druhy, pre ktoré ešte dodatočné dáta nie sú k dispozícii.

#### Zarovnanie dlhých DNA sekvencií

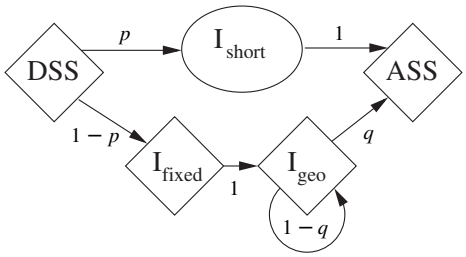
Zarovnaním správne oanotovanej sekvencie $S$ k sekvencii s neznámou anotáciou $S'$ nájdeme zhodné alebo veľmi podobné úseky. Môžeme predpokladať, že aj anotácie týchto podobných úsekov sú veľmi podobné. Zarovnanie je cenná informácia o pravdepodobnom výskyte génov a ich štruktúre. Na takéto zarovnanie sa dajú použiť pomerne dlhé DNA sekvencie (aj celé chromozómy) z evolučne príbuzných organizmov.

#### EST sekvencie

Zarovnaním veľkého množstva EST sekvencií k DNA sekvencii s neznámou anotáciou môžeme získať pomerne presné informácie o počte a polohe exónov. Zároveň sa tak dozvieme polohu intrónov a ich zodpovedajúcich signálov zostrihu. Veľmi často sú EST sekvencie jediným zdrojom externých dát.

#### Proteínové sekvencie

Ďalším dobrým zdrojom externých informácií sú zarovnania s proteínovými sekvenciami. DNA sekvencie sa nedajú k proteínom zarovnať priamo. Musia sa podľa genetického kódu (tab. 1.1) preložiť do proteínovej sekvencie. Preklad je možný šiestimi spôsobmi v takzvaných rámcoch (frames): kodóny sú trojperiodické a nie sú symetrické, takže posun začiatku prekladu o jedno alebo dve miesta a preklad komplementárneho vlákna v opačnom smere vedú k rôznym výsledkom. Tak ako zarovnanie EST sekvencií, aj zarovnanie proteínovej sekvencie dáva informácie predovšetkým o štruktúre exónov v génoch. Proteín sa navyše môže zarovnať k nefunkčnej kópii génu.




Obr. 2.5: Stavy HMM modelujúce intróny v programe Augustus. DSS – donor, ASS – akceptor. [Stanke and Waack, 2003b]

### 2.3.5 Programy

Súčasťou bakalárskej práce je aj porovnanie predikcií programu ExonHunter s vybranými programami založenými na skrytých Markovových modeloch. V nasledujúcom texte uvádzame prehľad použitých programov a stručný opis ich fungovania.

#### Augustus

Program Augustus [Stanke and Waack, 2003b] dokáže pri predikcií génov využiť externé dáta, no ich použitie nie je podmienkou. Jeho jadrom je skrytý Markovov model. Každý stav emituje určitý úsek sekvencie (viacero písmen naraz). Presné rozdelenie možných sekvencií a ich pravdepodobnosti sa určia trénovaním. Prechodové pravdepdobnosti medzi stavmi sú však pevne dané (trénovaním sa upravujú len prechody medzi stavmi modelujúcimi intróny).

Dôležitou súčasťou skrytého Markovovho modelu sú stavy modelujúce intróny. Keďže pri klasických HMM majú dĺžky emitovaných prvkov geometrické rozdelenie, uprednostňujú sa krátke intróny. Na druhej strane, HSMM s presným rozdelením dĺžok nie sú v porovnaní s klasickými modelmi výpočtovo efektívne. Programy Augustus a ExonHunter preto modelujú intróny viacerými stavmi (obr. 2.5), ktoré kombinujú oba spomenuté spôsoby. Intróny s dĺžkou menšou alebo rovnou nejakej konštante $d$ sú emitované stavom $I_{short}$, pre ktorý sa rozdelenie dĺžok intrónov určí z trénovacích dát. Intróny s dĺžkou $l$ väčšou ako $d$ sú emitované jedným prechodom do stavu $I_{fixed}$ a $l - d$ prechodmi cez stav $I_{geo}$.

#### ExonHunter

Program ExonHunter [Brejová et al., 2005] zakladá svoju predikciu na skrytom Markovovom modeli a na hierarchickom spracovaní externých dát.

Exóny a intróny sa modelujú podobne ako intróny v programe Augustus [Brejová and Vinař, 2002]. Rozšírený model je použitý aj pri modelovaní medzigénových úsekov, kde jeden z dvoch stavov generuje niekoľko $k$-tic (negeometrickým rozdelením) a druhý rovnomerne generuje sekvencie dĺžok 1 až $k$. Medzigénové úseky sú napríklad u človeka veľmi dlhé a tak tento prístup znamená kompromis medzi rýchlosťou a presnosťou.

ExonHunter sa snaží využiť čo najviac z dostupných externých dát – okrem samotnej DNA sekvecnice aj EST zarovnania, proteínové zarovnania a zarovnanie s iným genómom. Informácie z externých zdrojov reprezentuje ako „radcov“. Rady od všetkých dostupných radcov skombinuje do „superradcu“, ktorý ovplyvňuje priebeh výpočtu Viterbiho algoritmu. [Brejová et al., 2005] formálne definujú radcu pre sekvenciu $S = s_1 \dots s_n$ nasledovne:

> Rada od radcu $r$ pre bázu $s_i$ je partícia $\pi_a$ množiny $G$ (2.7) a pravdepodobnostná miera $P_a$ na prvkoch partície $g \in \pi_a$. Hodnota $P_a(g)$ je odhad pravdepodobnosti, že správna anotácia bázy $s_i$ je z množiny $g$, podľa informácií dostupných radcovi $a$.

Najdôležitejšou vlastnosťou takto definovaných radcov je možnosť posytnúť čiastkové rady. Informácie z externých dát nám totiž nemusia dať presnú anotáciu danej pozície. Príkladom môže byť zarovnanie EST sekvencie – zarovnaním k nejakému úseku ešte presne nevieme, či ide o neprekladaný úsek alebo exón. V takom prípade môže rada vyzerať nasledovne:

$$\pi_a = \left\{ \begin{array}{l} \{\textit{exón, 5'UTR, 3'UTR, štart kodón, stop kodón}\}, \\ \{\textit{intrón, donor, akceptor, medzigénový úsek}\}. \end{array} \right\}$$

$$\begin{aligned} P_a(\{\textit{exón, 5'UTR, 3'UTR, štart kodón, stop kodón}\}) &= 0.9 \\ P_a(\{\textit{intrón, donor, akceptor, medzigénový úsek}\}) &= 0.1. \end{aligned}$$

Takáto rada hovorí, že na 90% je zarovnaná báza exón alebo neprekladaný úsek, avšak nevieme presnejšie určiť, ktorá z týchto možností to naozaj je. V prípade lokálneho zarovnania krátkych EST alebo proteínových sekvencií k dlhej DNA sekvencii sa k mnohým úsekom DNA sekvencie nič nezarovná.

Koncept radcov vtedy použije „ničnehovoriacu radu“:

$$\begin{aligned} \pi_a &= \{G\} \\ P_a(G) &= 1. \end{aligned}$$

Opačným prípadom je úplná rada, kedy poznáme presnú pravdepodobnosť každého prvku množiny $G$

$$\pi_a = \{x \mid x \subseteq G, |x| = 1\}.$$

Rady od všetkých radcov sa skombinujú do superradcu. Počas tohoto procesu sa vyriešia konflikty medzi jednotlivými radami a výsledkom sú úplné rady pre každú pozíciu v skúmanej sekvencii. Rada superradcu na určitej poziícii je podľa [Brejová et al., 2005] definovaná ako pravdepodobnostná miera $p^*$ na množine $G$, kde $p^*(x)$ pre $x \in G$ je pravdepodobnosť daného prvku na danej pozícií, berúc do úvahy rady všetkých radcov.

