UNIVERZITA PAVLA JOZEFA ŠAFÁRIKA V KOŠICIACH

PRÍRODOVEDECKÁ FAKULTA

# Vizualizácia a analýza stochastických modelov RNA reťazcov

Mária Palušáková

2009

UNIVERZITA PAVLA JOZEFA ŠAFÁRIKA V KOŠICIACH

PRÍRODOVEDECKÁ FAKULTA

Ústav informatiky

# Vizualizácia a analýza stochastických modelov RNA reťazcov

Mária Palušáková

Vedúci práce: RNDr. Michal Mati

doc. RNDr. Gabriela Andrejková, CSc.

Košice 2009

# Analytický list

| | |
|---|---|
| Autor: | Mária Palušáková |
| Názov práce: | Vizualizácia a analýza stochastických modelov RNA reťazcov |
| Jazyk práce: | slovenský |
| Počet strán: | 32 |
| Univerzita: | Univerzita Pavla Jozefa Šafárika v Košiciach |
| Fakulta: | Prírodovedecká fakulta |
| Katedra/Ústav: | Ústav informatiky |
| Študijný odbor: | Informatika |
| Študijný program: | Informatika |
| Mesto: | Košice |
| Vedúci práce: | RNDr. Michal Mati<br>doc. RNDr. Gabriela Andrejková, CSc. |
| Dátum odovzdania: | 20. apríl 2009 |
| Dátum obhajoby: | 23. 4. 2009 |
| Kľúčové slová: | Stochastická bezkontextová gramatika, Kovariančný model |
| Názov práce v AJ: | Visualization and analysis of RNA sequences stochastic models |
| Kľúčové slová v AJ: | Stochastic context-free grammar, Covariance model |

**Abstrakt**

V tejto práci sa venujeme popisu stochastických bezkontextových gramatík a kovariančných modelov používaných na modelovanie sekundárnej štruktúry RNA reťazcov. Ako súčasť práce sme zostrojili prehliadač kovariančných modelov, ktorý ich zobrazuje v grafickej forme, ktorá je oproti textovému formátu získaného z Rfam databázy prehľadnejšia.

**Abstract**

In this work did we write about stochastic context-free grammars a covariance models, which are used to model secondary structure of RNA sequences. We have made CMBrowser, that shows covariance models in graphical form, which is much more comprendious than textual form of covariance models from Rfam database.

# Obsah

# Úvod

Bioinformatika je pomerne novým vedeckým smerom. Vznikla len pred 20-30 rokmi, kvôli potrebe spracovávania rozsiahlych biologických dát. Preto sa v prvej kapitole práce venujeme popisu aktuálnych bioinformatických problémov.

V druhej časti sa venujeme jednej z najčastejšie riešených úloh v bioinformatike. Je ňou viacnásobné zarovnávanie sekvencií, ktorým sa začína riešenie mnohých iných problémov.

V poslednej kapitole popisujeme jedno z využití viacnásobného zarovnania, ktorým je zostrojenie kovariančného modelu. Ten slúži na reprezentáciu primárnej a sekundárnej štruktúry DNA a RNA. Dá sa zostrojiť pomocou softvéru INFERNAL, ktorý je voľne dostupný na internete. Jeho príručka veľmi pomohla pochopiť jeho zostrojenie a zdrojové kódy reprezentáciu dát vo vygenerovaných súboroch.

Ako súčasť práce sme zostrojili prehliadač kovariančných modelov, ktorý je priložený ako príloha na CD. Tento slúži na intuitívnejšie grafické zobrazenie kovariančných modelov vytvorených INFERNAL-om. Jeho výzor a ovládanie popisujeme na konci tejto práce.

# 1 Bioinformatika

Bioinformatika vznikla kvôli potrebe spracovávania rozsiahlych biologických informácii, ako je napríklad evolučná história, štruktúra proteínov, DNA a RNA reťazcov. Zaoberá sa využitím informatických technológií predovšetkým v problémoch molekulárnej biológie. Dve rozsiahle časti bioinformatiky sa týkajú analýzy genómov a proteínov. Genómom môže byť nazvaná kompletná množina DNA sekvencií kódujúca dedičný materiál prenášaný z generácie na generáciu. Tieto DNA sekvencie zahŕňajú všetky gény a kópie reťazcov RNA.

V začiatkoch bioinformatiky bolo jej hlavným cieľom vytvorenie a udržiavanie databáz do ktorých by sa ukladali biologické informácie, ako napríklad nukleotidy a reťazce aminokyselín. Tieto databázy mali nielen sprístupniť vedcom dáta, ale mali im umožniť aj vkladanie nových dát. Preto však museli byť stanovené normy pre reprezentáciu týchto dát. Samotný proces analýzy a interpretácie dát je označovaný ako výpočtová biológia. Medzi dnešné bioinformatické databázy patria napríklad GenBank, Rfam, PROSITE, Pfam, či National DNA database. Databázou s bioinformatickými článkami a vedeckou literatúrou je PubMed Central.

Medzi hlavné problémy bioinformatiky a výpočtovej biológie patrí rozvoj a implementácia nástrojov, ktoré umožnia efektívny prístup a využitie rôznych typov informácií. Ďalším riešeným problémom je vývoj nových algoritmov, matematických formúl a štatistík vďaka ktorým by sa dala ohodnotiť príbuznosť medzi jednotlivými členmi rozsiahlych množín dát. Tu patria metódy na hľadanie génov v sekvenciách, predpovedanie proteínovej štruktúry a funkcie a zoskupovanie proteínových sekvencií do rodín obsahujúcich príbuzné sekvencie.

Z konkrétnych problémov, ktoré rieši bioinformatika môžeme spomenúť tieto:

1. **Zarovnávanie sekvencií** – ide o spôsob usporiadania dvoch, alebo viacerých reťazcov DNA, RNA alebo proteínov na identifikáciu podobných častí, ktoré môžu byť dôsledkom funkčnej, štrukturálnej, alebo evolučnej príbuznosti medzi týmito reťazcami. Tento problém a niektoré metódy jeho riešenia je popísaný v kapitole 2 tejto práce.
2. **Vyhľadávanie génov** – týka sa predovšetkým vyhľadávania úsekov sekvencií genotypovej DNA. Medzi tieto úseky patria gény kódujúce rôzne proteíny a RNA gény.

3. **Spájanie genómov** – ide o spájanie množstva krátkych DNA sekvencií na vytvorenie reprezentácie pôvodného chromozómu, z ktorého DNA vznikla. Tieto krátke DNA sekvencie vznikajú na základe projektu „Shotgun sequencing“, ktorý rozdelí DNA na milióny krátkych častí. Tie sú potom načítané pomocou zarovnávacieho stroja. Algoritmus na spájanie genómov tieto sekvencie zarovná. Na miestach, kde sa 2 sekvencie prelínajú sa môžu spojiť dokopy. Tento problém je zložitý pretože genómy obsahujú podreťazce, ktoré sa opakujú.
4. **Zarovnávanie proteínovej štruktúry** – štrukturálne zarovnávanie je forma zarovnávania sekvencií založená na porovnávaní tvaru a trojdimenzionálnej štruktúry.
5. **Predpovedanie proteínovej štruktúry** – ide o predpovedanie trojdimenzionálnej štruktúry proteínov na základe sekvencie aminokyselín. Táto metóda je dôležitá napríklad v medicíne pri navrhovaní liekov, alebo v biotechnológii pri navrhovaní nových enzýmov.
6. **Určovanie funkcie proteínu** – na riešenie tohto problému zatiaľ neexistujú vhodné dáta. Momentálnym cieľom je zostrojiť databázu obsahujúcu reakcie proteínov. Najprv však treba vytvoriť vhodný spôsob ich reprezentácie.
7. **Modelovanie evolučnej histórie** – ide o zoskupovanie sekvencií na základe ich podobnosti do stromu. Takýto fylogenetický strom potom reprezentuje zmeny v sekvenciách počas evolúcie.

# 2 Viacnásobné zarovnávanie sekvencií

V nasledujúcej časti sa budem podrobnejšie venovať viacnásobnému zarovnaniu sekvencií, ktoré je v bioinformatike jednou z najčastejšie riešených úloh. Je prvou fázou riešenia mnohých zložitejších problémov, ako napríklad určovanie sekundárnej štruktúry RNA, konštrukcia fylogenetického stromu, či identifikácia častí DNA alebo proteínov, ktoré súvisia s určitou funkciou.

