

Grafovski probabilistički modeli za analizu i predviđanje struktura sekvenci i njihove primene u bioinformatički

Nevena Ćirić
Lazar Vasović

Pregled

- Vrste grafovskih probabilističkih modela
- Sistematizacija modela struktura sekvenci
- HMM
- SCFG
- Struktura RNK
- Opis implementacije

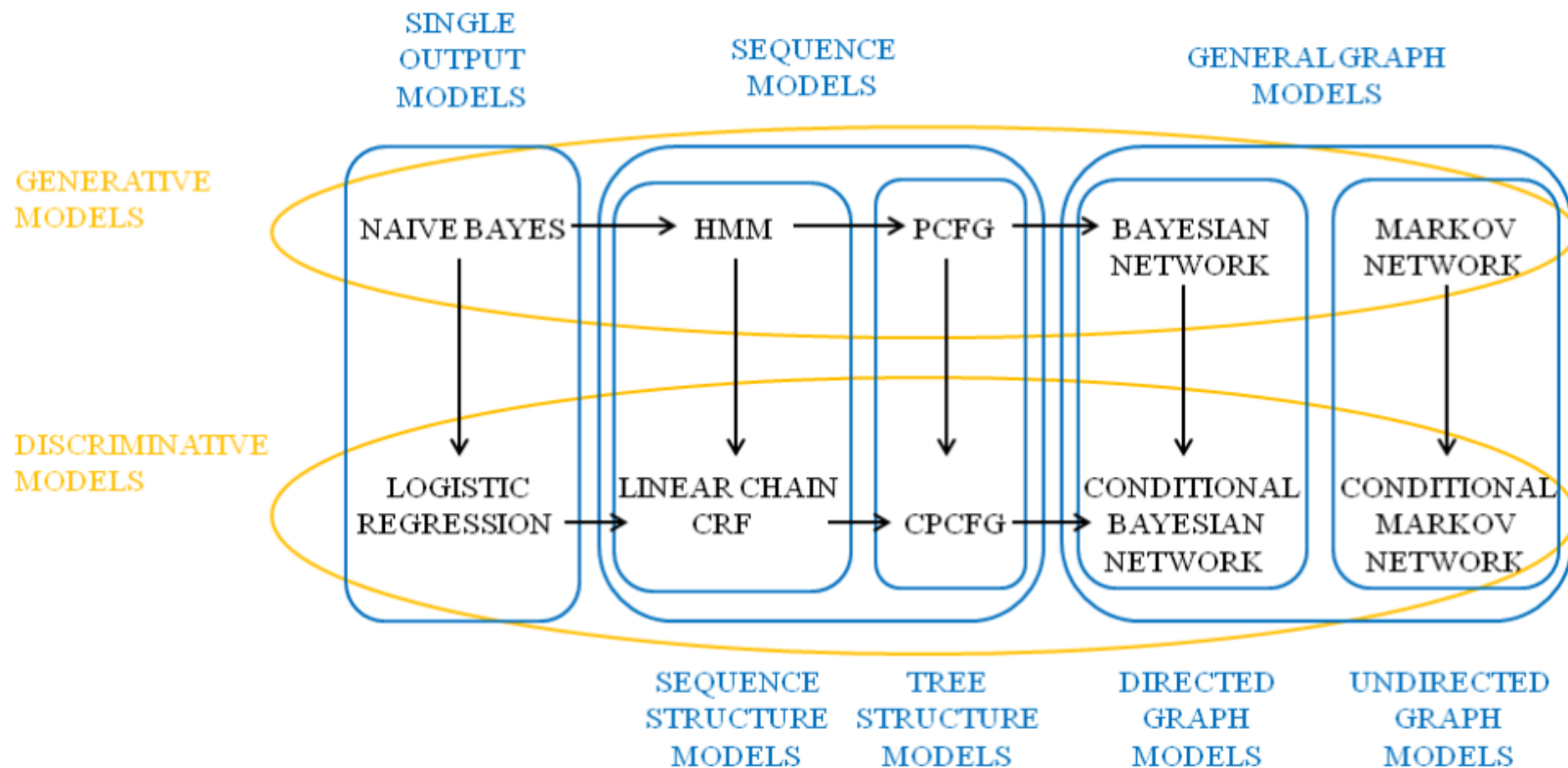
Vrste grafovskih probabilističkih modela

- grafovima se opisuju međusobne zavisnosti između slučajnih promenljivih koje se modeluju
- prema tipu raspodele koju modeluju i strukturi grafa, mogu se uočiti različite vrste grafovskih probabilističkih modela
- prema tipu raspodele koju modeluju – **generativni i diskriminativni modeli**
- generativni probabilistički modeli modeluju zajedničku raspodelu svih slučajnih promenljivih od interesa
- diskriminativni probabilistički modeli modeluju uslovnu raspodelu izlaznih (ciljnih) slučajnih promenljivih za date vrednosti ulaznih (opaženih) slučajnih promenljivih

Vrste grafovskih probabilističkih modela

- **grafovski probabilistički modeli određuju familije raspodela verovatnoće koje se faktorišu prema odgovarajućem grafu**
- u zavisnosti od toga da li se radi o usmerenom ili neusmerenom grafu, imamo podelu na **Bajesovske** i **Markovljeve mreže**
- ove dve vrste modela razlikuju se u tipu međuzavisnosti između slučajnih promenljivih koje mogu da opišu
- Bajesovske i Markovljeve mreže u opštem slučaju modeluju zajedničku raspodelu (generativni modeli), ali se za opažene vrednosti nekih slučajnih promenljivih mogu prilagoditi tako da modeluju uslovnu raspodelu u odnosu na date promenljive
- ista reprezentacija i parametrizacija može se iskoristiti za modelovanje uslovne raspodele tako što se raspodele pridružene faktorima renormalizuju u odnosu na fiksirane vrednosti opaženih slučajnih promenljivih — tada govorimo o **uslovnim Bajesovskim i Markovljevim mrežama***

Vrste grafovskih probabilističnih modela



Pregled

- Vrste grafovskih probablističkih modela
- **Sistematizacija modela struktura sekvenci**
- HMM
- SCFG
- Struktura RNK
- Opis implementacije

Sistematizacija modela struktura sekvenci

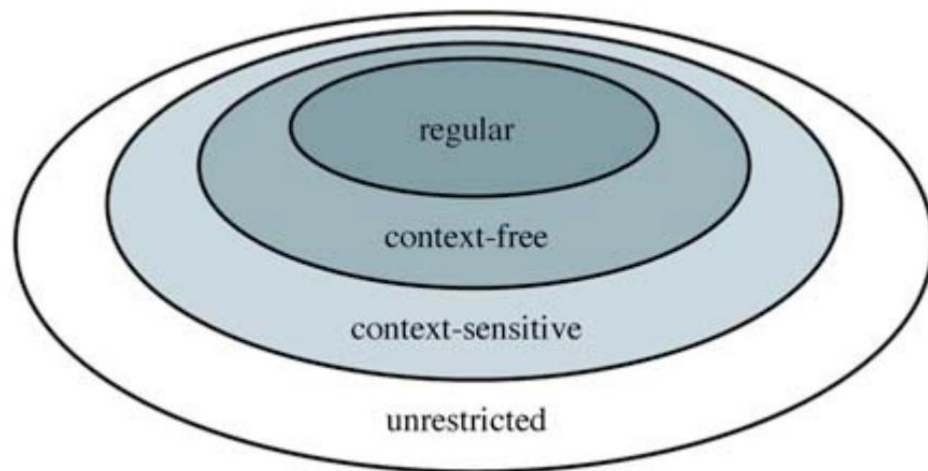
- nadalje će biti razmatrani samo sekvencijalni grafovski probabilistički modeli i kako se oni uklapaju u opštu teoriju modelovanja sekvenci
- modeli sekvenci su važni u bioinformatičari zato što se njima opisuju strukture poput DNK i RNK (sekvence nukleotidnih baza) i proteina (sekvence aminokiselina)
- najjednostavnija struktura sekvenci podrazumeva postojanje susednih zavisnosti između elemenata sekvenci – vrednost i -tog elementa sekvence zavisi od vrednosti $(i-1)$ -og elementa sekvence
- kompleksnije strukture sekvenci uključuju dugoročne zavisnosti između pojedinačnih elemenata sekvenci, zavisnosti jednog elementa od više prethodnih elemenata sekvenci, kao i njihove kombinacije