Miera $p^*$ musí byť čo najbližšie ku rade každého radcu. Vzdialenosť rady radcu $a$ od rady superradcu je definovaná ako

$$dist_a(p^*) = \sum_{g \in \pi_a} \frac{1}{prior(g)} \left( P_a(g) - \sum_{x \in g} p^*(x) \right)^2, \tag{2.17}$$

kde $prior(x)$ pre $x \in G$ je frekvencia výskytu daného prvku v sekvenciách z trénovacej množiny a $prior(g) = \sum_{x \in g} prior(x)$.

Samotný výpočet sa realizuje kvadratickým programom minimializujúcim vahovaný súčet vzdialeností medzi radcami a výslednou mierou

$$\sum_a w_a \cdot dist_a(p^*) \tag{2.18}$$

vzhľadom na podmienky

$$\sum_{x \in G} p^*(x) = 1 \tag{2.19}$$

$$p^*(x) \geq 0 \quad \forall x \in G. \tag{2.20}$$

Posledným krokom je zahrnutie $p^*$ do výpočtu Viterbiho algoritmu. Výsledná anotácia $A$ by mala zohľadňovať okrem danej sekvencie $S$ aj dané

externé informácie $E$. Podľa Bayesovho pravidla

$$P(A|\ S,E) = \frac{P(S,E\ |\ A)P(A)}{P(S,E)}. \tag{2.21}$$

Za predpokladu, že externé informácie sú nezávislé na DNA sekvencii môžme (2.21) zjednodušiť. Hoci tento predpoklad je vo všeobecnosti nepravdivý, závislostiam medzi $S$ a $E$ sa dá vyhnúť nepoužívaním spoločných vlastností $S$ a $E$ v HMM a predikcií superradcu.

$$\begin{aligned} P(A|\ S,E) &= \frac{P(S|\ A)P(E|\ A)P(A)}{P(S)P(E)} \\ &= P(A|S)\frac{P(E|\ A)}{P(E)} \\ &= P(A|S)\frac{P(A|\ E)}{P(A)}. \end{aligned}$$

Za predpokladu, že anotácie jednotlivých pozícií sú od seba nezávislé (čo opäť nie je vo všeobecnosti pravda, ale závislostiam sa dá vyhnúť) sa dá pravdepodobnosť $P(A|\ E)$ vypočítať ako súčin rád superradcu na každej pozícii.

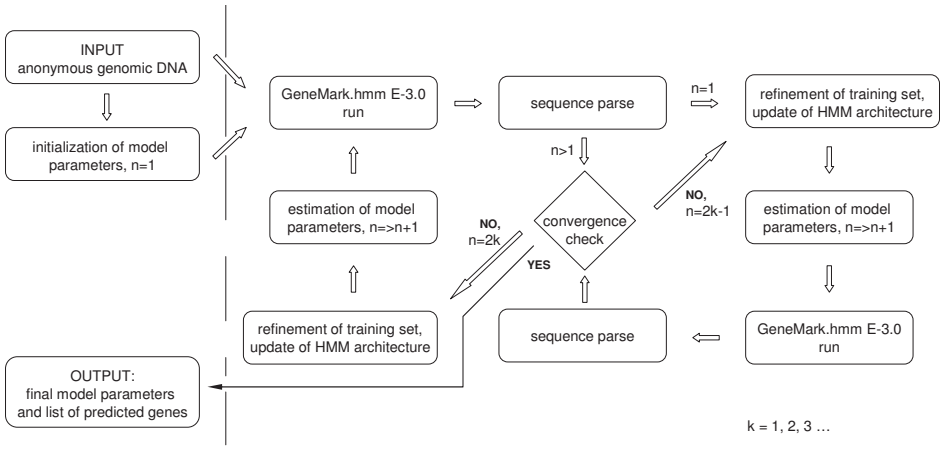
Najpravdepodobnejšia anotácia sa potom vypočíta upraveným Viterbiho algoritmom. Pri výpočte bunky $V[q,i]$ stačí prenásobiť emisné pravdepodobnosti číslom $p_i^*(f(q))/prior(f(q))$. Časová zložitosť algoritmu zostane zachovaná.

### GeneMark

Genemark.hmm ES-3.0 [Lomsadze et al., 2005] [Ter-Hovhanisyan et al., 2008] sa od ostatných programov výrazne líši najmä spôsobom, akým získava parametre pre svoj skrytý Markovov model. Zatiaľ čo napríklad ExonHunter sa sústredí na čo najlepšie využitie všetkých zdrojov informácií, GeneMark sa obmedzuje len na úplné minimum – vstupnú sekvenciu. To ho predurčuje na hľadanie génov v čerstvo oanotovaných organizmoch, pri ktorých dostatočný počet ďalších informácií absentuje. Vďaka iteratívnemu trénovaniu spojenému s predikciou génov Genemark nepotrebuje žiadne extra trénovacie dáta. Iteratívne trénovanie je zobrazené na obrázku 2.6.

*a*) *Inicializácia.* Inicializujú sa parametre skrytého polo-Markovovho modelu.

Obr. 2.6: Diagram trénovacieho algoritmu programu GeneMark. [Lomsadze et al., 2005]




*b*) *Predikcia.* Programom GeneMark.hmm E-3.0 sa vypočíta anotácia podľa aktuálnych parametrov.

*c*) *Úprava parametrov.* Vybrané anotácie sa použijú na úpravu aktuálnych parametrov.

*d*) *Iterácia.* Kroky *b*) a *c*) sa opakujú až kým sa nedosiahne konvergencia v biologicky relevantnom stave. Konvergenciu zaručuje výber anotácií v kroku *c*).

Autori uvádzajú, že tento postup vyžaduje vstupnú sekvenciu dlhú aspoň 10 miliónov báz. Dlhšie sekvencie neprispievajú k zlepšeniu výsledkov algoritmu, hoci vyžadujú väčší počet iterácií.

### GeneID

GeneID [Parra et al., 2000b] sa od ostatných programov odlišuje najmä použitou skórovacou schémou. Pracuje v troch fázach. V prvej fáze lokalizuje polohy signálov zostrihu a štart a stop kodónov. Na základe ich umiestnenia v druhej fáze určí potenciálne exóny. Na záver dynamickým programovaním [Guigó, 1998] zostaví výslednú predikciu.

Skórovacia schéma programu GeneID je založená na pozičných váhových maticiach (position weight matrix, PWM). PWM pre donor signál sa zostavia nasledovne. Z trénovacej množiny sa na základe úsekov okolo donor signálu zostavia matice $P$ a $Q$, kde $P_{ij}$ je apriórna pravdepodobnosť bázy $i$ na pozícii

$j$ a $Q_{ij}$ frekvencia výskytu bázy $i$ na pozícii $j$ ($j \in \langle -3, 6 \rangle$, kde 0 je pozícia prvej bázy intrónu). Z nich sa vypočíta PWM $D$,

$$D_{ij} = \log\left(\frac{P_{ij}}{Q_{ij}}\right).$$

Podobne sa vypočítajú PWM pre akceptor signál a štart kodón. Skóre každého potenciálneho donor signálu $s = s_1 \dots s_{10}$ sa vypočíta ako súčet príslušných buniek matice $L_D(s) = \sum_{i=1}^{10} D_{s_i i}$; obdobne sa počíta skóre akceptor signálu $L_A$ a skóre štart kodónu $L_S$.