Zarovnanie dvoch reťazcov, spočíva v ich doplnení o medzery a zapísaní jednotlivých znakov pod seba. Pre nás je dôležité umiestniť medzery na správne miesta tak, aby boli podobné časti zarovnané pod sebou(Obr. 1).

```
q a c _ d b d
q a v x _ b _
```

**Obr. 1 Zarovnanie reťazcov qacdbd a qavxb**

Aby sme mohli nájsť najvhodnejšie zarovnanie potrebujeme ho najprv nejako definovať. Využíva sa na to algoritmus **edit distance**, ktorý zarovnanie ohodnotí podľa počtu a typu zmien potrebných na to, aby sme prvý reťazec zmenili na druhý. Povolené zmeny sú vloženie znaku do prvého reťazca, vymazanie znaku z prvého reťazca a nahradenie znaku v prvom reťazci znakom z druhého reťazca. Každá z týchto zmien má svoje ohodnotenie (obvykle 1, ak s daným znakom nemusíme vykonať žiadnu operáciu pretože sa zhoduje so znakom v druhom reťazci hodnotíme to obvykle 0). Ohodnotenie zarovnania je rovné súčtu všetkých potrebných zmien vynásobených ich ohodnotením.

Za optimálne zarovnanie sa považuje zarovnanie s minimálnym edit distance ohodnotením. Optimálne zarovnanie sa hľadá pomocou dynamického programovania. Pre 2 reťazce $S_1$ a $S_2$ sa vypočíta tabuľka vzdialeností $D$, v ktorej $D(i, j)$ je najmenšie ohodnotenie operácii potrebných na zmenu prvých $i$ znakov $S_1$ na prvých $j$ znakov $S_2$. Na jej výpočet sa používajú tieto základné vzorce:

$$D(i,0) = i$$
$$D(0, j) = j$$

$$D(i, j) = \min\left[D(i-1, j)+1, D(i, j-1)+1, D(i-1, j-1)+t(i, j)\right]$$

Pričom $t(i, j) = 0$ ak $S_1(i) = S_2(j)$, inak $t(i, j) = 1$.

Príklad tabuliek vzdialeností a k nim prislúchajúcich zarovnaní je v Tab. 1 a Obr. 2.

| | | **A** | **B** | **C** | **A** |
|---|---|---|---|---|---|
| | 0 | 1 | 2 | 3 | 4 |
| **A** | 1 | 0 | 1 | 2 | 3 |
| **A** | 2 | 1 | 1 | 2 | 2 |
| **C** | 3 | 2 | 2 | 1 | 2 |
| **A** | 4 | 3 | 3 | 2 | **1** |

| | | **A** | **B** | **C** | **A** |
|---|---|---|---|---|---|
| | 0 | 1 | 2 | 3 | 4 |
| **A** | 1 | 0 | 1 | 2 | 3 |
| **A** | 2 | 1 | 1 | 2 | 2 |
| **B** | 3 | 2 | 1 | 2 | 3 |
| **C** | 4 | 3 | 2 | 1 | **2** |

| | | **A** | **A** | **B** | **C** |
|---|---|---|---|---|---|
| | 0 | 1 | 2 | 3 | 4 |
| **A** | 1 | 0 | 1 | 2 | 3 |
| **A** | 2 | 1 | 0 | 1 | 2 |
| **C** | 3 | 2 | 1 | 1 | 1 |
| **A** | 4 | 3 | 2 | 2 | **2** |

**Tab. 1 Príklad tabuliek tabuliek edit distance pre každú dvojicu z reťazcov AACA, AABC, ABCA**

```
A B C A      A _ B C A      A A B C _
A A C A      A A B C _      A A _ C A
```

**Obr. 2 Zarovnanie reťazcov podľa vzdialenostných tabuliek z Tab. 1**

Viacnásobné zarovnanie(Obr. 3) sa týka zarovnávania viacerých reťazcov. V bioinformatike predovšetkým DNA, alebo RNA reťazcov či proteínových reťazcov, ktoré nazývame sekvenciami.

```
G U U G G U G G U _ U A U U G U G U C G G
G U C G G U G G U _ G U U A G C G G U G G
U A C G G C G G U C A A U A G C G G C A G
U A C G G C G G U C C A U A G C G G C A G
U A C G G C G G C _ C A U A G C G G C A G
U A C G G C G G C _ C A C A G C G G C A G
U A C G G C G G U _ U A U A G C G G U G G
U A C G G C G G C _ C A U A G C G A C A G
U G C G G U G G U _ G A U A G U G G U G G
U G C G G U G G U _ G A U A G U G G U G G
_ G C C U U G G U _ C A C A G C C C C U G
```

**Obr. 3 Príklad viacnásobného zarovnania sekvencií**

Výpočet optimálneho zarovnania je časovo náročný – patrí medzi NP-úplné problémy. Preto sú všetky doterajšie algoritmy založené na heuristike, čo spôsobuje, že riešenie nemusí byť optimálne. Existujúce algoritmy sa rozdeľujú do troch skupín:

1) **Exaktné algoritmy** – používajú prepracované heuristiky, vďaka čomu sú riešenia blízke optimalite. Sú však časovo veľmi náročné, a teda použiteľné len pre malý počet sekvencií (obvykle menej ako 20).
2) **Postupné (progresívne) algoritmy** – konštruujú zarovnanie postupným pridávaním sekvencií k už zarovnaným. Tieto metódy sú najpoužívanejšie predovšetkým kvôli ich rýchlosti a jednoduchej implementácii. Napriek tomu, že sú heuristické poskytujú dostatočne presné výsledky.
3) **Iteratívne algoritmy** – hneď na začiatku vygenerujú zarovnanie, ktoré potom iteratívne vylepšujú. Končia keď už sa riešenie viac spresniť nedá. Môžu byť stochastické, alebo deterministické. Deterministické obvykle pracujú tak, že náhodne vyberú sekvenciu a znova ju zarovnajú.

## 2.1 Exaktné algoritmy

Exaktné algoritmy dokážu skonštruovať najoptimálnejšie viacnásobné zarovnanie. Ohodnotenie viacnásobného zarovnania je však o niečo zložitejší problém ako ohodnotenie zarovnania 2 reťazcov. Jedným z možných riešení je sum-of-pairs(SP) ohodnotenie, ktoré je súčtom ohodnotení jednotlivých dvojíc zarovnaných sekvencií.

Exaktné algoritmy vznikajú na základe dynamického programovania a zovšeobecnenia algoritmov slúžiacich na zarovnávanie 2 sekvencií. Pre n zarovnávaných reťazcov však musia zostrojiť až *n*-rozmerné pole vzdialeností, takže ich zložitosť veľmi rastie so zväčšujúcim sa počtom reťazcov. Zväčša sa dajú použiť na zarovnanie 5 - 20 sekvencií. Patrí tu metóda pre nájdenie minimálneho SP ohodnotenia založená na dynamickom programovaní. Táto metóda má však príliš veľkú zložitosť. Pri k sekvenciách, z ktorých každá má dĺžku n je to až $\Theta(n^k)$. To je dôvodom, že sa dá využiť len pri malom počte sekvencií.

Carrillo a Lipman[2] navrhli spôsob redukcie výpočtov nutných na zostrojenie optimálneho SP-zarovnania. Jej rozšírenie bolo použité v programe MSA[3]. V MSA je predchádzajúci algoritmus upravený tak, aby pracoval v opačnom smere. Po vyplnení *D(i, j, k)* sa jej hodnota pošle tým siedmim bunkám, ktoré môže ovplyvniť. Carrillove a Lipmanove urýchlenie určí ešte pred začiatkom hlavného výpočtu ktoré uzly z neho

môžu byť vynechané. V prípade 3 reťazcov tento algoritmus najprv v čase $O(n^3)$ vypočíta ohodnotenie $d_{m,n}(i, j)$ ,kde $m,n \in \{1, 2, 3\}$ zarovnania dvoch sekvencií $S_m(i...n)$ a $S_n(j...n)$. Najkratšia cesta z $(i, j, k)$ do $(n, n, n)$ je minimálne:

$$d_{1,2}(i, j) + d_{1,3}(i, k) + d_{2,3}(j, k)$$

Pri tomto urýchlení sa na začiatok zoberie ľubovoľné zarovnanie sekvencií $S_1$, $S_2$ a $S_3$. Ak je jeho ohodnotenie rovné $z$, tak z výpočtu môžeme vylúčiť všetky uzly, pre ktoré platí:

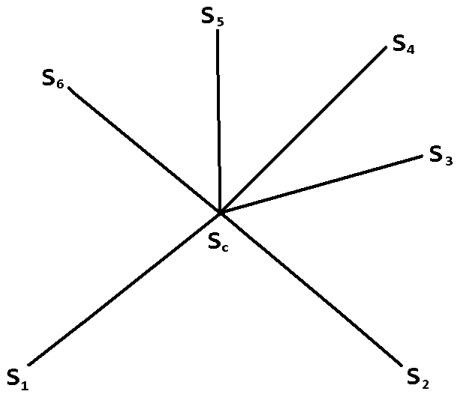
$$D(i, j, k) + d_{1,2}(i, j) + d_{1,3}(i, k) + d_{2,3}(j, k) > z$$

## 2.2 Postupné (progresívne) algoritmy

Progresívne algoritmy využívajú heuristiky na postupnú konštrukciu zarovnania. Začínajú obvykle tak, že zarovnajú malú skupinku veľmi podobných sekvencií (často len 2). Následne k vzniknutému zarovnaniu pridávajú ďalšie reťazce. Medzi sebou zarovnávané sekvencie si zväčša vyberajú na základe stromu a výsledné zarovnanie je s ním konzistentné:

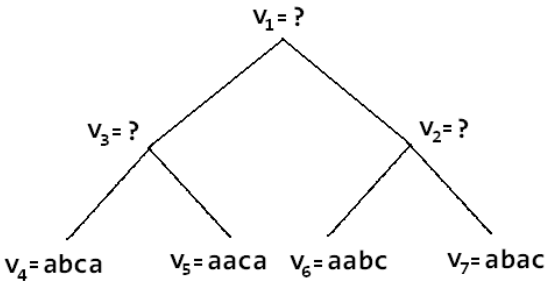
Nech $S$ je množina obsahujúca k reťazcov a $T$ je strom, v ktorom každý uzol je označený jedným z týchto reťazcov. Viacnásobné zarovnanie $M$ reťazcov z $S$ nazývame **konzistentným so stromom *T*,** ak pre každé dva susedné uzly $S_i$ a $S_j$ je ohodnotenie ich zarovnania v $M$ rovné ich minimálnej edit distance ($D(S_i, S_j)$).