Sistematizacija modela struktura sekvenci

- opštu teoriju modelovanja struktura sekvenci formalizovali su računarski lingvisti kroz **teoriju formalnih jezika i gramatika**
- formalne gramatike se definišu konačnim skupom simbola i pravila izvođenja $\alpha \rightarrow \beta$, gde su α i β nizovi simbola
- postoje dve vrste simbola – apstraktni **neterminalni (nezavršni) simboli** i **terminalni (završni) simboli** koji se pojavljuju u sekvencama (rečima) odgovarajućeg formalnog jezika
- leva strana pravila izvođenja α sadrži najmanje jedan neterminalni simbol kome se desnom stranom pravila izvođenja β pridružuje neki niz terminalnih i/ili neterminalnih simbola
- kako bismo ih jasno razlikovali, koristićemo mala slova za terminalne, a velika za neterminalne simbole

Sistematizacija modela struktura sekvenci

- Čomski definiše četiri tipa strukture pravila izvođenja gramatika
- rezultujuće četiri klase formalnih gramatika, koje se sastoje samo od pravila izvođenja odgovarajućeg tipa, čine hijerarhiju poznatu kao **hijerarhija Čomskog**



Sistematizacija modela struktura sekvenci

- ove četiri klase gramatika su ugneždene prema restriktivnosti njihovih pravila izvođenja, a samim tim i odnosu skupova jezika koje te gramatike mogu da opišu
- **regularne gramatike** – dozvoljena su samo pravila izvođenja oblika $W \rightarrow aW$ ili $W \rightarrow a$
- **kontekstno-slobodne gramatike** – sva pravila izvođenja su oblika $W \rightarrow \beta$
- **kontekstno-osetljive gramatike** – pravila izvođenja su oblika $\alpha_1 W \alpha_2 \rightarrow \alpha_1 \beta \alpha_2$
- **gramatike bez restrikcija** – ni leva ni desna strana pravila izvođenja nemaju restrikcije, odnosno oblika su $\alpha_1 W \alpha_2 \rightarrow \gamma$
- takođe, ova hijerarhija odražava mogućnost gramatika da opišu različite vrste međuzavisnosti elemenata sekvenci, odnosno struktura sekvenci

Sistematizacija modela struktura sekvenci

- regularne gramatike mogu da opišu samo najjednostavnije međuzavisnosti elemenata sekvenci – zavisnost sledećeg elementa od prethodnog
- u odnosu na regularne gramatike, kontekstno-slobodne gramatike dozvoljavaju dodatna pravila koja omogućavaju modelovanje ugnežđenih, dugoročnih zavisnosti (parova) elemenata sekvenci
- kontekstno-osetljive gramatike dozvoljavaju pravila izvođenja oblika $AB \rightarrow BA^*$; pravila izvođenja ovog oblika se nazivaju *pravila preuređivanja* i ona omogućavaju ukrštanje interakcija između parova terminalnih simbola
- drugim rečima, u odnosu na regularne i kontekstno-slobodne, kontekstno-osetljive gramatike dozvoljavaju dodatna pravila koja omogućavaju modelovanje svih vrsta zavisnosti parova elemenata sekvenci

Sistematizacija modela struktura sekvenci

- svaka od klasa formalnih gramatika ima odgovarajući apstraktni računarski formalizam koji se naziva **automat**
- automati su apstraktne mašine koje obrađuju (parsiraju) sekvencu deo po deo primenjujući pravila gramatike
- **konačni automati** – sastoje se od konačnog broja stanja koja su međusobno povezana prelazima; stanja odgovaraju neterminalnim simbolima, a prelazi pravilima izvođenja formalnih gramatika
- klasa jezika koje prepoznaje konačni automat ekvivalentna je klasi jezika koje generišu (izvode) regularne gramatike
- **potisni automati** – za razliku od konačnih automata, koji ne zahtevaju nikakvu memoriju (osim za praćenje trenutnog stanja), potisni automati imaju (ograničenu) pomoćnu memoriju koja funkcioniše po principu steka (po čemu je automat i dobio ime)

Sistematizacija modela struktura sekvenci

- parsiranje sekvence se vrši tako što se na stek stavi početni neterminalni simbol gramatike, a zatim se u svakom narednom koraku skida po jedan simbol sa steka i u zavisnosti od toga da li je neterminalni ili terminalni simbol u pitanju, vrši se jedna od akcija
- klasa jezika koje prepoznaje potisni automat ekvivalentna je klasi jezika koje generišu kontekstno-slobodne gramatike
- **linearno-ograničeni automati** – apstraktna mašina koja se sastoji od trake podeljene na ćelije (koja predstavlja memoriju mašine) i glave koja može da čita/piše po ćelijama i da se pomera duž trake; dužina trake je linearno ograničena u odnosu na dužinu sekvence koja se parsira
- linearno-ograničeni automat prepoznaje kontekstno-osetljive gramatike
- **Tjuringova mašina** – isto što i linearno-ograničeni automat, samo sa neograničenom dužinom trake (memorije)
- Tjuringove mašine su ekvivalent sa gramatikama bez restrikcija

Sistematizacija modela struktura sekvenci

- svaka od formalnih gramatika iz hijerarhije Čomskog može se koristiti u stohastičkom obliku kao osnova za probabilističke modele struktura sekvenci
- stohastičke gramatike generišu neku sekvencu x sa nekom verovatnoćom $P(x)$, dok nestohastičke gramatike ili generišu sekvencu x ili ne
- drugim rečima, stohastičke gramatike definišu raspodelu verovatnoća nad sekvencama x , tj. $\sum_x P(x) = 1$
- u stohastičkoj varijanti regularnih i kontekstno-slobodnih gramatika zbir verovatnoća svih mogućih pravila izvođenja iz bilo kog neterminalnog simbola mora biti 1
- pravila izvođenja i njima pridružene verovatnoće za stohastičke verzije kontekstno-osetljivih i gramatika bez restrikcija moraju biti formulisana pažljivije – potrebno je za svaku sekvencu obezbediti jedinstveno izvođenje

Sistematizacija modela struktura sekvenci

- to se postiže definisanjem pravila gramatike tako da kontekst u kome se pojedinačni neterminalni simbol pojavljuje jednoznačno određuje skup mogućih pravila izvođenja koja se u tom slučaju mogu primeniti, odnosno da za svaki neterminalni simbol ne postoji više od jedne forme leve strane pravila izvođenja u kome se on pojavljuje
- pridruživanjem verovatnoća pravilima izvođenja tako da se sumiraju na 1 za bilo koji neterminalni simbol u svakom mogućem kontekstu, dobija se stohastička gramatika
- u nastavku će detaljnije biti razmatrani samo modeli koji se zasnivaju na stohastičkim regularnim i kontekstno-slobodnim gramatikama, jer samo oni imaju praktičnu primenu u bioinformatici