Skóre exónov a intrónov je založené na kombinácii predikcie skrytého Markovovho modelu rádu 5 (emisné pravdepodobnosti závisia od 5 predchádzajúcich báz) a skóre prislúchajúcich signálov.

Z exónov v trénovacej množine sa zostaví matica prechodových pravdepodobností pre každý rámec v jednom smere, $F^1, F^2, F^3$. $F^j(s_1 s_2 s_3 s_4 s_5 s_6)$ je apriórna pravdepodobnosť šestice $s_1 s_2 s_3 s_4 s_5 s_6$ v rámci $j$ pri výskyte pätice $s_1 s_2 s_3 s_4 s_5$ v rámci $j$. Zo všetkých pätíc sa pre každý rámec zostaví matica počiatočných pravdepodobností $I^j$. Na základe intrónov v trénovacej množine sa vypočítajú matice prechodových pravdepodobností $F_0$ (zo šestíc) a počiatočných pravdepodobností $I_0$ (z pätíc).

Pre každú šesticu $h$ a rámec $j$ sa vypočíta log-likehood pomer

$$LF^j(h) = \log\frac{F^j(h)}{F_0(h)}$$

a pre každú päticu $p$ a rámec $j$ log-likehood pomer

$$LI^j(p) = \log\frac{I^j(p)}{I_0(p)}.$$

Skóre sekvencie $S$ dĺžky $l$ v rámci $j$ je definované ako

$$L_M(S) = LI^j(S_{1..5}) + \sum_{i=1}^{l-5} LF^j(S_{i..i+5}),$$

kde $S_{i..k}$ je úsek sekvencie $S$ od $i$ po $k$.

Následne sa skóre potenciálneho exónu v sekvencii $S$ s akceptor a donor signálmi $s_a, s_d$ počíta ako

$$L_E(S) = L_A(s_a) + L_D(s_d) + L_M(S).$$

Skóre vyjadruje log-likehood pomer pravdepodobnosti výskytu daných signálov a exónu k pravdepodobnosti výskytu náhodnej sekvencie medzi danými signálmi.

Celkovo sa teda skórovanie programu GeneID podobá na HMM, ale namiesto pravdepodobností sa používa log-likelihood pomer a namiesto prehľadávania všetkých postupností stavov Viterbiho algoritmom sa najskôr na základe signálov zostrihu určia exóny a z nich sa zostavia gény dynamickým programovaním. Hlavným dôvodom použitia takejto skórovacej schémy je výrazne nižšia časová a pamäťová náročnosť programu oproti klasickému Viterbiho algoritmu na HMM.

# Kapitola 3

# Zmeny v programe ExonHunter

Našou úlohou bolo integrovať nové programy spracúvajúce externé dáta do programu ExonHunter. Cieľom bolo predovšetkým skrátenie času potrebného na spracovanie externých zdrojov dát a teda urýchlenie celého programu.

## 3.1 Spracovanie EST sekvencií

V doterajšej implementácii spracovania EST dát bol výpočet zarovnania realizovaný programom Sim4. Výpočet zarovnania však často trval dlhšie ako samotná predikcia skrytým Markovovým modelom. Pridali sme teda možnosť použitia programu BLAT.

Ťažiskom implementácie boli funkcie, ktoré spustili program BLAT s príslušnými parametrami a po jeho skončení previedli výstup z formátu PSL do formátu GTF [UCSC, 2008]. Oba formáty sú textové súbory organizované v stĺpcoch oddelených tabulátormi. Hlavný rozdiel spočíva v tom, že zatiaľ čo v PSL súbore každý riadok zodpovedá zarovnaniu celej EST sekvencie ku genómu, riadok v GTF súbore označuje čiastkové zarovnanie exónu, takže zarovnanie jednej EST sekvencia spravidla znamená niekoľko riadkov. Podstatne väčšiu časť práce nám zabrala analýza výsledkov upraveného programu a optimálne nastavenie parametrov.

## 3.2 Spracovanie proteínových sekvencií

Podobne ako v prípade EST sekvencií, aj v prípade proteínových sekvencií doteraz používaný program BLASTx (jedne z variantov algoritmu BLAST na zarovnanie proteínových a DNA sekvencií) nevyhovoval kvôli času behu.

Opäť sme za náhradu zvolili program BLAT, ktorý dokáže zarovnať aj DNA k proteínu. Navyše sme pridali aj program Exonerate, ktorý dokáže na základe zarovnania predikovať miesta zostrihu, takže sme očakávali spresnenie výslednej predikcie. Aj v tomto prípade spočívala práca v konverzii formátov a nastavení parametrov.

## 3.3 Výsledky

V nasledujúcom texte považujeme za *správne predikovaný* prvok (exón, gén) taký prvok, ktorého predikovaná pozícia sa presne zhoduje so sktočnou pozíciou v sekvencii (podľa referenčnej anotácie). Za *nesprávne predikovaný* považujeme taký prvok, ktorého pozícia a typ v predikcii nezodpovedá žiadnemu prvku daného typu v referenčnej anotácii. Za *nepredikovaný* považujeme taký prvok, ktorého pozícia a typ v referenčnej anotácii nezodpovedá žiadnemu prvku daného typu v predikovanej anotácii.

Používame štandardnú definíciu *senzitivity*

$$\text{senzitivita} = \frac{\#\text{ správne predikovaných}}{\#\text{ správne predikovaných} + \#\text{ nepredikovaných}} \tag{3.1}$$

a alternatívnu definíciu *špecificity*

$$\text{špecificita} = \frac{\#\text{ správne predikovaných}}{\#\text{ správne predikovaných} + \#\text{ nesprávne predikovaných}}. \tag{3.2}$$

Tieto miery sú štandardne používané na vyhodnotenie presnosti predikcií v hľadaní génov [Burset and Guigó, 1996]. Na samotné vyhodnotenie používame program Eval [Keibler and Brent, 2003]. Všetky výpočty opisované v nasledujúcich sekciách boli realizované na počítačoch s 16 procesormi Intel Xeon E5520 2.27GHz a 16 GB operačnej pamäte.

Obr. 3.1: *Drosophila melanogaster*. [FlyBase, 2006]

### 3.3.1 Trénovacie a testovacie dáta

Upravený programe sme testovali na dátach mušky *Drosophila melanogaster* (obr. 3.1). Drozofila je dôležitý modelový organizmus v genetike a preto je jej genóm pomerne dobre osekvenovaný a oanotovaný.

Použili sme DNA sekvencie sekvenované a anotované projektom Berkeley Drosophila Genome Project [UCSC, 2006]. Stiahnuté sekvencie mali dohromady 44 MB a obsahovali 5164 génov. Rozdelili sme ich na 2MB úseky a z nich náhodne vybrali trénovacie (28 MB, 3368 génov) a testovacie (16 MB, 1796 génov) sekvencie.

Z projektu DFCI [DFCI, 2006] sme získali EST sekvencie *D. melanogaster* (62 MB, 42538 sekvencií) a z databázy GenBank EST sekvencie príbuzných druhov *D. simulans* (49 MB, 118742 sekvencií) a *D. pseudoobscura* (36 MB, 35042 sekvencií).