K týmto algoritmom patrí napríklad metóda center star, ktorá zostrojuje viacnásobného zarovnania konzistentného so stromom v tvare hviezdy(Obr. 4). Vrcholom stromu je reťazec $S_c$, pre ktorý je hodnota $\sum_{S_i \in S} D(S_i, S_c)$ minimálna. V tomto algoritme potrebujeme pre každý z $k$ reťazcov patriacich do $S$ vypočítať zarovnanie 2 reťazcov pomocou edit distance, ktorého zložitosť je $O(n^2)$. Celková zložitosť tohto algoritmu je teda $O(k*n^2)$. Zarovnanie, ktoré získame touto metódou má v najhoršom prípade SP ohodnotenie blízke dvojnásobku ohodnotenia optimálneho zarovnania.




**Obr. 4 Center star strom pre 7 reťazcov**

Je užitočné vedieť ovplyvniť viacnásobné zarovnanie reťazcov na základe známej evolučnej histórie, aby sme mohli lepšie zarovnať sekvencie príbuzných organizmov. Evolučná história je reprezentovaná evolučným stromom(Obr.5)., ktorého listy reprezentujú známe sekvencie existujúcich organizmov a vnútorné uzly reprezentujú neznáme sekvencie ich predkov. Aby sme podľa fylogenetického stromu mohli vytvoriť zarovnanie, musíme teda najprv určiť vhodné sekvencie pre jednotlivé vnútorné uzly. Táto operácia sa nazýva fylogenetické zarovnanie. Jedným z používaných algoritmov na jeho výpočet je algoritmus Lifted Alignment, kde je každému vnútornému uzlu priradený reťazec, ktorým je označený aj niektorý z jeho potomkov. Ak označíme k počet listov stromu *T* a *N* celkovú dĺžku reťazcov, ktorými sú označené bude časová zložitosť tohto výpočtu $O(N^2 + k^3)$, pričom $O(N^2)$ je zložitosť predspracovania(výpočtu *D* pre všetky dvojice reťazcov). $O(k^2)$ je zložitosť výpočtu *d* pre jeden vnútorný uzol.




**Obr. 5 Príklad fylogenetického stromu**

## 2.3 Iteratívne algoritmy

Iteratívne algoritmy pracujú podobne ako progresívne, ale na rozdiel od nich v iteratívnych krokoch opravujú vzniknuté chyby opätovným zarovnávaním vybraných

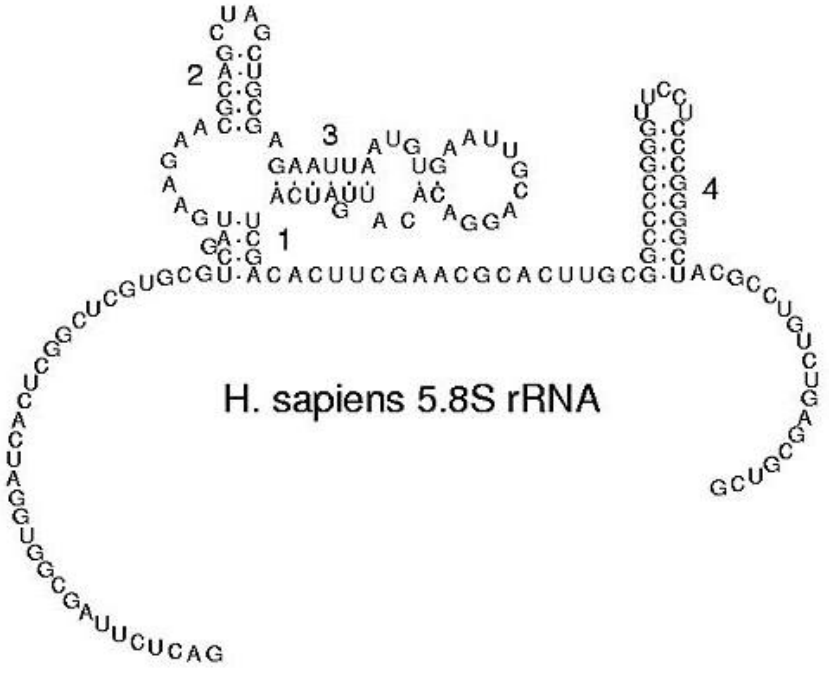
reťazcov k vzniknutému zarovnaniu. Algoritmus končí, keď už nie je možné zarovnanie viac vylepšiť.

Príkladom je metóda opakovaných motívov. Táto metóda spočíva vo vyhľadávaní najdlhších podreťazcov, ktoré sa opakujú v mnohých zarovnávaných sekvenciách. Tento podreťazec sa zvykne nazývať motív. Po nájdení motívu sa podľa neho zarovnajú všetky sekvencie, ktoré ho obsahujú. Na zarovnanie zvyšku sa rekurzívne používa rovnaký algoritmus. Sekvencie, ktoré neobsahujú motív sa zarovnjú samostatne a pripíšu k zarovnaniu tých s motívom.

# 3 Stochastické bezkontextové gramatiky (SCFG) a kovariančné modely (CM)

Na modelovanie primárnej a sekundárnej štruktúry RNA a DNA reťazcov sa v bioinformatike využívajú viacero metód. Patria medzi ne napríklad stochastické bezkontextové gramatiky a kovariančné modely, ktoré popisujem v tejto kapitole. Keďže RNA ani DNA nie sú náhodné reťazce, ale ich zloženie závisí od funkcie a organizmov existujú pre ne podľa toho rôzne gramatiky a kovariančné modely. Kovariančný model je svojim spôsobom akýmsi nahradením gramatiky a je možné ho na ňu prepísať, rovnako ako je možné zo stochastickej bezkontextovej gramatiky vytvoriť kovariančný model.

V RNA nukleotidy adenín(A), cytozín(C), guanín(G) a uracil(U) interagujú a formujú charakteristickú sekundárnu štruktúru, ktorá reprezentuje jej 3D tvar. Skladá sa zo špirál a rôznych druhov slučiek(). Sakakibara[9] ukázal, že tento jej tvar môže byť popísaný pomocou bezkontextových gramatík..

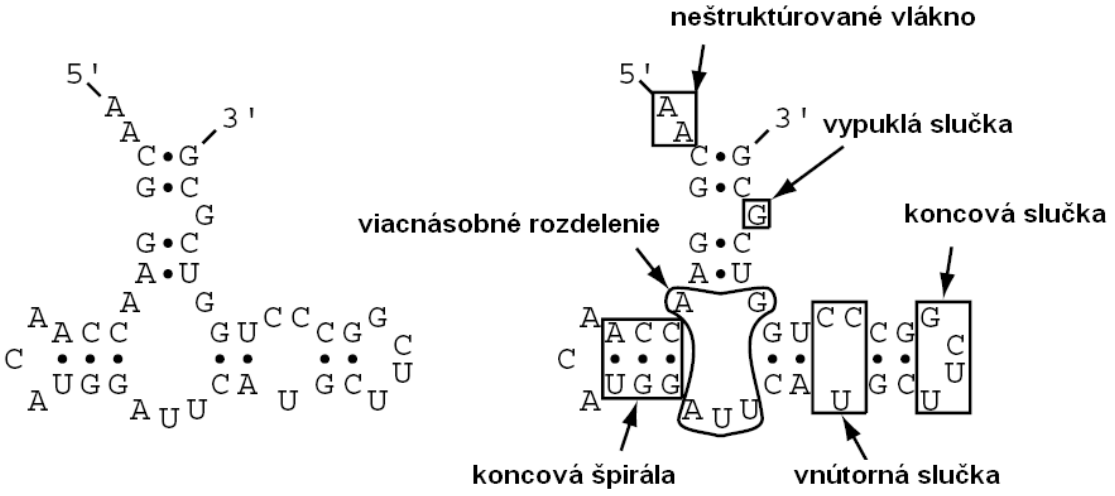



**Obr. 6 Príklad sekundárnej štruktúry.**

Jednoduchý spôsob reprezentácie sekundárnej štruktúry je pomocou zátvoriek. Napríklad ( ( ( ( ( . . . . ) ) ) ) ) reprezentuje špirálu zloženú z piatich párov nukleotidov a slučku zo štyroch nukleotidov. V takomto zápise je však ťažké interpretovať

