Vizualizace sekundární struktury RNA s využitím existujících struktur

Richard Eliáš, David Hoksza

Matematicko-fyzikální fakulta

richard.elias@matfyz.cz

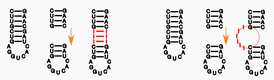
Motivácia a ciele práce

Molekula RNA sa stáva predmetom mnohých štúdií, vďaka čomu rastie dopyt po nástrojoch pomáhajúcich pri jej analýze. Vlastnosti molekuly sú síce ovplyvnené primárou štruktúrou (poradím nukleotidov v reťazci), no viac závisia na ich priestorovom usporiadaní (terciárna štruktára). My sa v práci zaoberáme trochu zjednodušeným modelom - sekundárnou štruktúrou. Tú reprezentuje zoznam nukleotidov spojených väzbou. Tieto nukleotidy musia byť blízko aj v priestore a tak nám sekundárna štruktúra relatívne dobre aproximuje terciárnu, pre ktorú neexistujú spoľahlivé metódy zisťovania štruktúry už ani pre relatívne malé molekuly.

Prvým krokom pri analýze RNA molekuly je často rozbor obrázka jej sekundárnej štruktúry. Medzi základné kritéria ktoré musia obrázky molekúl spĺňať patrí rovinnosť nakreslenia, kreslenie loopov na kružnice a stemy na priamkách. Pri porovnávaní štruktúr sa využíva taktiež kreslenie častí majúcich podobnú funkciu a tvar na rovnaké miesta v obrázkoch, čo pomáha lepšej orientácií v molekule a napomáha nájsť konzervované časti v molekulách. V súčastných nástrojoch (mFold, RNAViz, RNAView, $\cdots$) sa toto posledné kritérium nedodržuje, čo má za následok ťažké nachádzanie konzervovaných častí v molekulách.

Kreslenie chýbajúcich častí v molekule

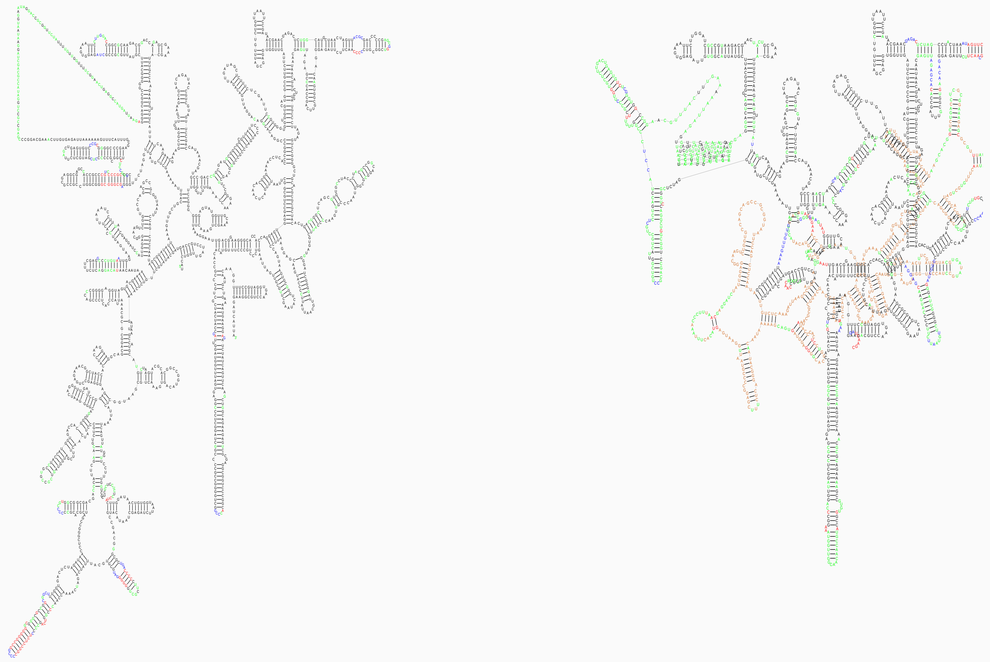
Po transformácií šablónovej molekuly na cieľovú, získavame čiastočnú vizualizáciu cieľovej RNA a jej zvyšok potrebujeme dopočítať. Po operáciách delete nám v obrázku ostávajú prázdne miesta, naopak po insertoch potrebujeme pre dané vrcholy urobiť v obrázku miesto.



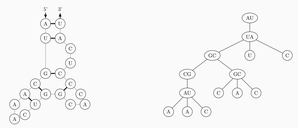
Mazanie vrcholov stromu je inverzná operácia ku vkladaniu a tak si ukážeme iba vkladnie. Vkladanie báz do loopu (okrem multibranch) je jednoduché, vytvoríme si iba novú kružnicu na ktorú všetky bázy uložíme. Vkladanie báz do stemu potrebuje najprv urobiť pre nich najprv urobiť miesto a tak celú štruktúru posunie smerom od rodičovského vrcholu stromu. Algoritmus vkladania je uvedený na obrázku.

Pre multibranch loopy je to trochu zložitejšie, chceme sa totiž vyhnúť prípadom, kedy ju musíme celú prekresliť, keďže pri tom vznikajú veľké problémy s prekryvmi. Prekresleniu štruktúry sa nevyhneme, ak sa jedná o vkladanie veľkého počtu báz, alebo vkladáme celú novú vetvu RNA.