Proteínové sekvencie sme stiahli z databázy NCBI Protein. Podľa evolučnej vzdialenosti od *D. melanogaster* sme ich rozdelili do dvoch skupín. Bližšia skupina obsahovala druhy *D. melanogaster, D. simulans* (34 MB, 67893 sekvencií); a vzdialenejšia skupina obsahovala druhy *D. pseudoobscura, Anopheles gambiae, Bombyx mori* (15 MB, 30712 sekvencií).

### 3.3.2 Spracovanie EST sekvencií

Pri spracovaní EST dát ExonHunter najskôr spustí program na zarovnanie EST sekvencií k DNA sekvencii, potom na základe tohoto zarovnania vhodné miesta označí ako intróny (v určitej vzdialenosti medzi dvomi exónmi) a vyfiltruje „zlé“ zarovnania (napr. ak majú nízke skóre zarovnania, ak je zarovnanie príliš krátke, ak si dve časti zarovnania odporujú, alebo ak sa v zarovnaniach nestriedajú exóny a intróny). Predikcia HMM je ovplyvnená len vyfiltrovanými dátami.

Tabuľka 3.1 ukazuje presnosť zarovania exónov podľa EST dát z *D. melanogaster*. Zarovnaní programu BLAT sa vyfiltruje podstatne menej ako zarovnaní Sim4. Zároveň pokrývajú viac z testovacej DNA sekvencie.

Celková výsledná predikcia aj podľa EST dát z ostatných druhov je v

Tabuľka 3.1: Presnosť zarovnania exónov z EST *D. melanogaster*. Exónov v anotácií bolo 8624.

| | Sim4 | Sim4, filtrované | BLAT | BLAT, filtrované |
|---|---|---|---|---|
| Senzitivita | 84.49% | 69.07% | 86.46% | 79.80% |
| Špecificita | 22.19% | 58.99% | 58.96% | 65.63% |
| Počet zarovnaní | 51274 | 13298 | 19997 | 14967 |
| Pokrytie | 31.15% | 26.06% | 30.27% | 27.13% |

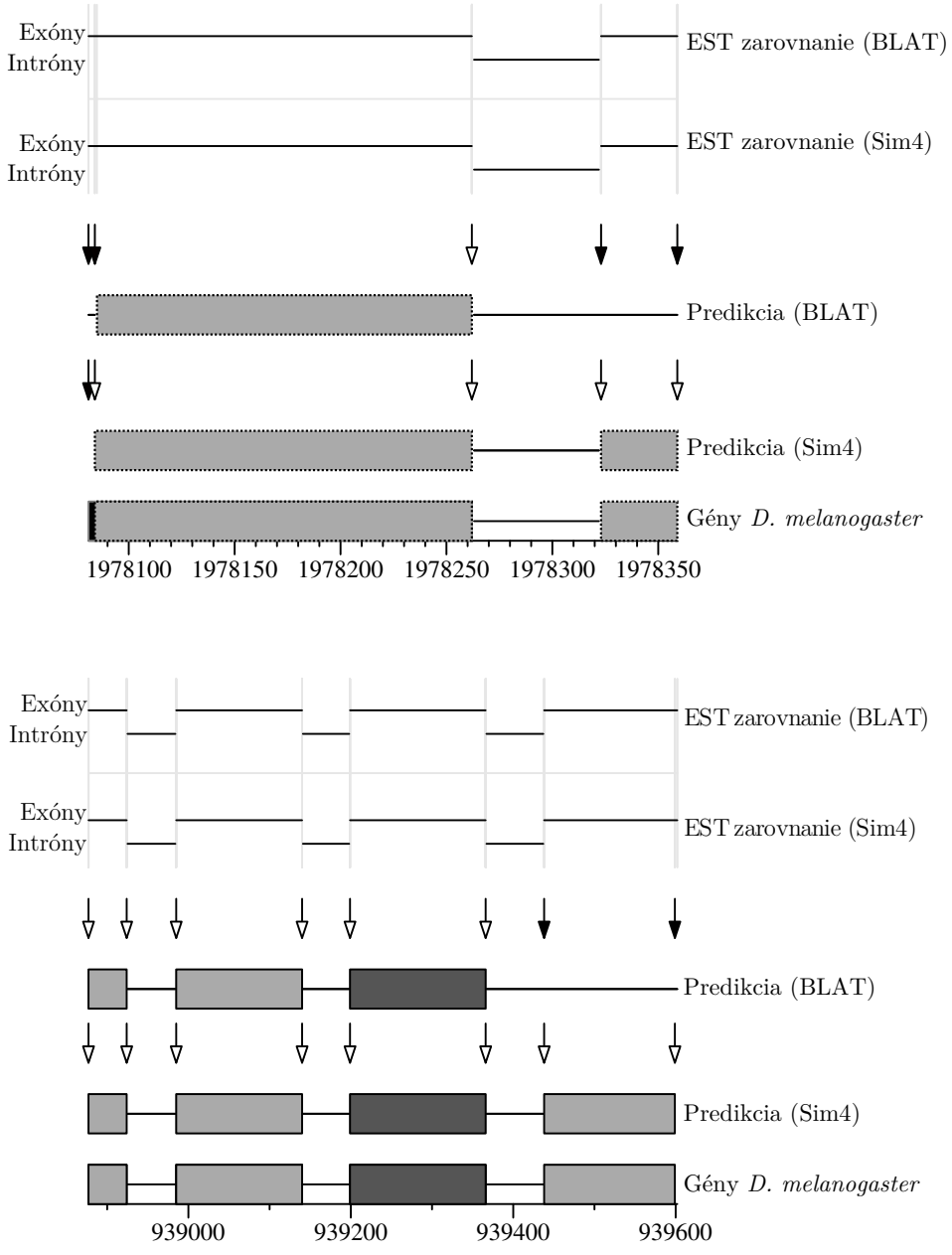
Tabuľka 3.2: Presnosť predikcií s využitím EST dát. Všetky EST zahŕňajú EST z *D. melanogaster, D. simulans a D. pseudoobscura*.

| | | *D. melanogaster* | | Všetky EST | |
|---|---|---|---|---|---|
| | bez EST | Sim4 | BLAT | Sim4 | BLAT |
| Gény Senzitivita | 42.43% | 54.40% | 50.84% | 55.62% | 53.45% |
| Gény Špecificita | 49.16% | 50.81% | 49.40% | 48.17% | 48.41% |
| Exóny Senzitivita | 72.14% | 77.92% | 77.70% | 78.31% | 78.25% |
| Exóny Špecificita | 73.92% | 73.87% | 73.61% | 71.20% | 71.77% |
| Čas behu (min.) | 121 | 343 | 143 | 1255 | 227 |

tabuľke 3.2. Ukazuje sa, že lepšie EST zarovnania programu BLAT neviedli k lepšej predikcii génov, hoci rozdiely nie sú veľmi veľké. Pre zarovnania programu BLAT totiž nevieme určiť správnu orientáciu zarovnania (nevieme s istotou povedať v ktorom smere je zarovnaný dotaz a v ktorom databáza, takže nevieme určiť vlákno, na ktorom je gén). V niektorých prípadoch sa potom stane, že hoci zarovnania oboch programov majú totožné súradnice a dĺžky, správna predikcia vzíde len zo zarovnania s definovanou orientáciou – z programu Sim4 (obr. 3.2).