Sistematizacija modela struktura sekvenci

- pored (stohastičke) gramatike, model strukture sekvenci definišu još i algoritmi za:
 - 1) određivanje optimalnih vrednosti parametara modela, tj. pridruživanje verovatnoća pravilima izvođenja tako da opisuju strukturu sekvence koja se modeluje
 - 2) poređenje (poravnanje) struktura novih sekvenci prema strukturi sekvence koju dati model modeluje, što odgovara određivanju optimalnog izvođenja za novu sekvencu
 - 3) kvantifikovanje sličnosti struktura novih sekvenci sa strukturom sekvence koju dati model modeluje, što odgovara izračunavanju verovatnoće da nova sekvenca bude generisana datom gramatikom

Sistematizacija modela struktura sekvenci

KLASA GRAMATIKA	REGULARNE GRAMATIKE	KONTEKSTNO- SLOBODNE GRAMATIKE	KONTEKSTNO- OSETLJIVE GRAMATIKE	GRAMATIKE BEZ OGRANIČENJA
ODGOVARAJUĆI ANALITIČKI FORMALIZMI	KONAČNI AUTOMATI	POTISNI AUTOMATI	LINEARNO- OGRANIČENI AUTOMATI	TJURINGOVA MAŠINA
ODGOVARAJUĆI STOHAISTIČKI MEHANIZMI	STOHAISTIČKE REGULARNE GRAMATIKE HMM	STOHAISTIČKE KONTEKSTNO- SLOBODNE GRAMATIKE SCFG (PCFG)	STOHAISTIČKE KONTEKSTNO- OSETLJIVE GRAMATIKE SCSG (PCSG)	STOHAISTIČKE GRAMATIKE BEZ OGRANIČENJA
DISKRIMINATIVNI ANALOGON	LINEAR CHAIN CRF	CSCFG (CPCFG)	—	—

Pregled

- Vrste grafovskih probabilističkih modela
- Sistematizacija modela struktura sekvenci
- **HMM**
- SCFG
- Struktura RNK
- Opis implementacije

HMM

- **HMM (eng. Hidden Markov Model)** je ekvivalent stohastičkim regularnim gramatikama
- najčešće se predstavlja i definiše kao probabilistički konačni automat, tj. graf koji se sastoji od konačnog skupa čvorova (stanja) koja su međusobno povezana granama (prelazima) sa pridruženim verovatnoćama, pri čemu se iz jednog stanja sa određenom verovatnoćom može preći u drugo stanje, a iz svakog od stanja se po dolasku emituje simbol sekvence sa određenom verovatnoćom
- kako se iz svakog stanja u opštem slučaju može emitovati bilo koji od simbola (sa različitim verovatnoćama), znajući sekvencu nije moguće odrediti koji simbol je generisan iz kog stanja – sekvenca stanja iz kojih je generisana sekvenca simbola je **skrivena** (otuda potiče naziv modela)

HMM

- dakle, HMM je određen grafom (skupom stanja i prelaza) i sledećim skupom parametara:
 - a_{kl} – verovatnoće prelaska iz stanja k u stanje l
 - $e_k(b)$ – verovatnoće emitovanja simbola b iz stanja k
- kako bi se jasnije razlikovalo kada se govori o sekvencama simbola a kada o sekvencama stanja, sekvence stanja se radije nazivaju **putanjama** (kroz graf stanja), dok se sekvence simbola nazivaju kraće samo sekvencama
- zajednička raspodela sekvence x i njoj odgovarajuće putanje π može se dobiti kao

$$P(x, \pi) = a_{0\pi_1} \prod_{i=1}^L e_{\pi_i}(x_i) a_{\pi_i \pi_{i+1}}$$

gde je sa 0 označeno početno stanje, dok je L dužina date sekvence

HMM

- kako putanja u većini slučajeva nije poznata, a upravo je ona od interesa*, od svih mogućih putanja najsmislenije je odabrati onu optimalnu – za koju se dobija najveća vrednost zajedničke verovatnoće

$$\pi^* = \underset{\pi}{\operatorname{argmax}} P(x, \pi)$$

- optimalna putanja π^* može se odrediti rekurzivno, dinamičkim programiranjem
- pretpostavimo da su poznate verovatnoće $v_k(i)$ optimalne putanje za deo sekvence do $(i-1)$ -og elementa (uključujući i njega), a koje se završavaju u stanju k
- tada se verovatnoća optimalne putanje za deo sekvence zaključno do i -tog elementa, a koja se završava u stanju l , može izračunati kao

$$v_l(i) = e_l(x_i) \max_k (v_k(i-1) a_{kl})$$

HMM

Algorithm: Viterbi

Initialization ($i = 0$): $v_0(0) = 1, v_k(0) = 0$ for $k \neq 0$

Recursion ($i = 1, \dots, L$): $v_l(i) = e_l(x_i) \max_k (v_k(i-1)a_{kl})$
 $ptr_i(l) = \operatorname{argmax}_k (v_k(i-1)a_{kl})$

Termination: $P(x, \pi^*) = \max_k (v_k(L)a_{k0})$
 $\pi_L^* = \operatorname{argmax}_k (v_k(L)a_{k0})$

Traceback ($i = L, \dots, 1$): $\pi_{i-1}^* = ptr_i(\pi_i^*)$

HMM

- ukoliko želimo da odredimo kolika je verovatnoća da neka sekvenca x bude generisana datim modelom, tada treba uzeti u obzir sve moguće putanje

$$P(x) = \sum_{\pi} P(x, \pi)$$

- to se može izračunati algoritmom koji je analogan Viterbijevom algoritmu, s tim što se umesto maksimuma računa zbir
- u zavisnosti da li se taj zbir izračunava posmatranjem podsekvenci s početka ili s kraja date sekvence x , imamo sledeća dva algoritma:

Algorithm: Forward algorithm

Initialization ($i = 0$): $f_0(0) = 1, v_k(0) = 0$ for $k \neq 0$

Recursion ($i = 1, \dots, L$): $f_l(i) = e_l(x_i) \sum_k f_k(i-1) a_{kl}$

Termination: $P(x) = \sum_k f_k(L)$

Algorithm: Backward algorithm

Initialization ($i = L$): $b_k(L) = 1$ for all k

Recursion ($i = L-1, \dots, 1$): $b_k(i) = \sum_l a_{kl} e_l(x_{i+1}) b_l(i+1)$

Termination: $P(x) = \sum_l a_{0l} e_l(x_1) b_l(1)$

HMM

- prethodno navedena tri algoritma zapravo predstavljaju deo samog modela HMM – Viterbijev algoritam odgovara stavki (2), a algoritmi unapred i unazad stavki (3) sa slajda 16
- ostaje još da vidimo na koji način se dolazi do optimalnih vrednosti parametara HMM modela (stavka (1))
- kada su putanje poznate može se izbrojati koliko puta je svaki od prelaza korišćen, kao i koliko puta je koji simbol emitovan iz svakog od stanja – neka su sa A_{kl} i $E_k(b)$ označene redom te vrednosti
- tada se vrednosti parametara modela a_{kl} i $e_k(b)$ mogu oceniti metodom maksimalne verodostojnosti kao