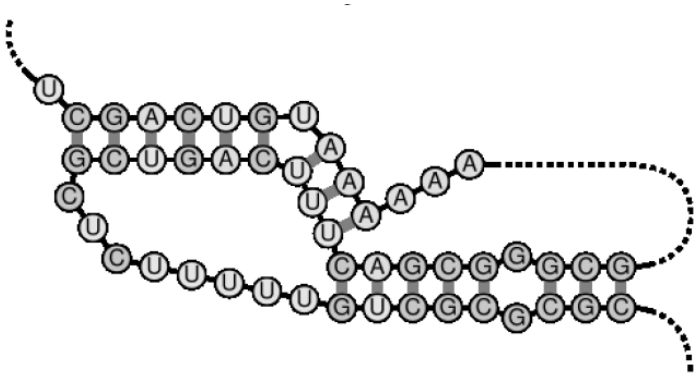
zložitejšie štruktúry. Jednoduchšie je to pomocou WUSS notácie(Washington University Secondary Structure notation - Obr. 7). Tá používa nasledujúce znaky:

1. „< >“ na reprezentáciu jednoduchých koncových špirál
2. „( )“ na reprezentáciu vnútorných špirál ukončujúcich rozdelenia obsahujúce len koncové špirály.
3. „[ ]“ na reprezentáciu vnútorných špirál ukončujúcich rozdelenia obsahujúce aspoň jednu slučku označenú „( )“.
4. „{ }“ na reprezentáciu všetkých vnútorných špirál ukončujúcich hlbšie rozdelenia.
5. „ _ “ na reprezentáciu koncových slučiek.
6. „ - “ na reprezentáciu vypuklých a vnútorných slučiek.
7. „ , “ na reprezentáciu zvyškov v rozdeľovacích slučkách
8. „ : “ na reprezentáciu samostatných vlákien mimo štruktúry
9. „ . “ na reprezentáciu vložení nepatriacich do známej štruktúry
10. Na reprezentáciu jednoduchých pseudouzlov(Obr. 8) sa používa označenie „Aa“, napríklad <<<_AA___>>>aa. Ďalšie pseudouzly môžu byť označené „Bb“, „Cc“, atd..




```
::(((,<<<____>>>,,,<<-<<_____>>-->>,))-))
AACGGAACCAACAUGGAUUCAUGCUUCGGCCCUGGUCGCG
```

**Obr. 7 Príklad WUSS notácie sekundárnej štruktúry**



**Obr. 8 Pseudouzol v sekundárnej štruktúre**

## 3.1 Stochastické bezkontextové gramatiky

Bezkontextová gramatika je je štvorica *G = (N, T, P, S)* v ktorej:

1. *N* je konečná množina neterminálových symbolov
2. *T* je konečná množina terminálových symbolov, pričom $N \cap T = \emptyset$
3. *P* je množina pravidiel typu $A \to \alpha$, kde $A \in N, \alpha \in (N \cup T)^*$
4. *S* je neterminálový štartovací symbol

Podľa konvencie sa neterminálové symboly zvyknú označovať veľkými písmenami a terminálové symboly malými písmenami. Pravidlá popisujú, ako gramatika *G* generuje reťazce terminálových symbolov postupom krokov(odvodením) začínajúc štartovacím symbolom *S*. Ak sa v reťazci zloženom z terminálov a neterminálov nájde znak, ktorý sa nachádza na ľavej strane niektorého pravidla môže sa prepísať reťazcom z pravej strany tohto pravidla. Odvodenie končí, keď už sa na odvodzovaný reťazec nedá použiť žiadne pravidlo. Jednoduchá bezkontextová gramatika generujúca slová typu $a^n b^n$ (slová, v ktorých za reťazcom *n* áčok nasleduje reťazec *n* béčok) môže vyzerať napríklad takto:

$$N = \{S, A, B\}$$
$$T = \{a, b\}$$
$$S = S$$
$$P = \begin{Bmatrix} S \to aSb \\ S \to \varepsilon \end{Bmatrix}$$

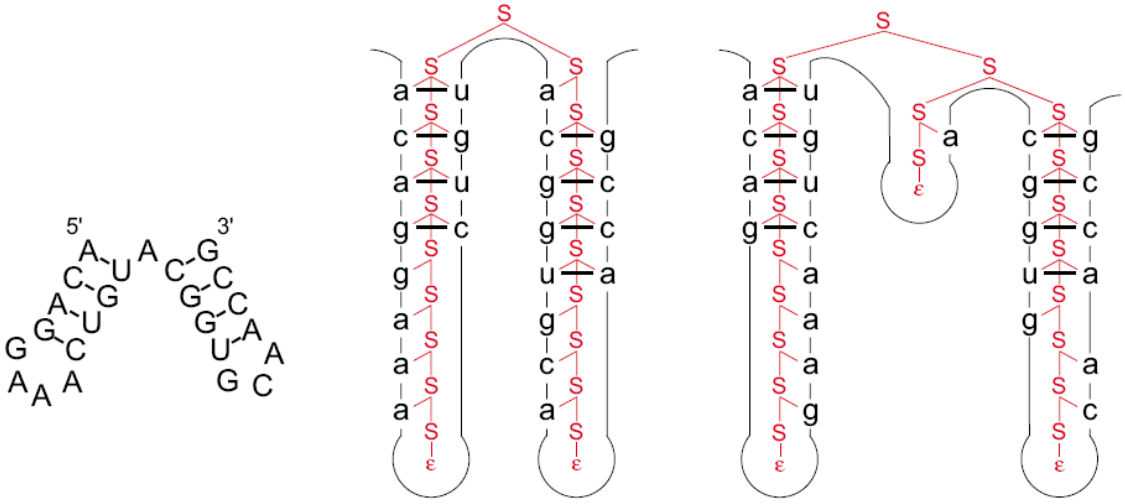
Znak $\varepsilon$ znamená prázdny reťazec. Odvodenie slova aaaaabbbbb by v tejto gramatike vyzeralo takto:

$$S \to aSb \to aaSbb \to aaaSbbb \to aaaaSbbbb \to aaaaaSbbbbb \to aaaaabbbbb$$

Možnosť použitia pravidiel typu $A \to aBc$ nám dovoľuje generovať reťazce na základe sekundárnej štruktúry RNA, pretože môžeme generovať znaky, ktoré sa navzájom spájajú v jednom kroku. Bezkontextová gramatika pre RNA reťazce vyzerá takto:

$$N = \{S\}$$
$$T = \{a, c, g, u\}$$
$$S = S$$
$$P = \begin{Bmatrix} S \to aSu & S \to aS & S \to Sa & S \to SS \\ S \to uSa & S \to cS & S \to Sc & S \to \varepsilon \\ S \to cSg & S \to gS & S \to Sg & \\ S \to gSc & S \to uS & S \to Su & \end{Bmatrix}$$

Odvodenie pomocou bezkontextovej gramatiky má elegantnú reprezentáciu – takzvaný parsovací strom na ktorom dobre vidieť sekundárnu štruktúru RNA. Príklad takéhoto stromu uvedený v [8] je na Obr. 9.




**Obr. 9 Sekundárna štruktúra RNA a parsovací strom znázorňujúci jej odvodenie**

Rozdiel medzi bezkontextovou gramatikou a stochastickou bezkontextovou gramatikou spočíva v tom, že je každému jej pravidlu priradená pravdepodobnosť, s ktorou sa používa. To nám pre zmenu umožňuje zahrnúť do výpočtu aj bežné zmeny

v sekvenciách(inzercie, delécie a substitúcie jednotlivých znakov) vznikajúce počas vývoja.