Zároveň sa ukázalo, že zarovnania EST sekvencií ďalších druhov viedli k vyššiemu počtu správne predikovaných génov a exónov, avšak väčšou mierou prispeli k počtom chybne predikovaných génov a exónov. To je spôsobené tým, že programy Sim4 a BLAT nie sú navrhnuté na zarovnávanie sekvencií z rôznych organizmov (takéto zarovnania obsahujú veľké množstvo inzercií a delécií) a pri medzidruhovom zarovnaní produkujú viac chýb. Napríklad v prípade programu BLAT prešlo filtráciou ďalších 36 654 zarovnaní, z ktorých bolo presne zarovnaných exónov len 3253; čo viedlo k zvýšeniu počtu exónov predikovaných skrytým Markovovým modelom o 266, z ktorých bolo správne predikovaných len 42.

Obr. 3.2: Vizualizácia predikcie génov s využitím EST dát z *D. melanogaster*. V oboch prípadoch boli súradnice exónov a intrónov podľa EST sekvencií u oboch programov totožné, avšak zarovnanie programu BLAT nedefinovalo orientáciu. Dôsledkom je chýbajúci pravý exón vo výslednej predikcii. Podobných prípadov sme zaznamenali niekoľko. Biele šípky ukazujú správne určené a čierne šípky nesprávne určené (resp. chýbajúce) súradnice exónov. Vizualizácia bola získaná programom Mikroskop [Brejová and Vinař, 2008].



Tabuľka 3.3: Počet a pokrytie kódujúcich úsekov zo zarovnaní proteínových sekvencií *D. melanogaster* a *D. simulans*. Znak # označuje počet; dĺžka je súčet dĺžok všetkých zarovnaných úsekov.

| | BLASTx | BLASTx filtrované | BLAT | BLAT filtrované |
|---|---|---|---|---|
| # zarovnaní | 36258 | 25538 | 61777 | 53159 |
| Dĺžka | 26295951 | 11268973 | 87598103 | 10350767 |
| Pokrytie | 13.44% | 9.28% | 66.54% | 14.3% |

Najdôležitejším výsledkom je výrazné skrátenie času potrebného na spracovanie EST sekvencií použitím programu BLAT. Rozdiel v rýchlosti bol ešte výraznejší pri zarovnávaní menej podobných EST dát z ostatných druhov drozofily.

Keďže pokles presnosti výslednej predikcie nebol výrazný, ale ušetrený čas bol značný, pri budúcich aplikáciách programu ExonHunter sa bude na zarovnanie EST sekvencií používať program BLAT.

### 3.3.3 Spracovanie proteínových sekvencií

Po zarovnaní proteínových sekvencií sa podobne ako v prípade EST sekvencií odstránia nevyhovujúce zarovnania; navyše sa zarovnania obsahujúce stop kodón označia ako pseudogény, vyznačia sa intróny, štart a stop kodóny na krajoch zarovnaní a medzigénové úseky. Takto upravené dáta ovplyvňujú predikciu HMM.

Tabuľka 3.3 obsahuje vybrané štatistiky získaných zarovnaní. Program BLAT nájde takmer dvakrát toľko zarovnaní ako program BLASTx. Na druhej strane, pokrývajú skoro presne dve tretiny celej sekvencie, čo už na prvý pohľad vzbudzuje podozrenie, že v zarovnaniach je veľké množstvo náhodných zhôd. Túto domnieku potvrdzuje štatistika vyfiltrovaných zarovnaní. Počet zarovnaní filtrovaním klesol približne o 14%, zatiaľ čo súhrnná dĺžka o viac ako 88% a pokrytie všetkých testovacích DNA sekvencií o viac ako 52%. Teda odfiltrovali sa hlavne tie zarovnania na tých úsekoch, kde sa zarovnalo menej sekvencií a zostali akési ostrovy úsekov s veľkým počtom zarovnaných proteínových sekvencií.

Ukážka použitia externých proteínových dát je na obrázku 3.3. Výslednú predikciu uvádzame v tabuľke 3.4. Použitím programu BLAT sme opäť významne skrátili čas výpočtu. Navyše, výsledné predikcie sú presnejšie. To je

Tabuľka 3.4: Presnosť predikcií s využitím proteínových sekvencií.

| | | Blízke | | Všetky | |
|---|---|---|---|---|---|
| | bez | BLASTx | BLAT | BLASTx | BLAT |
| Gény Senzitivita | 42.43% | 50.17% | 66.48% | 53.45% | 66.26% |
| Gény Špecificita | 49.16% | 52.81% | 59.21% | 53.36% | 58.95% |
| Exóny Senzitivita | 72.14% | 75.02% | 80.48% | 76.34% | 80.39% |
| Exóny Špecificita | 73.92% | 76.80% | 79.84% | 76.67% | 79.40% |
| Čas behu (min.) | 121 | 1017 | 197 | 1240 | 215 |

v súlade so štatistikami v tabuľke 3.3, kde je uvedené, že zarovnania z programu BLAT pokrývajú približne o 5% viac úsekov testovacích DNA sekvencií ako zarovnania programu BLASTx.

Zaujímavé je porovnanie výsledkov použitia proteínov iba z blízkych druhov a zo všetkých druhov. Zatiaľ čo v prípade programu BLASTx znamená použitie všetkých proteínov zvýšenie senzitivity (génov a exónov) aj špecificity (len génov), v prípade zarovnaní programom BLAT dodatočné zarovnania presnosť výslednej predikcie takmer nemenia.

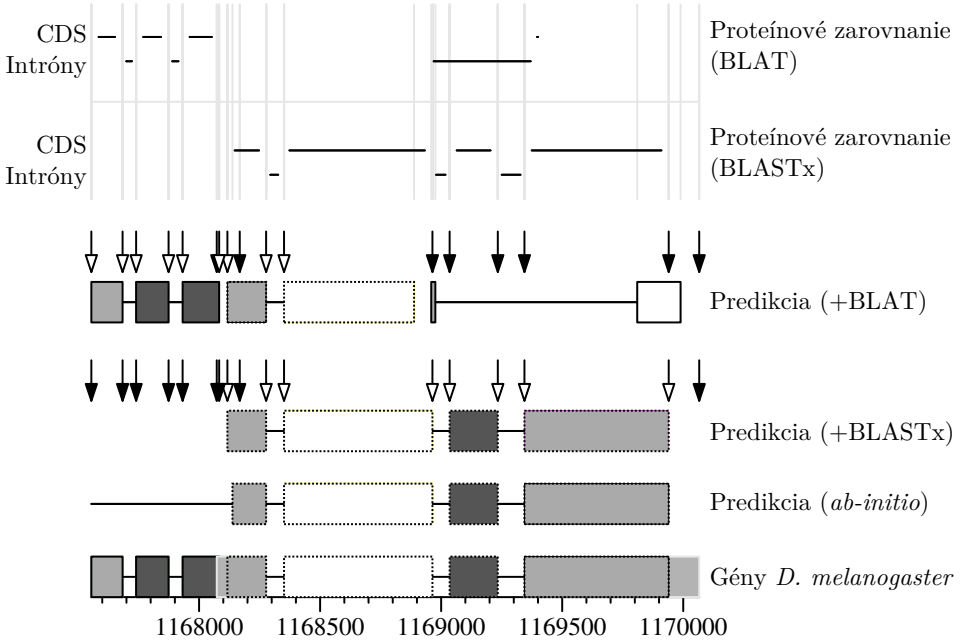
Na druhej strane, neuspeli sme pri pokuse dosiahnuť ešte lepšie výsledky predikovaním signálov zostrihu pomocou programu Exonerate. Po spustení s východzími parametrami na dátach z blízkych druhov program bežal viac ako 24 hodín a ešte pred dopočítaním kompletného zarovnania sme ho ukončili, keďže taký dlhý čas behu nie je prijateľný pre praktické použitie. Následne sme sa snažili lepšie nastaviť parametre modelu a obmedziť počítané zarovnania len na tie najkvalitnejšie avšak opäť sme nezískali uspokojivé výsledky. Po viac ako piatich hodinách bolo zarovnaných len 262 úsekov, čo v konečnom dôsledku len nepatrne zlepšilo senzitivitu génov o 0.05% a špecificitu génov o 0.07% oproti *ab-initio* predikcii (celkový počet správne predikovaných génov zostal nezmenený).