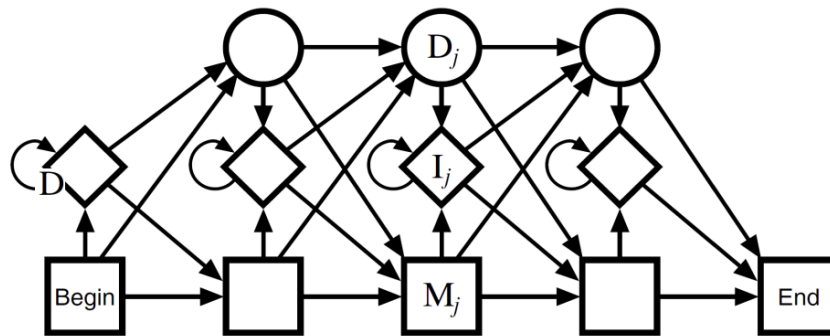
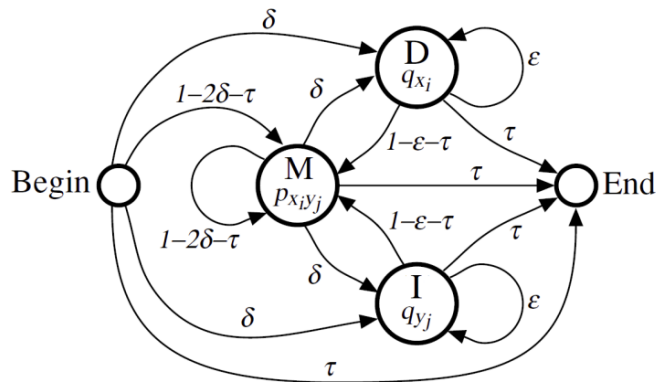
$$a_{kl} = \frac{A_{kl}}{\sum_{l'} A_{kl'}} \quad e_k(b) = \frac{E_k(b)}{\sum_{b'} E_k(b')} \quad (1)$$

HMM

- kada putanje nisu poznate ne postoji direktna jednačina za ocenu vrednosti parametara, već se mora koristiti neki oblik iterativne procedure (optimizacioni algoritam)
- ovi algoritmi iterativno ocenjuju A_{kl} i $E_k(b)$ na osnovu trenutnih vrednosti a_{kl} i $e_k(b)$ uzimajući u obzir sve moguće putanje za svaku od trening sekvenci, a zatim na osnovu jednačina (1) izračunavaju nove vrednosti za a_{kl} i $e_k(b)$
- **Baum-Welch algoritam** A_{kl} i $E_k(b)$ izračunava kao očekivani broj odgovarajućih prelaza i emisija koji su korišćeni za generisanje datog skupa trening sekvenci
- **algoritam Viterbijevog učenja** do ocena za A_{kl} i $E_k(b)$ dolazi na osnovu optimalnih putanja trening sekvenci dobijenih pomoću Viterbijevog algoritma za tekuće vrednosti parametara modela

HMM

- HMM modeli imaju široku primenu, kako u bioinformatici tako i u mnogim drugim oblastima
- najvažnija primena HMM u bioinformatici je za modelovanje primarne strukture pojedinačnih sekvenci i familija sekvenci, odnosno za jednostruko i višestruko poravnanje sekvenci



Pregled

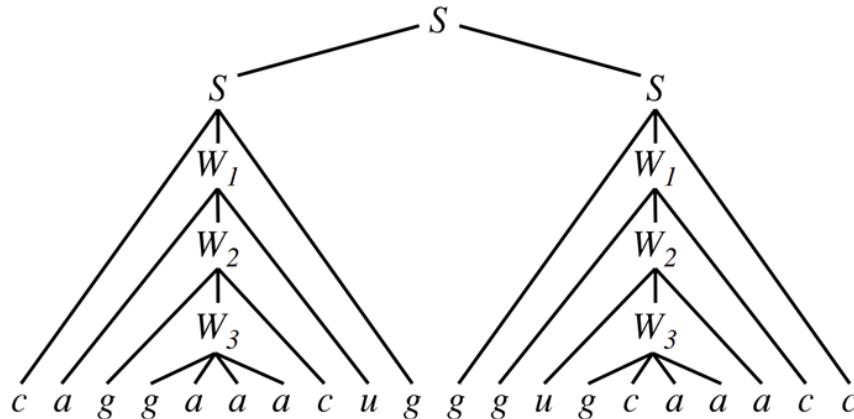
- Vrste grafovskih probabilističkih modela
- Sistematizacija modela struktura sekvenci
- HMM
- SCFG
- Struktura RNK
- Opis implementacije

SCFG

- za razliku od primarne strukture, modelovanje sekundarne strukture mnogih bioloških sekvenci zahteva modelovanje ugnežđenih, dugoročnih zavisnosti elemenata sekvenci
- kontekstno-slobodne gramatike modeluju upravo ovakvu vrstu međuzavisnosti
- pandan HMM modelima su tzv. SCFG modeli, modeli koji se zasnivaju na stohastičkim kontekstno-slobodnim gramatikama
- isto kao HMM modeli, i SCFG modeli su pored gramatike određeni algoritmima (1), (2) i (3) sa slajda 16
- pre nego što damo opise ovih algoritama uvešćemo pojmove **stabala parsiranja** i **normalne forme** kontekstno-slobodnih gramatika

SCFG

- kod kontekstno-slobodnih gramatika, izvođenja (parsiranja) sekvenci imaju strukturu stabla – početni neterminalni simbol gramatike odgovara korenu stabla, ostali neterminalni simboli unutrašnjim čvorovima, a terminalni simboli (elementi sekvence) listovima stabla, dok grane između čvorova odgovaraju pravilima izvođenja



$S \rightarrow SS$
 $S \rightarrow aW_1u \mid cW_1g \mid gW_1c \mid uW_1a$
 $W_1 \rightarrow aW_2u \mid cW_2g \mid gW_2c \mid uW_2a$
 $W_2 \rightarrow aW_3u \mid cW_3g \mid gW_3c \mid uW_3a$
 $W_3 \rightarrow gaaa \mid gcaa$

SCFG

- kako desna strana pravila izvođenja kod kontekstno-slobodnih gramatika može imati proizvoljnu formu, da bi se mogli formulisati opšti algoritmi za parsiranje sekvenci potrebno je nametnuti neku vrstu normalne forme za pravila izvođenja
- jedna takva normalna forma je **normalna forma Čomskog** koja zahteva da su sva pravila izvođenja oblika $W_v \rightarrow W_y W_z$ ili $W_v \rightarrow a$
- svaka kontekstno-slobodna gramatika se može transformisati u normalnu formu zamenom pojedinačnih pravila izvođenja (koja nisu u normalnoj formi) nizom pravila izvođenja odgovarajućeg oblika uz pomoć dodatnih neterminalnih simbola
- dakle, svi algoritmi koji se definišu za kontekstno-slobodne gramatike u normalnoj formi Čomskog generalno su primenjivi na bilo koju kontekstno-slobodnu gramatiku

SCFG

- nadalje posmatramo stohastičku kontekstno-slobodnu gramatiku u normalnoj formi Čomskog sa skupom neterminalnih simbola $\{W_1, \dots, W_M\}$, gde je sa W_1 označen početni neterminal
- pravila izvođenja su, dakle, oblika $W_v \rightarrow W_y W_z$ ili $W_v \rightarrow a$
- neka su verovatnoće pridružene ovim pravilima izvođenja (verovatnoća tranzicije iz stanja W_v u stanja W_y i W_z i verovatnoća emisije simbola a iz stanja W_v) označene redom sa $t_v(y, z)$ i $e_v(a)$
- **algoritam iznutra** izračunava verovatnoću $\alpha(i, j, v)$ stabla parsiranja za podsekvencu $x_i \dots x_j$ sa korenom u W_v
- izračunavanje počinje od podsekvencama dužine 1 ($i = j$) i rekurzivno ide ka sve dužim i dužim podsekvencama, sve dok se ne odredi verovatnoća stabla parsiranja za kompletnu sekvencu x sa korenom u W_1