Stochastická bezkontextová gramatika je teda pätica *G = (N, T, P, S, Pr)* v ktorej:

1. *N* je konečná množina neterminálov
2. *T* je konečná množina terminálov, pričom $N \cap T = \emptyset$
3. *P* je množina pravidiel typu $A \to \alpha$, kde $A \in N, \alpha \in (N \cup T)^*$
4. *S* je štartovací symbol
5. *Pr* je funkcia $\Pr : P \to \langle 0;1 \rangle$, ktorá každému pravidlu z P priradzuje pravdepodobnosť jeho použitia z intervalu $\langle 0;1 \rangle$

Pravdepodobnosť pravidla $A \to \alpha$ označujeme $\Pr(A \to \alpha)$ a pre konkrétny neterminálový symbol *A* platí:

$$\sum_{(A \to \alpha) \in P} \Pr(A \to \alpha) = 1$$

Pravdepodobnosť odvodenia reťazca $\alpha$ sa počíta ako súčin pravdepodobností všetkých pravidiel použitých pri jeho odvodzovaní umocnených na počet použití a označuje sa *Pr(* $\alpha$ *)*

Jednoduchý príklad SCFG:

$$N = \{A, B, C, D\}$$
$$T = \{a, b, c\}$$
$$S = S$$
$$P = \begin{Bmatrix} S \to ABC & B \to b \\ A \to aAa & C \to cCc \\ A \to a & C \to Cabc \\ B \to bBb & C \to \varepsilon \end{Bmatrix}$$

$$\Pr(S \to ABC) = 1 \qquad \Pr(B \to b) = 0{,}4$$
$$\Pr(A \to aAa) = 0.55 \qquad \Pr(C \to cCc) = 0{,}75$$
$$\Pr(A \to a) = 0{,}45 \qquad \Pr(C \to Cabc) = 0{,}1$$
$$\Pr(B \to bBb) = 0{,}6 \qquad \Pr(C \to \varepsilon) = 0.15$$

Odvodenie reťazca aaabbbbbcccabcc by v tejto gramatike vyzeralo takto:

$$S \to ABC \to aAaBC \to aaaBC \to aaabBbC \to aaabbBbbC \to aaabbbbbcCc \to$$
$$\to aaabbbbbcCabcc \to aaabbbbbccCcabcc \to aaabbbbbcccabcc$$

Jeho pravdepodobnosť je:

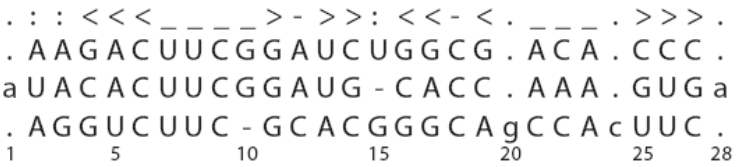
$$\Pr(aaabbbbbcccabcc) = \Pr(S \to abc) \times \Pr(A \to aAa) \times \Pr(A \to a) \times$$
$$\times \Pr(B \to bBb)^2 \times \Pr(B \to b) \times \Pr(C \to cCc)^2 \times \Pr(C \to Cabc) \times \Pr(C \to \varepsilon) =$$
$$= 1 \times 0{,}55 \times 0{,}45 \times 0{,}6^2 \times 0{,}4 \times 0{,}75^2 \times 0{,}1 \times 0{,}15 = 0{,}0003007125$$

## 3.2 Kovariančné modely

Kovariančný model je špeciálna SCFG architektúra pozostávajúca zo skupín modelových stavov, ktoré súvisia so sekundárnou štruktúrou RNA. Dalo by sa tiež povedať, že je to automat. Vzniká na základe viacnásobného zarovnania a sekundárnej štruktúry(riadok v zarovnaní na základe WUSS notácie). Má sedem typov stavov a produkčných pravidiel uvedených v Tab. 2.

| Typ stavu | Popis | Pravidlo | Emisná pravdepodobnosť | Prechodová pravdepodobnosť |
|---|---|---|---|---|
| **P** | emituje pár | $P \to aYb$ | $e_v(a,b)$ | $t_v(Y)$ |
| **L** | emituje vľavo | $L \to aY$ | $e_v(a)$ | $t_v(Y)$ |
| **R** | emituje vpravo | $R \to Ya$ | $e_v(a)$ | $t_v(Y)$ |
| **B** | rozdvojenie | $B \to SS$ | 1 | 1 |
| **D** | vymazanie | $D \to Y$ | 1 | $t_v(Y)$ |
| **S** | začiatok | $S \to Y$ | 1 | $t_v(Y)$ |
| **E** | koniec | $E \to \varepsilon$ | 1 | 1 |

**Tab. 2 Typy stavov v kovariančnom modeli**




**Obr. 10 Príklad zarovnania s 1. riadkom reprezentujúcim sekundárnu štruktúru vo WUSS notácii uvedený v [10]**

Kovariančný model pozostáva z mnohých stavov týchto siedmych typov. Každý z nich má svoju vlastnú emisnú a prechodovú pravdepodobnosť ako aj vlastnú množinu stavov, do ktorých môže prechádzať. Spoločné základné páry sú modelované pomocou stavov typu P, spoločné jednovláknové zvyšky pomocou stavov typu L a R, inzercie pomocou ďalších stavov typu L a R, delécie pomocou stavov typu D a celková sekundárna štruktúra na základe stavov typu S, B a E.

Zostrojenie CM je možné pomocou softvéru INFERNAL, ktorý najprv vyrobí pomocný binárny strom uzlov reprezentujúcich sekundárnu štruktúru. Týchto uzlov je osem typov uvedených v Tab. 3.

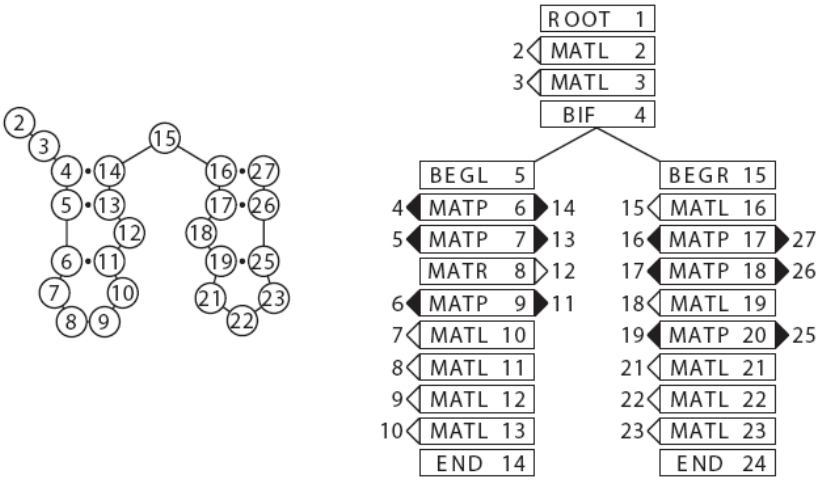
| Uzol | Popis | Typ stavu |
|---|---|---|
| **MATP** | pár | P |
| **MATL** | jednovláknový zvyšok vľavo | L |
| **MATR** | jednovláknový zvyšok vpravo | R |
| **ROOT** | vrchol | S |
| **BIF** | rozdvojenie | B |
| **BEGL** | začiatok ľavej vetvy | S |
| **BEGR** | začiatok pravej vetvy | S |
| **END** | ukončenie vetvy | E |

**Tab. 3 Typy uzlov v pomocnom strome**

Každý z týchto uzlov bude nakoniec obsahovať jeden alebo viac stavov. Pomocný strom korešponduje so zhodujúcou sa časťou zarovnania. Pre jednotlivé sekvencie však ešte treba ošetriť inzercie a delécie. Pomocný strom je kostrou na ktorej INFERNAL stavia kovariančný model. Uzol MATP napríklad obsahuje stav typu P aby mohol vymodelovať zhodujúci sa základný pár, ale aj ďalšie stavy, ktoré sa postarajú o prípadné vloženie alebo vymazanie znaku v, či vedľa tohto páru.

Vrcholom pomocného stromu je ROOT, na znázornenie páru je použité MATP, na zhodné nespárované stĺpce zarovnania MATR a MATL. Na rozvetvenie sa používa BIF a hneď za ním ako začiatok pravej strany BEGR a začiatok ľavej strany BEGL. Ukončenie vetvy sa znázorňuje pomocou END. Stĺpce s vkladanými znakmi, ktoré nie

sú v zhodných častiach sa pri vytváraní pomocného stromu ignorujú. Príklad pomocného stromu je na Obr. 11.




**Obr. 11 Príklad sekundárnej štruktúry zarovnania na Obr. 10 a jemu zodpovedajúceho pomocného stromu taktiež uvedený v [10]. Čísla označujú stĺpce zarovnania.**

Aby bolo zostrojenie pomocného stromu jednoznačné dodržiavajú sa tieto pravidlá:

1. Ak je možné použiť uzol typu MATL namiesto MATR, tak sa použije.
2. Pri opise vnútorných okruhov sa používajú uzly typu MATL pred uzlami typu MATR.
3. Uzly typu BIF sa používajú len ak je to naozaj nevyhnutné.
4. Ak je potrebné viacnásobné rozdelenie rozdelíme najprv i..j na 2 časti i..k a k+1..j tak, aby bola ich dĺžka čo najzhodnejšia.