Ďalej sme sa použitiu programu Exonerate na predikciu miest zostrihu nevenovali. V budúcnosti by cesta k zlepšeniu mohla viesť cez viacstupňové zarovnanie, kedy by sme rýchlym zarovnaním (napríklad programom BLAT) z množiny EST sekvencií odstránili tie, ktoré nemajú ani jedno zarovnanie (resp. majú len nekvalitné zarovnania) a potom by sme zvyšné sekvencie zarovnali ešte raz programom Exonerate.

Obr. 3.3: Vizualizácia vplyvu zarovnaní proteínových sekvencií *D. melanogaster* a *D. simulans* na predikciu.

*Ab-initio* predikcia (bez externých dát) identifikuje niektoré exóny. Program BLAT zarovná niekoľko kódujúcich sekvencií na začiatku zobrazeného úseku (ľavá strana obrázku) a zároveň nenájde zarovnanie s proteínom na konci (pravá strana). Naopak, program BLASTx zarovná kódujúce úseky od stredu po koniec úseku. Predikcie skrytého Markovovho modelu sa v oboch prípadoch riadia externými dátami, čo v prípade programu BLAT vedie k správnej predikcii exónov na začiatku a nesprávnej predikcii na konci zobrazeného úseku. V prípade programu BLASTx sa výsledná predikcia od *ab-initio* predikcie líši len v prvom predikovanom exóne, ktorého začiatok sa mierne posunul na správnu pozíciu.

Biele šípky ukazujú správne určené a čierne šípky nesprávne určené (resp. chýbajúce) súradnice exónov. CDS je skratka pre coding sequence, sekvenciu kódujúcu proteín. Vizualizácia bola získaná programom Mikroskop [Brejová and Vinař, 2008].



# Kapitola 4

# Porovnanie vybraných programov

Pre dotvorenie celkovej predstavy o úspešnosti predikcií z predchádzajúcej kapitoly sme porovnali ExonHunter s programami Augustus, GeneID a GeneMark.

Všetky štyri programy sú založené na skrytých Markovových modeloch. Všetky umožňujú *ab-initio* predikciu. Tri z nich (ExonHunter, Augustus a GeneID) sme porovnali aj s použitím externých dát – EST sekvencií a proteínových sekvencií.

## 4.1 Dáta

Na porovnanie sme použili rovnaké DNA a EST aj proteínové sekvencie ako pri testovaní v predchádzajúcej kapitole. *Drosophila melanogaster* je tradičným modelovým organizmom, takže potrebné parametre modelov autori distribuujú priamo so samotným softvérom. Museli sme však overiť, či trénovacia množina niektorého z programov neobsahuje nejakú časť našej testovacej množiny, čo by mohlo umelo zvýšiť nameranú presnosť v porovnaní s programami, ktoré používajú rôzne sekvencie na trénovanie a testovanie.

Autori programu Augustus uvádzajú dva súbory sekvencií *D. melanogaster* (320 a 400 génov) použité pri trénovaní [Stanke and Waack, 2003a]. Zistili sme, že 84 génov je spoločných s testovacou množinou, čo je menej ako 5% celkového počtu génov v testovacej množine.

Podobne sme analyzovali trénovaciu množinu programu GeneID [Parra

Tabuľka 4.1: Porovnanie *ab-initio* predikcií.

| | ExonHunter | GeneMark | Augustus | GeneID |
|---|---|---|---|---|
| Gény Senzitivita | 42.43% | 46.16% | 49.50% | 32.35% |
| Gény Špecificita | 49.16% | 37.29% | 51.99% | 32.88% |
| Exóny Senzitivita | 72.14% | 71.53% | 69.12% | 65.58% |
| Exóny Špecificita | 73.92% | 64.11% | 79.81% | 66.08% |
| Čas behu (min.) | 121 | 95 | 34 | <1 |

et al., 2000a]. Opäť sme zistili nepatrný prienik, tentoraz 40 génov, čo je menej ako 2.5% počtu génov v našej testovacej množine.

Keďže program GeneMark používa iteratívne učenie bez učiteľa, zisťovanie prieniu nebolo potrebné. Zistené prieniky neboli príliš veľké, preto sme nepovažovali za potrebné upraviť testovaciu sekvenciu.

## 4.2 Výsledky

### 4.2.1 *Ab-initio* predikcia

Výsledky *ab-initio* predikcií sú v tabuľke 4.1. Najpresnejšie výsledky dosiahol program Augustus, keď úspešne označil v podstate každý druhý gén. Zároveň na výpočet potreboval relatívne krátky čas. Program ExonHunter bol najsenzitívnejší na úrovni exónov, no bol zároveň aj najpomalší.

Odlišný skórovací model programu GeneID ukázal, že hoci dokáže predikcie počítať veľmi rýchlo, jeho presnosť na úrovni génov za ostatnými programami zaostáva. Pritom rozdiely na úrovni exónov nie sú také výrazné, čo by mohlo znamenať nedostatky vo fáze zostavovania génov z exónov.

Výsledky programu GeneMark ukazujú, že aj modely s iteratívnym učením bez učiteľa dokážu relatívne dobre predikovať gény. Senzitivita je porovnateľná s programami ExonHunter a GeneID, avšak mnohé z predikovaných génov v skutočnosti génmi nie sú. Veľkou výhodou programu GeneMark je predikcia bez trénovacích dát, čo z neho robí jedinečný nástroj pri anotáciách novosekvenovaných genómov, kde by sa štandardné programy nemohli aplikovať.

Tabuľka 4.2: Porovnanie predikcií s EST dátami *D. melanogaster*.

| | EH+Sim4 | EH+BLAT | Augustus | GeneID |
|---|---|---|---|---|
| Gény Senzitivita | 54.40% | 50.84% | 62.92% | 25.17% |
| Gény Špecificita | 50.81% | 49.40% | 52.64% | 16.80% |
| Exóny Senzitivita | 77.92% | 77.70% | 79.16% | 66.18% |
| Exóny Špecificita | 73.87% | 73.61% | 74.37% | 53.77% |
| Čas behu (min.) | 343 | 143 | 216 | 103 |

### 4.2.2 Predikcia s EST dátami

Tri programy deklarujú možnosť použitia externých dát. Použitie externých dát v programe ExonHunter sme opísali v predchádzajúcej kapitole. Externé dáta pre programy Augustus a GeneID sme spracovali odlišným spôsobom.

Softvérový balíček programu Augustus obsahuje aj utilitu na spracovanie zarovnania programu BLAT do vhodnej podoby pre program Augustus, takže spracovanie EST dát bolo v tomto prípade s použitím programu BLAT priamočiare.