SCFG

Algorithm: Inside

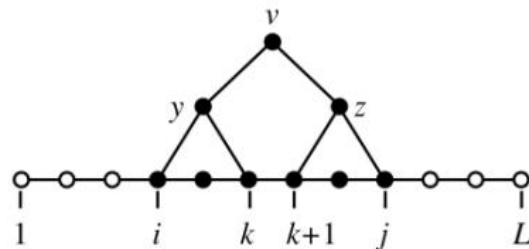
Initialization: for $i = 1$ to L , $v = 1$ to M

$$\alpha(i, i, v) = e_v(x_i)$$

Iteration: for $i = 1$ to $L - 1$, $j = i + 1$ to L , $v = 1$ to M

$$\alpha(i, j, v) = \sum_{y=1}^M \sum_{z=1}^M \sum_{k=i}^{j-1} t_v(y, z) \alpha(i, k, y) \alpha(k + 1, j, z)$$

Termination: $P(x) = \alpha(1, L, 1)$



SCFG

- dakle, algoritam iznutra izračunava ukupnu verovatnoću parsiranja (izvođenja) sekvence x , uzimajući u obzir sva moguća stabla parsiranja (izvođenja) za datu sekvencu
- isto to se može dobiti **algoritmom spolja** koji rekurzivno izračunava verovatnoću $\beta(i, j, v)$ stabla parsiranja za sekvencu x sa korenom u W_1 isključujući sva podstabla sa korenom u W_v koja generišu podsekvencu $x_i \dots x_j$
- izračunavanje počinje od najveće isključene podsekvence $x_1 \dots x_L$ i rekurzivno ide ka isključivanju sve kraćih podsekvenci, sve dok se ne odredi verovatnoća stabla parsiranja za kompletnu sekvencu x
- algoritmi iznutra i spolja za SCFG modele su pandan algoritmima unapred i unazad za HMM modele

SCFG

Algorithm: Outside

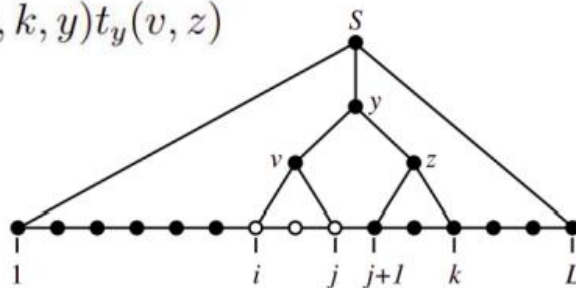
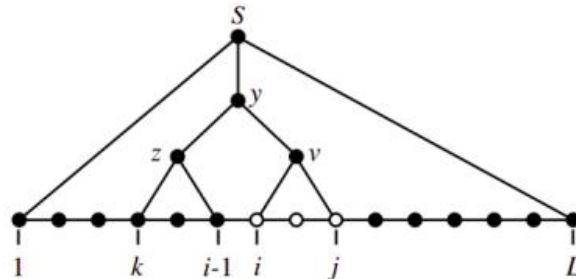
Initialization: $\beta(1, L, 1) = 1$
 $\beta(1, L, v) = 0$ for $v = 2$ to M

Iteration: for $i = 1$ to L , $j = L$ to i , $v = 1$ to M

$$\beta(i, j, v) = \sum_{y,z} \sum_{k=i}^{i-1} \alpha(k, i-1, z) \beta(k, j, y) t_y(z, v) +$$

$$\sum_{y,z} \sum_{k=j+1}^L \alpha(j+1, k, z) \beta(i, k, y) t_y(v, z)$$

Termination: $P(x) = \sum_{v=1}^M \beta(i, i, v) e_v(x_i)$ for any i



SCFG

- pandan Viterbijevom algoritmu za HMM modele je **CYK (Cocke-Younger-Kasami) algoritam** za SCFG modele koji pronalazi optimalno stablo parsiranja za datu sekvencu
- CYK algoritam izračunava verovatnoću $\gamma(i, j, v)$ optimalnog stabla parsiranja za podsekvencu $x_i \dots x_j$ sa korenom u W_v
- pored toga, čuvaju se tzv. traceback promenljive $\tau(y, z, k)$ koje zapravo predstavljaju trojke (y, z, k) potrebne za rekonstrukciju optimalnog stabla parsiranja
- u nastavku je prikazan algoritam CYK zajedno sa CYK traceback algoritmom koji tehnikom bektrekinga i korišćenjem pomoćne memorije u vidu steka rekonstruiše optimalno stablo parsiranja

SCFG

Algorithm: CYK

Initialization: for $i = 1$ to L , $v = 1$ to M

$$\gamma(i, i, v) = e_v(x_i)$$

$$\tau(i, i, v) = (0, 0, 0)$$

Iteration: for $i = 1$ to $L - 1$, $j = i + 1$ to L , $v = 1$ to M

$$\gamma(i, j, v) = \max_{y, z} \max_{i \leq k \leq j-1} t_v(y, z) \gamma(i, k, y) \gamma(k + 1, j, z)$$

$$\tau(i, j, v) = \underset{y, z, i \leq k \leq j-1}{\operatorname{argmax}} t_v(y, z) \gamma(i, k, y) \gamma(k + 1, j, z)$$

Termination: $P(x) = \gamma(1, L, 1)$

Algorithm: CYK traceback

Initialization: push $(1, L, 1)$ on the stack

Iteration: pop (i, j, v)

$$(y, z, k) = \tau(i, j, v)$$

if $\tau(i, j, v) = (0, 0, 0)$ (implying $i = j$)

attach x_i as the child of v

else

attach y, z to parse tree as children of v

push $(k + 1, j, z)$

push (i, k, y)

SCFG

- u tabeli koja sledi sumirani su i upoređeni svi dosad predstavljeni algoritmi
- oznake: x (sekvenca), θ (parametri), L (dužina sekvence), M (broj pravila)

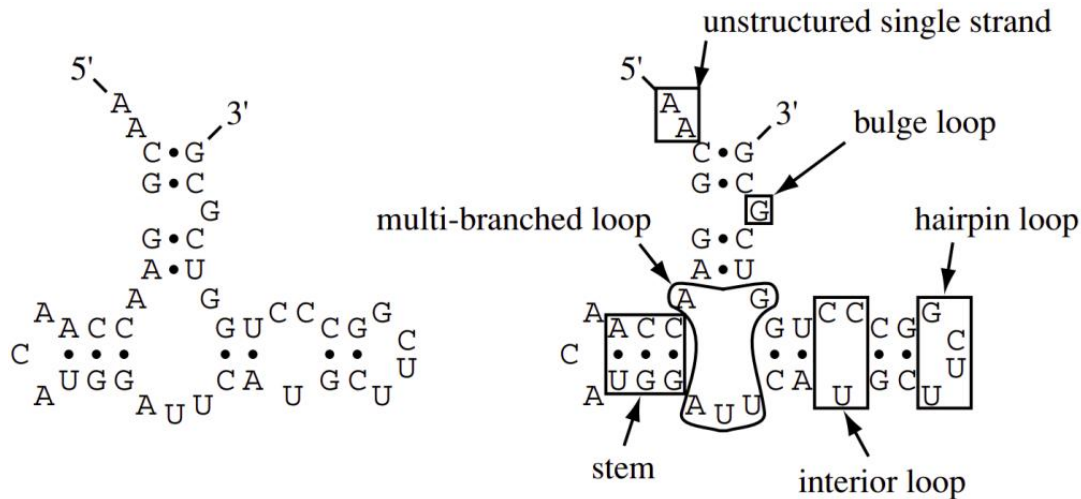
Goal	HMM algorithm	SCFG algorithm
optimal alignment	Viterbi	CYK
$P(x \theta)$	forward	inside
EM parameter estimation	forward–backward	inside–outside
memory complexity:	$O(LM)$	$O(L^2M)$
time complexity:	$O(LM^2)$	$O(L^3M^3)$