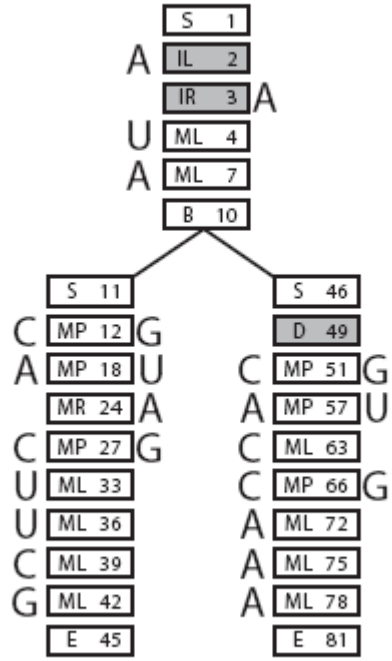
Kvôli tomu, že CM musí riešiť aj mazania a vkladanie častí RNA, ktoré nie sú vo viacnásobnom zarovnaní zhodné je každý uzol zložený z viacerých stavov. Napríklad v uzle typu P musí byť možné okrem vloženia páru aj vymazanie ktoréhokoľvek z jeho členov, ako aj vloženie nezhodných zvyškov. V Tab. 4 je uvedené na aké stavy sú rozčlenené jednotlivé typy uzlov.

| Uzol | Stavy |
|---|---|
| **MATP** | MP MR ML D IR IL |
| **MATL** | ML D IL |
| **MATR** | MR D IR |
| **ROOT** | S IL IR |
| **BIF** | B |
| **BEGL** | S |
| **BEGR** | S IL |
| **END** | E |

**Tab. 4 Rozčlenenie uzlov na stavy**

Stavy MP, ML a MR sú určené pre zhodné časti zarovnania, stavy IL a IR sú určené na vkladanie znakov, ktoré sa v zarovnaní nezhodujú s ostatnými v danom stĺpci. V [10] rozdeľujú stavy do 2 skupín. V prvej sú stavy MP, ML, MR, B, S, E a D, v druhej stavy IL a IR. Zatiaľ čo z prvej skupiny je v každom uzle navštívený práve jeden stav, z druhej skupiny nemusí byť navštívený ani jeden, resp. môže ich byť navštívených viacero. Dokonca je možné v jednom uzle navštíviť ktorýkoľvek stav z druhej skupiny viac krát. Zo stavu B sa dá dostať iba do stavov S v nasledujúcich uzloch typu BEGL a BEGR. Pre ostatné uzly platí, že z každého stavu z prvej skupiny sa dá dostať do každého stavu druhej skupiny toho istého uzla a do každého stavu prvej skupiny nasledujúceho uzla. Stavy druhého typu majú hranu do každého stavu prvého typu nasledujúceho uzla a do seba samého. Okrem toho má ešte stav typu IL hranu smerujúcu do stavu IR v tom istom uzle.

Aby sme získali prechodové a emisné pravdepodobnosti k jednotlivým hranám musíme pre každú sekvenciu zo vstupného viacnásobného zarovnania zostrojiť parsovací strom na základe vytvoreného kovariančného modelu.



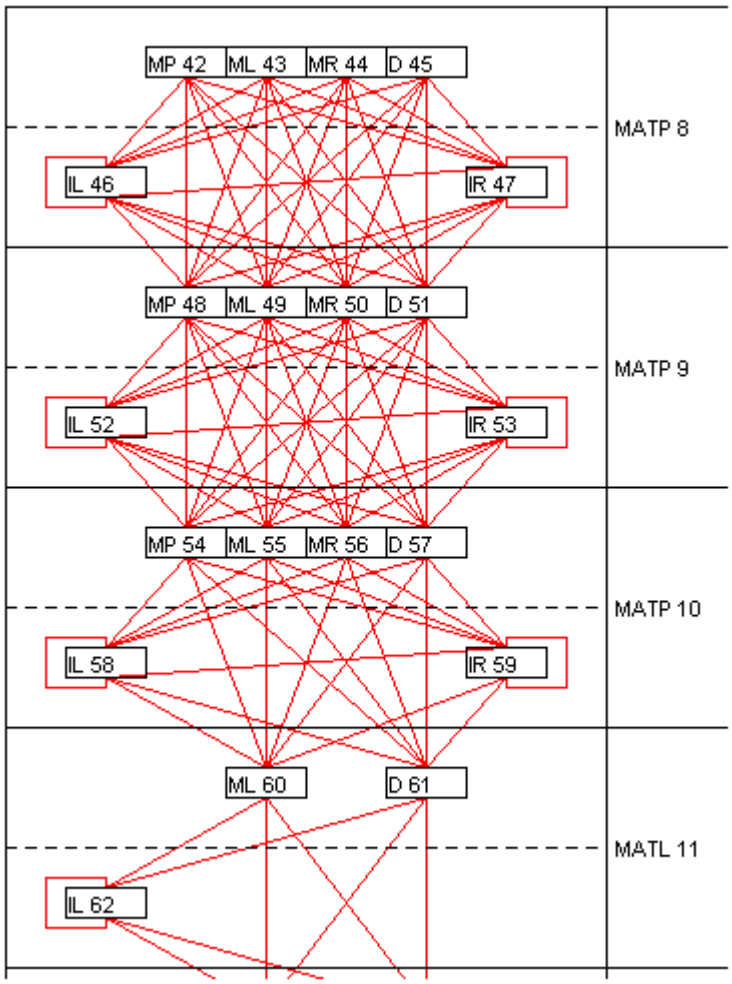

**Obr. 12 Príklad parsovacieho stromu pre druhú sekvenciu z Obr. 10 uvedený v[10]**

## 3.3 CM Browser

Kovariančné modely vygenerované INFERNAL-om majú neprehľadnú textovú formu(Obr. 13). Preto som spravila prehliadač, ktorý zobrazuje INFERNAL-om vygenerované CM vo forme grafu(Obr. 14).

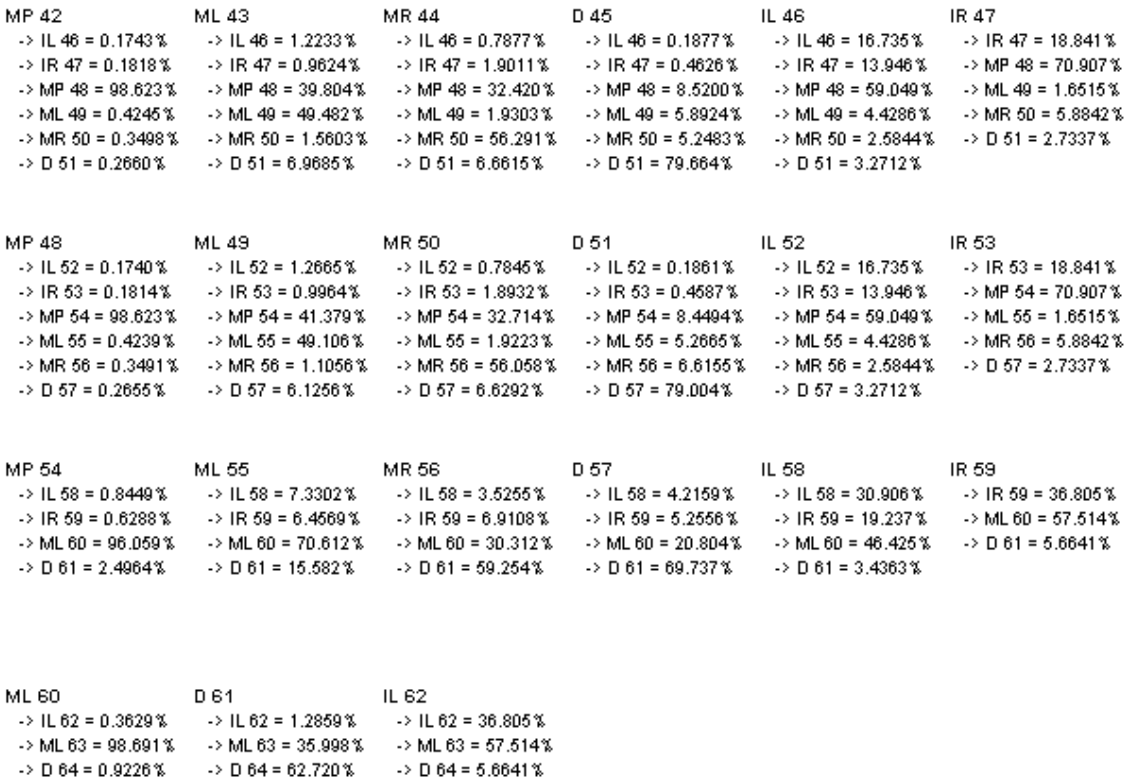
```
                                [ MATL   11 ]
ML    60    59 6    62     3  -8.106  -0.019  -6.760                  -0.328  0.844 -0.972 -0.155
 D    61    59 6    62     3  -6.281  -1.474  -0.673
IL    62    62 3    62     3  -1.442  -0.798  -4.142                  0.660 -0.612 -0.293 -0.076
                                [ MATL   12 ]
ML    63    62 3    65     3  -8.117  -0.019  -6.771                  1.658 -2.070 -2.065 -1.443
 D    64    62 3    65     3  -6.174  -1.687  -0.566
IL    65    65 3    65     3  -1.442  -0.798  -4.142                  0.660 -0.612 -0.293 -0.076
                                [ MATL   13 ]
ML    66    65 3    68     3  -8.117  -0.019  -6.771                  -0.783 -1.219 -1.577  1.408
 D    67    65 3    68     3  -6.174  -1.687  -0.566
IL    68    68 3    68     3  -1.442  -0.798  -4.142                  0.660 -0.612 -0.293 -0.076
                                [ MATL   14 ]
ML    69    68 3    71     3  -6.837  -0.511  -1.788                  1.486 -1.722 -1.238 -1.082
 D    70    68 3    71     3  -5.620  -0.734  -1.403
IL    71    71 3    71     3  -1.908  -0.570  -4.061                  0.654 -0.640 -0.301 -0.039
                                [ MATR   15 ]
MR    72    71 3    74     2  -7.909  -0.006                          -0.027  0.748 -0.956 -0.279
 D    73    71 3    74     2  -6.324  -0.018
IR    74    74 3    74     2  -1.823  -0.479                          0.660 -0.612 -0.293 -0.076
                                [ BIF    16 ]
 B    75    74 3   205    76
                                [ BEGR   57 ]
 S    76    75 1    77     5  -7.227  -0.116  -4.048  -7.255  -8.146
IL    77    77 2    77     5  -2.408  -0.496  -4.087  -5.920  -5.193  0.660 -0.612 -0.293 -0.076
                                [ MATP   58 ]
```