Opačná situácia bola v prípade programu GeneID. Ten vyžaduje externé informácie v podrobnejšej forme ako ExonHunter a Augustus. Keď sme sa pokúsili použiť tie dáta, ktoré vzišli zo spracovania EST sekvencií pre program ExonHunter (oba programy akceptujú externé dáta vo formáte GTF), GeneID pri predikcii tieto dáta úplne ignoroval (výsledná predickia bola identická ako *ab-initio* predikcia). Dôvodom je, že v informáciách o pravdepodobných zarovnaniach exónov vyžaduje uvedené aj poradie exónu v géne (tj. označenie exónu ako prvý exón v génu, vnútorný exón génu, posledný exón génu alebo jednoexónový gén). Táto informácia sa však zo zarovnaní EST dát ťažko určuje. Keďže zarovnanie EST dát programom BLAT neprinieslo očakávaný výsledok, pokúsili sme sa zarovnať EST sekvencie programom Exonerate, ktorý poradie exónov v rámci génu uvádza (hoci ide v skutočnosti len o odhad, keďže EST sekvencie sú len fragmenty mRNA a nevieme presne povedať, či ich začiatok zodpovedá začiatku génu).

Výsledky uvádzame v tabuľkách 4.2 a 4.3. Program Augustus, podobne ako oba varianty programu ExonHunter, ukázal pri zarovnaní EST z *D. melanogaster* zvýšenie senzitivity pri zachovanej úrovni špecificity (oproti *ab-initio* predikcii). To znamená, že väčšina z dodatočne predikovaných génov a exónov bola určená správne.

Tabuľka 4.3: Porovnanie predikcií so všetkými EST dátami.

| | EH+Sim4 | EH+BLAT | Augustus | GeneID |
|---|---|---|---|---|
| Gény Senzitivita | 55.62% | 53.45% | 60.13% | 6.51% |
| Gény Špecificita | 48.17% | 48.41% | 43.82% | 3.69% |
| Exóny Senzitivita | 78.31% | 78.25% | 77.87% | 49.32% |
| Exóny Špecificita | 71.20% | 71.77% | 68.14% | 37.16% |
| Čas behu (min.) | 1255 | 227 | 456 | 282 |

Pri dodaní EST dát aj zo vzdialenejších organizmov senzitivita programu Augustus klesla (v porovnaní s výsledkami pri použití EST len z *D. melanogaster*). Súčasne klesla aj špecificita a to až pod úroveň špecificity *ab-initio* predikcie. To môžeme vysvetliť tým, že nových externých informáciách boli nesprávne idetifikované exóny (ako sme spomenuli v druhej kapitole, program BLAT nie je navrhnutý pre medzidruhové zarovnania), ktoré negatívne ovplyvnili predikciu. Zároveň zahrnutie nových externých dát znížilo celkový počet správne predikovaných génov (zarovnania z rôznych druhov si odporovali, takže HMM ich pri predikcii ignoroval).

Osobitým prípadom je predikcia programu GeneID. Ukázalo sa, že ani presnejšie definované exóny zo zarovnaní nepriniesli úžitok a práve naopak, výrazne zhoršili celkovú predikciu. Je však možné, že ak by autori sami poskytli program, ktorý by zarovnania z najčastejšie používaných zarovnávacích programov transformoval do vhodnej podoby, predikcia programu GeneID by sa mohla zlepšiť.

Výsledky navyše úkázali, že pri zarovnaní EST dát programom BLAT v programoch ExonHunter a Augustus je celkový čas potrebný na predikciu kratší v prípade programu ExonHunter, hoci pri *ab-initio* predikcii to bolo naopak. To svedčí o tom, že použitý model spracovania externých informácií je v programe ExonHutner pomerne efektívny (najväčšiu porciu času v spracovaní EST dát zaberá samotné zarovnanie). Časy uvádzané v tabuľkách 4.2 a 4.3 sú súčtom času potrebného na spracovanie EST dát a na samotnú predikciu. Keďže program Exonerate potrebuje na zarovnanie relatívne dlhý čas, celkový čas potrebný na predikciu programu GeneID s externými informáciami je omnoho dlhší ako čas potrebný na *ab-initio* predikciu (samotná predikcia trvala aj v prípade použitia externých dát menej ako minútu).

Tabuľka 4.4: Porovnanie predikcií s proteínovými sekvenciami *D. melanogaster* a *D. simulans*.

| | EH+BLASTx | EH+BLAT | Augustus |
|---|---|---|---|
| Gény Senzitivita | 50.17% | 66.48% | 50.17% |
| Gény Špecificita | 52.81% | 59.21% | 47.98% |
| Exóny Senzitivita | 75.02% | 80.48% | 68.62% |
| Exóny Špecificita | 76.80% | 79.84% | 74.41% |
| Čas behu (min.) | 1017 | 197 | 198 |

Tabuľka 4.5: Porovnanie predikcií so všetkými proteínovými sekvenciami.

| | EH+BLASTx | EH+BLAT | Augustus |
|---|---|---|---|
| Gény Senzitivita | 53.45% | 66.26% | 49.39% |
| Gény Špecificita | 53.36% | 58.95% | 44.82% |
| Exóny Senzitivita | 76.34% | 80.39% | 68.00% |
| Exóny Špecificita | 76.67% | 79.40% | 71.91% |
| Čas behu (min.) | 1240 | 215 | 189 |

### 4.2.3 Predikcia s proteínovými sekvenciami

Keďže spracovanie externých dát v programe GeneID sa ukázalo ako problémové, predikcie s proteínovými dátami sme porovnávali len pre ExonHunter a Augustus.

Proteínové sekvencie pre Augustus sme k DNA zarovnali rovnako ako EST sekvencie, programom BLAT a rovnako sme ich aj spracovali príslušným programom z balíčka Augustus.

Výsledky pre blízke aj všetky proteíny sú v tabuľkách 4.4 a 4.5. Na rozdiel od EST sekvencií je tentokrát výraznejšie presnejší program ExonHunter so zarovnaním z BLAT-u. Presnosť pri použití zarovnaní z programu BLASTx je tiež mierne lepšia, no potrebný čas na zarovnanie omnoho dlhší. ExonHunter je úspešnejší aj pri predikovaní s použitím všetkých dát, hoci tu už má časovo mierne navrch Augustus. Pri použití všetkých proteínov sa zhoršila celková predikcia programu Augustus, čo sme už pozorovali pri programe ExonHunter v predchádzajúcej kapitole.

# Kapitola 5

# Záver

Prvoradým cieľom našej práce bolo integrovať nové programy na zarovnanie EST sekvencií a zarovnanie proteínových sekvencií do programu ExonHunter. V oboch prípadoch sme použili program BLAT.

Testovanie na DNA sekvenciách z *D. melanogaster* ukázalo, že nová verzia programu je v oboch prípadoch výrazne rýchlejšia ako doterajšia verzia. V prípade spracovania EST sekvencií sme zaznamenali mierny pokles presnosti výslednej predikcie skrytého Markovovho modelu, spôsobený absenciou informácie o orientácií zarovnania, ktorá sa z výstupu programu BLAT nedá získať. Pri proteínových sekvenciách sme, naopak, zaznamenali aj zlepšenie výslednej predikcie.

Na druhej strane, nepodarilo sa nám uspokojivo zahrnúť do predikcie informácie o miestach zostrihu z proteínových zarovnaní programom Exonerate. Buď bol čas behu programu Exonerate príliš dlhý, alebo, pri obmedzení na nájdenie potenciálne najkvalitnejších zarovnaní, bolo nájdených zarovnaní príliš málo na to, aby dokázali výraznejšie ovplyvniť celkovú predikciu programu ExonHunter.