Pregled

- Vrste grafovskih probabilističkih modela
- Sistematizacija modela struktura sekvenci
- HMM
- SCFG
- **Struktura RNK**
- Opis implementacije

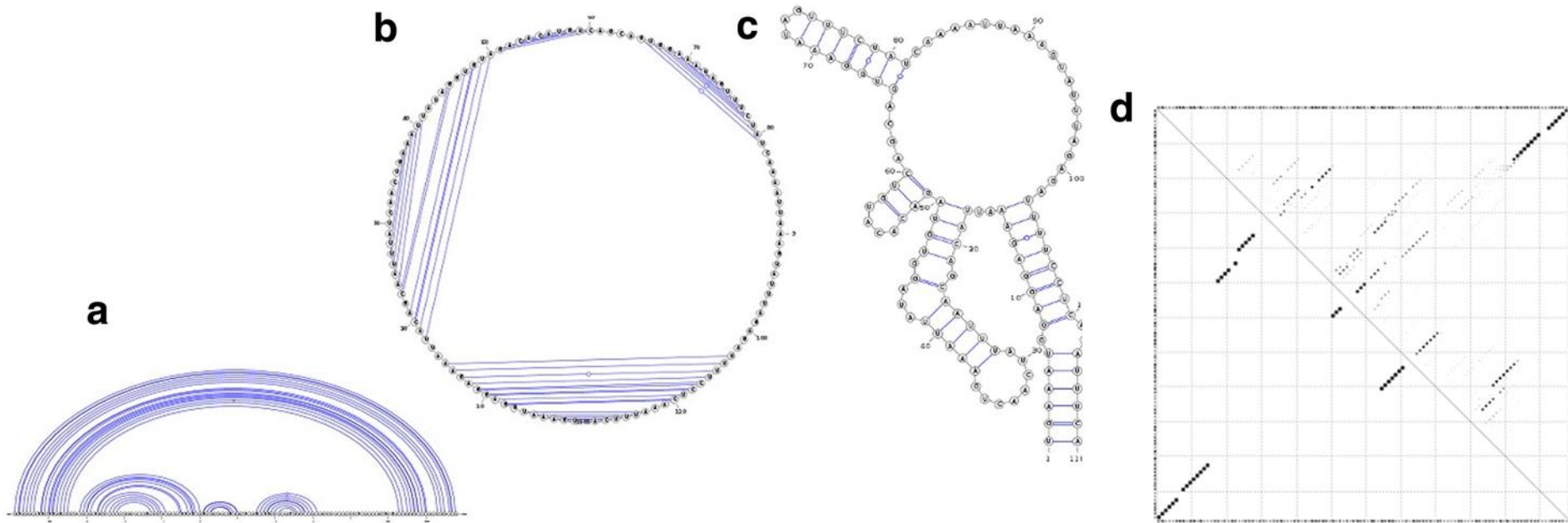
Struktura RNK

- RNK molekuli se tipično sastoje od jednog lanca nukleotidnih baza koji može da se savija intramolekularno i formira segmente uparenih nukleotidnih baza
- bazni parovi A – U i G – C su kanonski (prema Votsonu i Kriku), ali su moguće i druge (nekanonske) varijante uparivanja, pri čemu se posebno izdvaja U – G
- struktura formiranih baznih parova naziva se **sekundarna struktura RNK**



Struktura RNK

- još neki načini za grafičku reprezentaciju sekundarne strukture RNK



Struktura RNK

- za RNK je karakteristično to da homologne sekvence (istog porekla, sa zajedničkim evolutivnim pretkom) imaju sličnu sekundarnu strukturu, dok im primarne strukture ne moraju imati značajne sličnosti
- drastične promene (mutacije) u primarnoj strukturi sekvenci mogu se tolerisati sve dok kompenzacione mutacije održavaju uparivanja baza na odgovarajućim pozicijama
- to znači da sekundarna struktura RNK evoluira (mutira) sporije od primarne strukture, što modele sekundarne strukture čini podesnim za traženje homologija kod RNK sekvenci
- kako se bazni parovi skoro uvek javljaju na ugnežđeni način u sekundarnoj strukturi RNK, a kontekstno-slobodne gramatike modeluju upravo ovakav tip međuzavisnosti, to SCFG modele čini najprikladnijim izborom za probabilističko modelovanje sekundarne strukture RNK

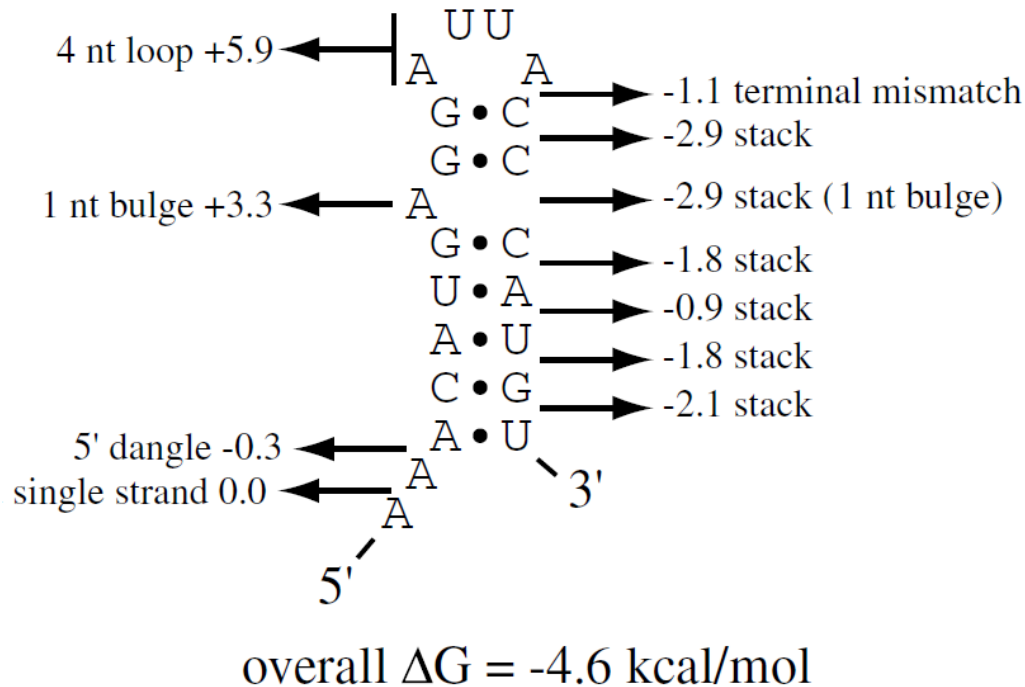
Struktura RNK

- postoje različiti pristupi problemu modelovanja sekundarne strukture RNK sekvenci pomoću SCFG modela
- jedan mogući pristup je pronalaženje strukture sa najviše baznih parova
- **algoritam Nussinov** i njemu odgovarajući SCFG model imaju upravo ovakav pristup
- mane ovog pristupa su to da ne uzima u obzir važne strukturne karakteristike kao što su preferencije ka određenim dužinama petlji ili preferencije ka određenim kombinacijama susednih baznih parova