**Obr. 13 Úryvok kovariančného modelu vygenerovaného INFERNAL-om**




**Obr. 14 Časť kovariančného modelu vygenerovaná CM Browserom**

Prechodové pravdepodobnosti jednotlivých hrán sú v CM Browseri vypísané vedľa modelu vo formáte aký je vidieť na Obr. 15.




**Obr. 15 Prechodové pravdepodobnosti kovariančného modelu v CM Browseri**

### 3.3.1 Rozbor CM vygenerovaného INFERNAL-om

INFERNAL ukladá kovariančné modely do textových súborov s príponou cm. Tieto súbory sú dostupné na stránke Rfam databázy[13]. Ich popis však nie je uvedený ani v užívateľskej príručke INFERNAL-u, kvôli tomu že sa často mení. Po preštudovaní zdrojového kódu som však dospela k poznatkom o jeho štruktúre.

Z Obr. 13 si môžete všimnúť, že označenie jednotlivých uzlov je zapísané v hranatých zátvorkách, v ktorých je uvedený typ a poradové číslo uzla. Hneď za týmto označením nasleduje popis stavov v danom uzle.

V riadku popisujúcom stav sú postupne uvedené tieto údaje:

1. Typ stavu.
2. Poradové číslo stavu.
3. Poradové číslo posledného stavu, z ktorého vedie hrana do popisovaného stavu.
4. Počet stavov, z ktorých vedie hrana do popisovaného stavu(všetky tieto stavy sú vždy zoradené za sebou).

5. Poradové číslo prvého stavu do ktorého vedie hrana z popisovaného stavu.
6. Počet stavov do ktorých vedie hrana z popisovaného stavu(všetky tieto stavy sú vždy zoradené za sebou).
7. Prechodové pravdepodobnosti pre hrany idúce z popisovaného stavu(ich formát popíšem neskôr).
8. Emisné pravdepodobnosti pre jednotlivé emitované znaky(v poradí A,C,G,U). V prípade stavu typu MP emisné pravdepodobnosti pre každú možnú dvojicu emitovaných znakov(v poradí AA, AC, AG, AU, CA, CC, CG, CU, GA, GC, GG, GU, UA, UC, UG, UU).

Popis stavu typu B neobsahuje body 5, 6, 7 a 8. Namiesto nich obsahuje poradové čísla jediných 2 stavov, do ktorých z neho vedie hrana. Keďže ide o rozdvojenie neemitujú sa žiadne znaky a obe prechodové pravdepodobnosti sú rovné 1.

Pravdepodobnosti v infernalovskom CM súbore sú vo formáte nazývanom „log odds ratio“. Na prevod týchto pravdepodobností na pravdepodobnosť z intervalu (0;1) sa používajú tieto vzorce:

$$t_{v(0;1)}(Y) = 2^{t_v(Y)}$$

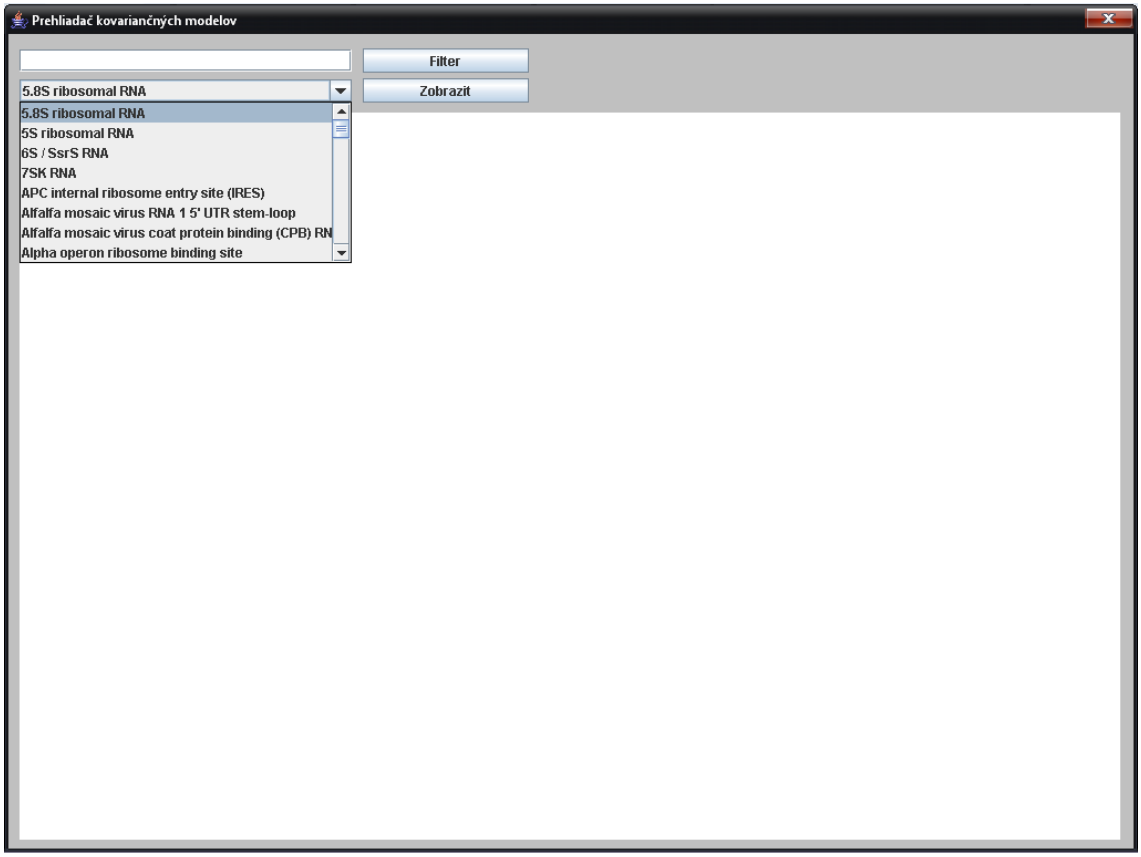
$$e_{v(0;1)}(a) = 2^{e_v(a)} \cdot 0{,}25$$

$$e_{v(0;1)}(a,b) = 2^{e_v(a,b)} \cdot 0{,}25^2$$

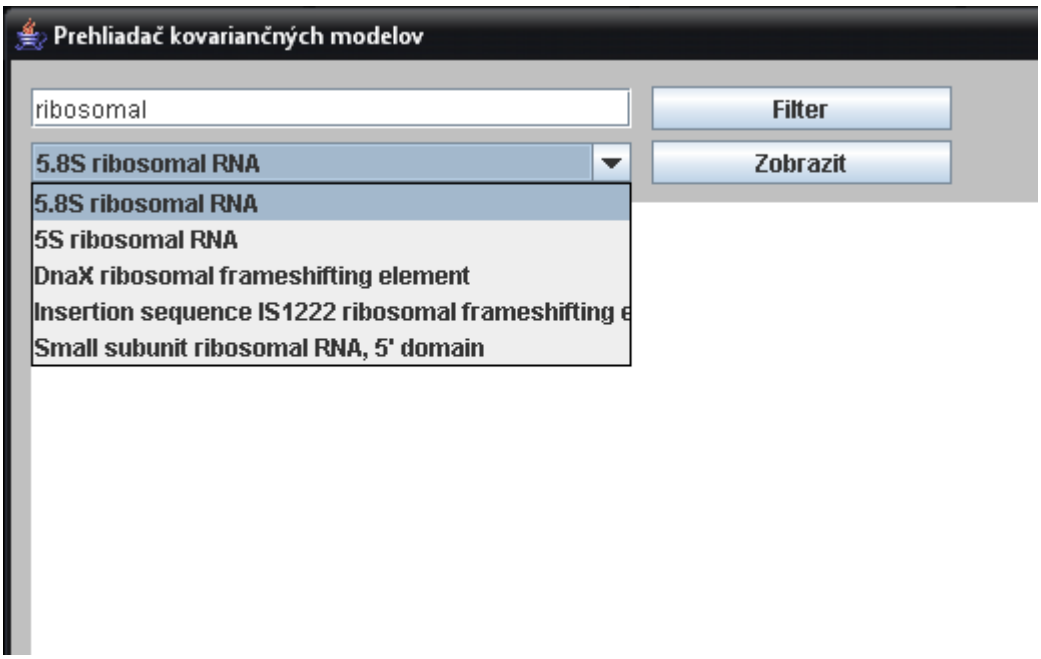
$t_v(Y)$, $e_v(a)$ a $e_v(a,b)$ v tomto vzorci sú pravdepodobnosti vygenerované INFERNAL-om . Tento vzťah pochádza priamo zo zdrojového kódu INFERNAL-u.