Výkon pôvodnej aj upravenej verzie programu ExonHunter sme porovnali s vybranými programami na hľadanie génov, založenými na skrytých Markovových modeloch. *Ab-initio* porovnanie (bez externých dát) ukázalo, že presnosť predikcii programu ExonHunter je len o málo menšia ako presnosť najlepšieho programu v porovnaní (Augustus). Zároveň bol program ExonHunter pri *ab-initio* predikicii najpomalší. Zistili sme, že aj pri použití EST dát z *D. melanogaster* je program Augustus presnejší, hoci tu už nie je rýchlejší ako upravená verzia programu ExonHunter. Pri využití zarovnaní

ďalších EST sekvencií z druhov *D. simulans* a *D. pseudoobscura* sú už rozdiely v presnosti menšie a ExonHunter je takmer dvakrát rýchlejší. Oproti programu Augustus je ExonHunter presnejší pri predikciách s využitím proteínových dát.

Hlavné ciele našej práce sa nám podarilo splniť a to znamená, že pri budúcom bioinformatickom výskume budeme používať novú verziu programu ExonHunter.

# Literatúra

[Altschul et al., 1990] Altschul, S. F., Gish, W., Miller, W., Myers, E. W., and Lipman, D. J. (1990). Basic Local Alignment Search Tool. *Journal of Molecular Biology*, 215:403–410.

[Altschul et al., 1997] Altschul, S. F., Madden, T. L., Schäffer, A. A., Zhang, J., Zhang, Z., Miller, W., and Lipman, D. J. (1997). Gapped BLAST and PSI-BLAST: a new generation of protein database searching programs. *Nucleic Acids Research*, 25(17):3389–3402.

[Brejová et al., 2005] Brejová, B., Brown, D. G., Li, M., and Vinař, T. (2005). ExonHunter: a comprehensive approach to gene finding. *Bioinformatics*, 21(S1):i57–i65. Intelligent Systems for Molecular Biology (ISMB 2005).

[Brejová and Vinař, 2002] Brejová, B. and Vinař, T. (2002). A better method for length distribution modeling in HMMs and its application to gene finding. In Apostolico, A. and Takeda, M., editors, *Combinatorial Pattern Matching, 13th Annual Symposium (CPM),*, volume 2373 of *Lecture Notes in Computer Science*, pages 190–202, Fukuoka, Japan. Springer.

[Brejová and Vinař, 2008] Brejová, B. and Vinař, T. (2008). Mikroskop, flexible tool for creating diagrams of sequence annotations. Online. `http://www.bioinformatics.uwaterloo.ca/downloads/mikroskop/`.

[Burset and Guigó, 1996] Burset, M. and Guigó, R. (1996). Evaluation of gene structure prediction programs. *Genomics*, 34(3):353–367.

[Chao et al., 1997] Chao, K.-M., Zhang, J., Ostell, J., and Miller, W. (1997). A tool for aligning very similar DNA sequences. *Bioinformatics*, 13(1):75–80.

[DFCI, 2006] DFCI (2006). DFCI gene index for Drosophila Release 11.0. Online. `http://compbio.dfci.harvard.edu/cgi-bin/tgi/gimain.pl?gudb=drosoph`.

[Dolgin, 2009] Dolgin, E. (2009). Human genomics: The genome finishers. *Nature*, 462:843–845.

[Durbin et al., 1998] Durbin, R., Eddy, S., Krogh, A., and Mitchison, G. (1998). *Biological sequence analysis*. Cambridge University Press.

[Florea et al., 1998] Florea, L., Hartzell, G., Zhang, Z., Rubin, G., and Miller, W. (1998). A Computer Program for Aligning a cDNA Sequence with a Genomic DNA Sequence. *Genome Res.*, 8(9):967–974.

[FlyBase, 2006] FlyBase (2006). Patterson (1943) Plate V Drosophila melanogaster male. Online.
http://flybase.org/reports/FBim0000487.html.

[Guigó, 1998] Guigó, R. (1998). Assembling genes from predicted exons in linear time with dynamic programming. *Journal of Computational Biology*, 5:681–702.

[Heinkoff and Heinkoff, 1992] Heinkoff, S. and Heinkoff, J. G. (1992). Amino acid substitution matrices from protein blocks. *Proceedings of the National Academy of Sciences of the USA, Biochemistry*, 89:10915–10919.

[Keibler and Brent, 2003] Keibler, E. and Brent, M. R. (2003). Eval: A software package for analysis of genome annotations. *BMC Bioinformatics*, 4:50.

[Kent, 2002] Kent, W. J. (2002). BLAT: The BLAST Like Alignment Tool. *Genome Res.*, 12:656–664.

[Lomsadze et al., 2005] Lomsadze, A., Ter-Hovhanisyan, V., and Chernoff, Y. (2005). Gene identification in novel eucaryotic genomes by self-training algorithm. *Nucleic Acids Research*, 33, No. 20:6494–6506.

[Nagaraj et al., 2006] Nagaraj, S. H., Gasser, R. B., and Ranganathan, S. (2006). A hitchhiker's guide to expressed sequence tag (EST) analysis. *Briefings in Bioinformatics*, 8(1).

[NCBI, a] NCBI. GenBank, The National Center for Biotechnology Information. Online. `http://www.ncbi.nlm.nih.gov/genbank/`.

[NCBI, b] NCBI. NCBI Protein, The National Center for Biotechnology Information. Online. `http://www.ncbi.nlm.nih.gov/protein`.

[NCBI, c] NCBI. RefSeq, The National Center for Biotechnology Information. Online. `http://www.ncbi.nlm.nih.gov/refseq/`.

[Needleman and Wunsch, 1970] Needleman, S. B. and Wunsch, C. D. (1970). A general method applicable to the search for similarities in the amino acid sequence of two proteins. *Journal of Molecular Biology*, 48:443–453.

[Parra et al., 2000a] Parra, G., Blanco, E., and Guigó, R. (2000a). Datasets: GeneID in Drosophila. Online. `http://genome.crg.es/datasets/Dro_me/`.

[Parra et al., 2000b] Parra, G., Blanco, E., and Guigó, R. (2000b). GeneID in Drosophila. *Genome Research*, 10:511–515.

[Rabiner, 1989] Rabiner, L. (1989). A tutorial on hidden Markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2):257–286.

[Russell, 2010] Russell, P. J. (2010). *iGenetics: A molecular approach*. Third edition. San Francisco: Pearson Education, 2010. 828 s., ISBN: 0-321-61022-9.

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

[Smith and Waterman, 1981] Smith, T. F. and Waterman, M. S. (1981). Identification of Common Molecular Subsequences. *Journal of Molecular Biology*, 147:195–197.

[Stanke and Waack, 2003a] Stanke, M. and Waack, S. (2003a). Augustus: datasets. Online. `http://augustus.gobics.de/datasets/`.

[Stanke and Waack, 2003b] Stanke, M. and Waack, S. (2003b). Gene prediction with a hidden Markov model and new intron submodel. *Bioinformatics*, 19(2):ii215–ii225.

[Ter-Hovhanisyan et al., 2008] Ter-Hovhanisyan, V., Lomsadze, A., Chernoff, Y., and Borodovsky, M. (2008). Gene prediction withc in novel fungal genomes using an ab initio algorithm with unsupervised training. *Genome Research*, 18:1979–1990.

[UCSC, 2006] UCSC (2006). Drosophila melanogaster draft assembly (BDGP Release 5), provided by the Berkeley Drosophila Genome Project (BDGP). Online. `http://genome.ucsc.edu/cgi-bin/hgGateway?clade=insect&org=0&db=0`.

[UCSC, 2008] UCSC (2008). Frequently Asked Questions: Data File Formats. Online. `http://genome.ucsc.edu/FAQ/FAQformat.html`.