Príklady výslednej vizualizácie (človek (K03432) a žaba (X04025), mušľa (L24489) a cikáda (U06478))



Stromová reprezentácia RNA a použitie tree-edit-distance algoritmu



Vďaka zanedbaniu existencie pseudouzlov, môžeme sekundárnu štruktúru reprezentovať ako usporiadaný zakorenený strom (pseudouzly sú páry vedúce medzi vetvami stromu). Transformácia sekundárnej štruktúry do stromovej podoby je na obrázku. Vnútorný vrchol stromu reprezentuje bázový pár a listy nespárované nukleotidy.

Pôvodným nakreslením chceme hýbať čo najmenej a tak potrebujeme spôsob ako odlíšiť časti, ktoré sa v oboch molekulách nemenia. K tomu využijeme algoritmus *tree-edit-distance*, ktorý nám dá návod ako určiť najmenší počet úprav, ktorými vieme transformovať šablónovú molekulu na cieľovú. Úpravami myslíme editačné operácie update, insert a delete (zmena bázy vo vrchole, vloženie lebo zmazanie vrcholu).

Základom algoritmu je rekurzívny vzorec, ktorý určí vzdialenosť medzi dvoma lesmi (r_F a r_G označuje najpravejší, alebo najľavejší vrchol lesa F a G, c_{del}, c_{ins}, c_{upd} sú ceny mazania, vkladania a updatu vrcholu v strome). Využitím algoritmu *RTED* môžeme vzdialenosť medzi stromami spočítať v čase $\mathcal{O}(n^3)$. Ten nám totiž vie predpočítať optimálnu dekompozíciu, teda do ktorej vetvy rekurzie sa zanoriť, a ktorú naopak vynechať.

Bolo dokázané, že pre každú dekompozičnú stratégiu existujú stromy, ktoré potrebujú aspoň $\mathcal{O}(n^3)$ času a tak je tento algoritmus optimálny.

Nakoniec, algoritmus *TED* nám dáva návod ako stromy navzájom transformovať jeden na druhý. Ide len o pamätanie si, že ak som v nejakom kroku rekurzie vrchol mazal, tak ho potrebujem zmazať aj pri transformácií.

$$\delta(\emptyset,\emptyset) = 0$$
$$\delta(F,\emptyset) = \delta(F - r_F,\emptyset) + c_{del}(r_F)$$
$$\delta(\emptyset,G) = \delta(\emptyset,G - r_G) + c_{ins}(r_G)$$
$$\delta(F,G) = \begin{cases} \delta(F - r_F, G) + c_{del}(r_F) \\ \delta(F, G - r_G) + c_{ins}(r_G) \\ \delta(F - F_{r_F}, G - G_{r_G}) \\ \quad +\delta(F_{r_F} - r_F, G_{r_G} - r_G) \\ \quad +c_{upd}(r_F, r_G) \end{cases}$$

Obdobne je to aj s ostatnými operáciami.

Nástroj TRAVeLer

V rámci práce sme implementovali nástroj TRAVeLer, ktorý je schopný podľa vstupného obrázka vizualizovať cieľovú štruktúru.

Program implementuje *tree-edit-distance* algoritmus, ktorý nám zabezpečí správne mapovanie medzi vzorovou a cieľovou štruktúrou. Následným použitím nášho dokresľovacieho algoritmu vznikne výsledná vizualizácia.

V nasledujúcich príkladoch uvažujeme premenné INDIR=../InFiles/ a OUTDIR=/tmp/.

Príklad 1: generovanie vizualizácie RNA myši pomocou obrázka RNA človeka

```
# ./build/traveler –match-tree $INDIR/mouse.fasta –template-tree $INDIR/human.ps $INDIR/human.fasta –all $OUTDIR/mouse_to_human
```

Príklad 2: získavanie mapovania medzi štruktúrami

```
# ./build/traveler –match-tree $INDIR/mouse.fasta –template-tree $INDIR/human.ps $INDIR/human.fasta \
    –ted $OUTDIR/mouse_to_human.map
```

Príklad 3: generovanie vizualizácie z existujúceho mapovania

```
# ./build/traveler –match-tree $INDIR/mouse.fasta –template-tree $INDIR/human.ps $INDIR/human.fasta \
    –draw –colored –overlaps $INDIR/mouse_to_human.map $OUTDIR/mouse_to_human
```

Argumentom *–colored* zapíname farebné kódovanie v obrázku: **červená** - insert, **zelená** - edit, **modrá** - prekreslenie báz, **hnedá** - prekreslenie multibranch loopy.

Výsledky experimentov

Program sme testovali na reálnych dátach z CRW databázy. Ako testovaciu sadu sme zvolili malú podjednotku 16S ribozomálnej RNA v živočíšnej ríši. V databáze je ulozených 16 molekúl (kompletných - obrázok aj štruktúra), z ktorých sme získali 256 výsledných vizualizácií (každý s každým).

V počte prekryvov sme sa u väčšiny prípadov držali počtu do 10 (151), ak vynecháme molekuly ktoré potrebovali prekresliť multibranch loopy, tak do 10 prekryvov bolo nakreslených 150 molekúl a nad 10 ich bolo iba 29.

Počty prekryvov v závislosti na vzdialenosti TED nieje štatisticky až taká zjavná. Ak vezmeme najbližšie štruktúry, je v priemere asi 5 prekryvov a so zväčšujúcou sa vzdialenosťou rástli výkyvy hlavne kvôli tomu, že niektoré štruktúry boli dobre konzervované a na druhej strane bolo pár extrémov, ktoré sa vizualizovali veľmi ťažko.

V budúcnosti by bolo vhodné upraviť kresliace algoritmy, implementovať otáčanie vetiev RNA stromov v prípade indikovania prekryvov alebo pridať interaktívny nástroj na úpravu obrázkov, čím by užívateľ prekryvy ručne odstránil.