### 3.3.2 Ovládanie CM Browsera

Ovládanie CM Browsera je jednoduché. V druhom riadku okna sa nachádza políčko, v ktorom je možné vybrať si RNA rodinu, ktorej kovariančný model chceme zobraziť. Hneď za týmto políčkom je tlačidlo, po ktorého stlačení sa model zobrazí. Keďže ponúkaných rodín je dosť veľa(približne 600) zabudovala som do prehliadača aj filter. Políčko pre filter sa nachádza v prvom riadku okna. Po stlačení vedľa neho ležiaceho tlačidla „Filter“ sa zredukujú RNA rodiny ponúkané browserom na rodiny, ktoré vo svojom názve obsahujú reťazec zadaný vo filtri(Obr. 17).

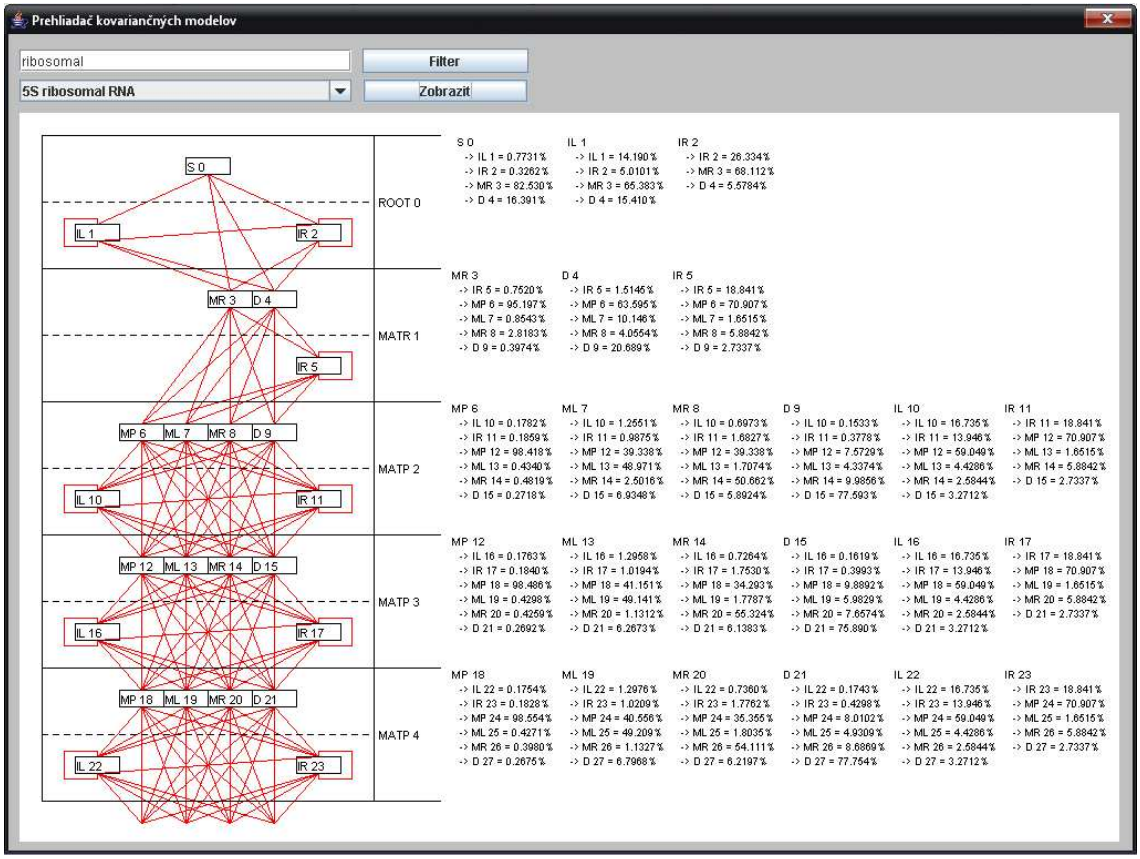



**Obr. 16 Okno CMBrowsera**




**Obr. 17 Výber RNA rodín po použití filtra**

Po aktivovaní plátna, na ktorom sa zobrazí kovariančný model je možné týmto modelom prechádzať pomocou šípok smerom hore a dole. Ak je niektorý zo zobrazených uzlov typu BIF je možné prepnúť sa na prezeranie pravej vetvy pomocou šípky vpravo a na prezeranie ľavej vetvy pomocou šípky vľavo. Pre jednoduchší návrat k poslednému rozdvojeniu je možné použiť tlačidlo Backspace. Po viacerých stlačeniach Backspace sa dostanete naspať k vrcholu kovariančného modelu.




**Obr. 18 Okno CMBrowsera po zobrazení kovariančného modelu**

# 4 Záver

V práci sme popísali stochastické bezkontextové gramatiky a kovariančné modely slúžiace na modelovanie primárnej a sekundárnej štruktúry DNA a RNA. Popísali sme spôsob zostrojenia kovariančného modelu na základe viacnásobného zarovnania, ktorý využíva softvér INFERNAL. Zostrojili som prehliadač kovariančných modelov RNA rodín z Rfam databázy, ktorý zobrazuje v grafickej podobe kovariančné modely vygenerované INFERNAL-om.

V budúcnosti by sme ešte mohli do CM Browsera doplniť zobrazenie emisných pravdepodobností, generovanie reťazcov podľa kovariančného modelu, resp. vyhľadávanie vzorov v sekvenciách pomocou kovariančného modelu.

# Zoznam použitej literatúry


[1] Gusfield,D.: Algorithms on Strings, Trees and Sequences. Cambridge Univ. Press, Cambridge, 1997.

[2] H. Carrillo and D. Lipman: The multiple sequence alignment problem in biology. SIAM J. Appl. Math, 1988.

[3] J. Monod: Chance and Necessity; An Essay on the Naturel Philosophy of Modern Biology. Knopf, New York, 1971.

[4] http://en.wikipedia.org/wiki/Multiple_sequence_alignment

[5] Sankoff, D.: Minimal mutation trees of sequences. SIAM J. Appl. Math., 1975.

[6] M. Waterman, M. Perlwitz: Line geometries for sequence comparisons. Bull. Math. Biol., 1984.

[7] M. Vingron, P. Argos: Motif recognition and alignment for many sequences by comparison of dot-matrices. J. Mol. Biol., 1991.

[8] R. D. Dowell, S. R. Eddy: Evaluation of several lightweight stochastic context-free grammars for RNA secondary structure prediction. BMC Bioinformatics, 2004. Dostupné na internete http://www.biomedcentral.com/1471-2105/5/71.

[9] Y. Sakakibara, M.Brown, R. Hughey, I. S. Mian, K. Sjolander, R. C. Underwood, D. Haussler: Stochastic context-fee grammars for tRNA modeling. Nucleic acids research, vol. 22, 1994.

[10] INFERNAL user's guide, 2007. Dostupné na internete http://infernal.janelia.org/.

[11] S. R. Eddy, R. Durbin: RNA sequence analysis using covariance models. Nucleic Acids Research, 1994. Dostupné na internete http://selab.janelia.org/pub/publications/Eddy94/Eddy94-preprint.pdf.

[12] N. C. Jones, P.A. Pevzner: An introduction to bioinformatics algorithms. Massachusetts Institute of Technology, 2004.

[13] http://www.sanger.ac.uk/Software/Rfam/

[14] http://en.wikipedia.org/wiki/Bioinformatics

[15] http://www.ncbi.nlm.nih.gov/About/primer/bioinformatics.html

[16] http://bioinformatics.ubc.ca/about/what_is_bioinformatics/

[17] http://www.roseindia.net/bioinformatics/bioinformatics.shtml

# Zoznam obrázkov

# Zoznam tabuliek