Algorithm: Nussinov RNA folding, fill stage

$$\gamma(i, j) = \max \begin{cases} \gamma(i + 1, j), \\ \gamma(i, j - 1), \\ \gamma(i + 1, j - 1) + \delta(i, j), \\ \max_{i < k < j} [\gamma(i, k) + \gamma(k + 1, j)] \end{cases}$$

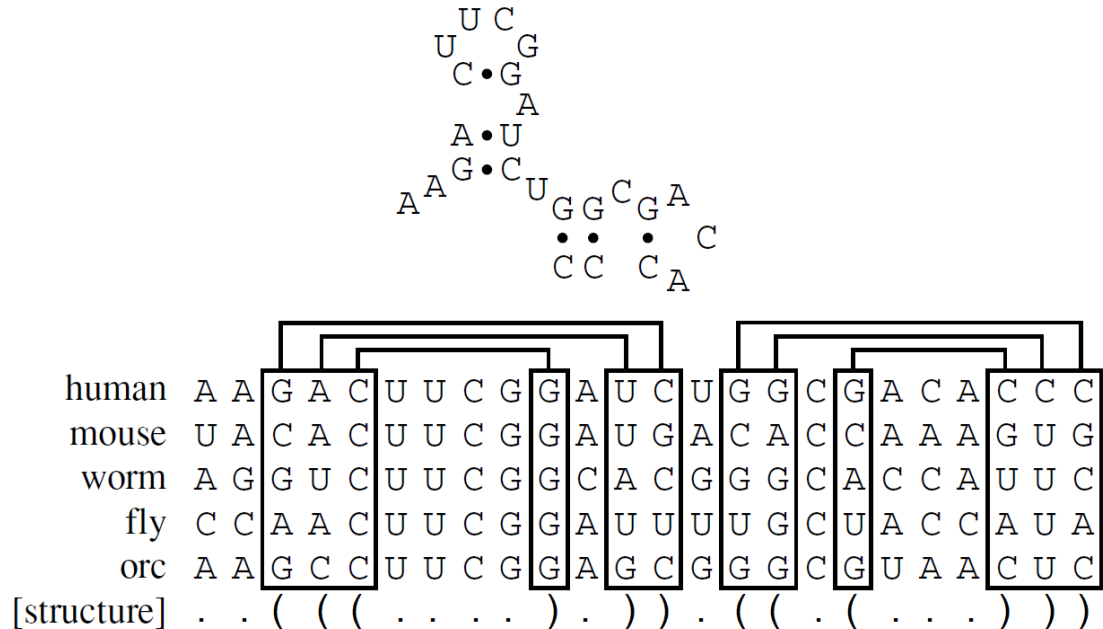
Struktura RNK



- drugačiji pristup se zasniva na tome da intramolekularno savijanje RNK diktiraju biofizički procesi
- najsofisticiraniji metod za predviđanje sekundarne strukture pojedinačnih RNK sekvenci je **Zukerov termodinamički model** i njemu odgovarajuća SCFG, koji pretpostavljaju da je optimalna struktura ona sa najnižom slobodnom energijom

Struktura RNK

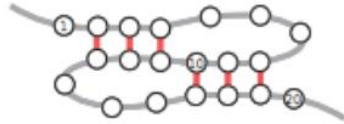
- za modelovanje sekundarne strukture familija RNK sekvenci koriste se tzv. **modeli kovarijacije**, kontekstno-slobodni pandan profilnih HMM modela
- za razliku od profilnih HMM modela, koje karakteriše repetitivna linearna arhitektura, kovarijacioni modeli imaju repetitivnu drvoliku arhitekturu koja je pogodna za modelovanje konsenzusnih sekundarnih stuktura familije RNK sekvenci



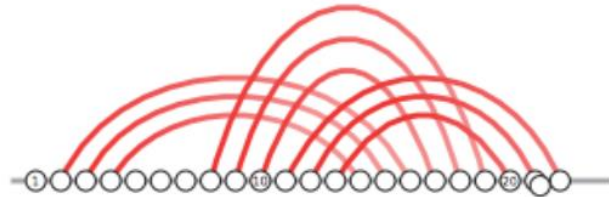
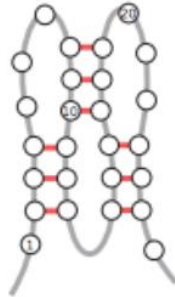
Struktura RNK



H-type pseudoknot



Three chain (kissing hairpin)



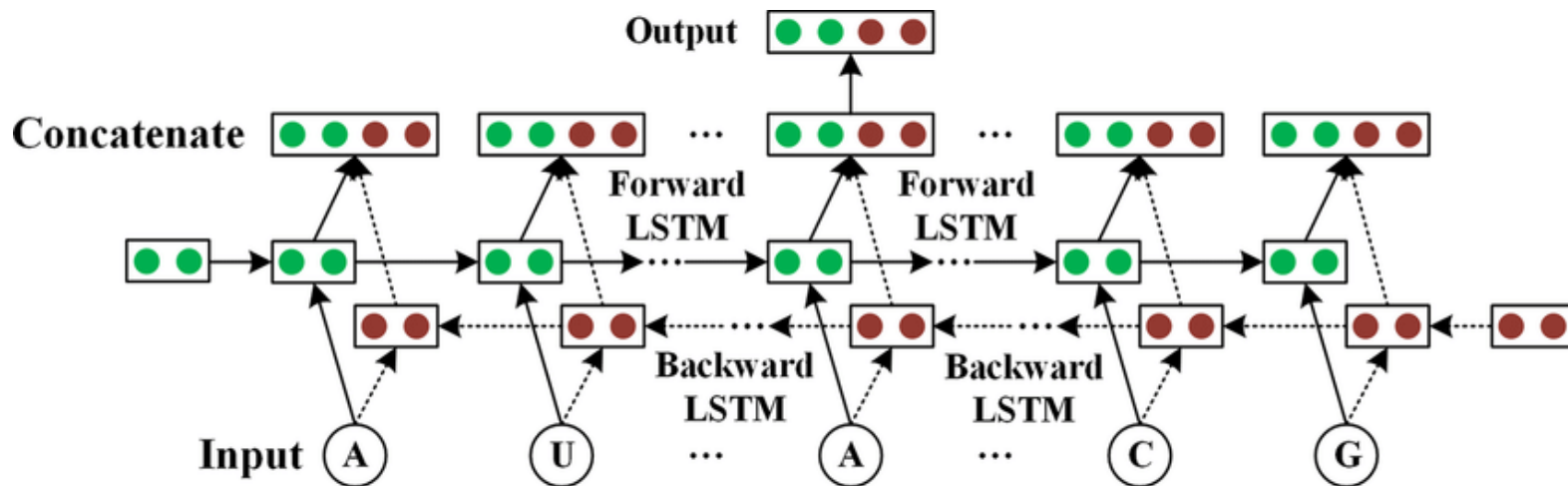
Three knot



- pored generativnih PCFG, moguće je koristiti i diskriminativne CPCFG, koji pridružuju attribute (npr. prefiks podsekvence) pravilima izvođenja
- osnovna mana predstavljenih modela je u tome što ne modeluju neugneždene interakcije (**pseudočvorove**)
- postoje određeni algoritmi dinamičkog programiranja koji mogu da predvide pojedine tipove pseudočvorova, ali su znatno veće složenosti
- oni se stoga najčešće predviđaju metodama homologije

Struktura RNK

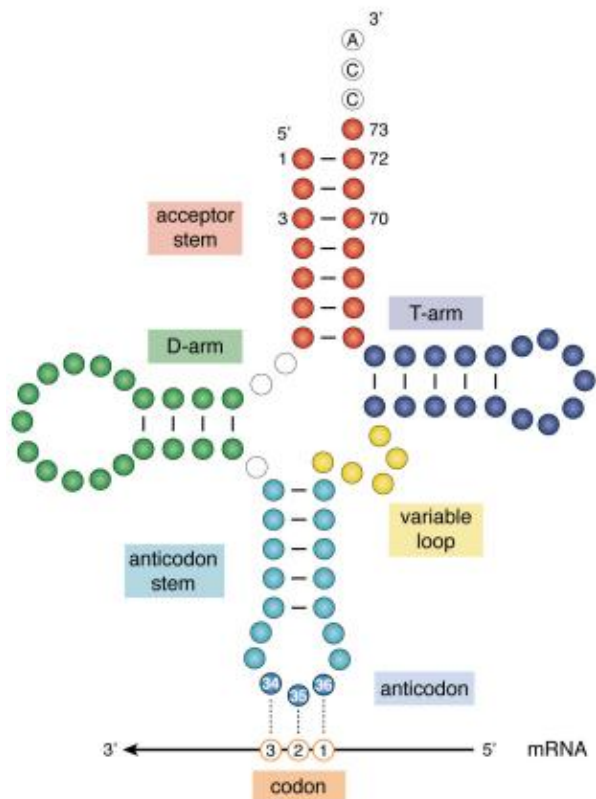
- postoje i drugačiji pristupi, a savremeni su zasnovani na **dubokom učenju**, koje je u stanju da uoči najrazličitije vrste međuzavisnosti
- dosad su se najbolje pokazale složene neuronske mreže tipa Bi-LSTM, a u budućnosti se očekuje šira upotreba modela ove vrste



Pregled

- Vrste grafovskih probabilističkih modela
- Sistematizacija modela struktura sekvenci
- HMM
- SCFG
- Struktura RNK
- Opis implementacije

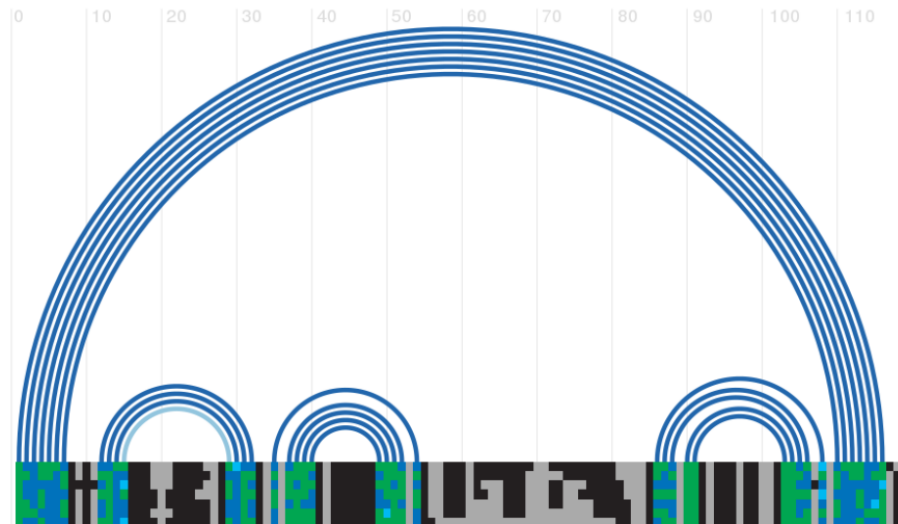
Opis implementacije



- posebno zanimljiva vrsta RNK je **transportna RNK**, sa ulogom u translaciji proteina i karakterističnom sekundarnom strukturom u obliku deteline sa tri lista
- listove (petlje, *loop*) čine neuparene baze, dok se kao veza između njih nalaze četiri zavojnice (držke, *stem*) i umetnuti nukleotidi (V petlja, pseudočvor...)
- na drugom listu, otprilike u sredini sekvence, nalazi se antikodon aminokiseline koja se prenosi

Opis implementacije

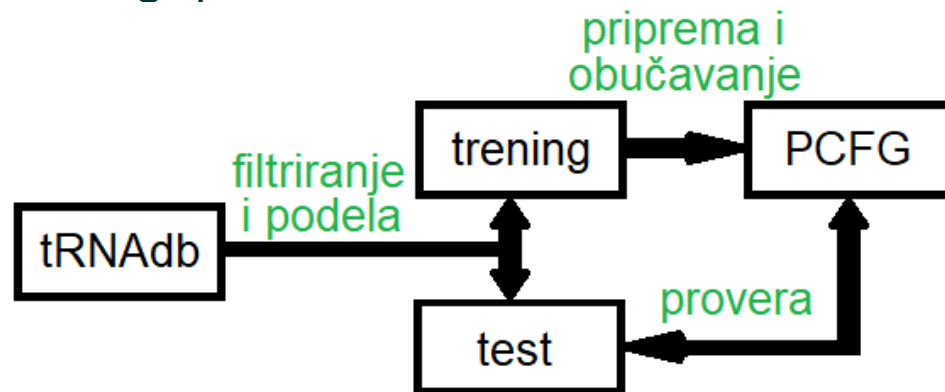
- u okviru rada na temi predviđanja sekundarne strukture tRNK, implementirana su tri SCFG modela s tim ciljem
- korišćeni skup podataka preuzet je iz baze **tRNAdb**, koja čuva sekvence tRNK sa pridruženim sekundarnim strukturama



Acc-stem	D-stem	D-loop	D-stem	Ac-stem	Ac-loop	Ac-stem	V-region			T-stem	T-loop	T-stem	Acc-stem	CCA
-1 1	8 10	14	22	26 27	32	39	44			49	53	61	66	73 74
- GGGCCCG	UA GCUC	AGCCAGGAC - A	GAGC G	CCGGC	CUCUCAA	GCCGG	UG-----	-----CUG	CCGGG	UUCAAAU	CCCGG	CGGGCCC	G	CCA
- GCCGCGG	UA GUAU	AGCCUGGACUA	GUAU G	CGGGC	CUGUCA	GCCCG	UG-----	-----A-C	CCGGG	UUCAAAU	CCCGG	CCGCGGC	G	CCA
- GCCGGGG	UG GCCG	AGC--GGUCUA	AGGC G	GCGGG	CUGCAGA	CCCGU	UA-----	-----UUC	CCGGG	UUCGAU	CCCGG	CCCCGGC	U	CCA
- GGGCCCG	UA GCUC	AGCCUGGU--A	GAGC G	GCGGG	CUCUUA	CCCGC	GAGG-----	-----AAGUC	CCGGG	UUCAAAU	CCCGG	CGGGCCC	G	CCA
- GGGCCCG	UA GCUC	AGCCCGGC--A	GAGC G	GCGGG	CUUUUA	CCCGC	GG-----	-----AAG	CCGGG	UUCAAAU	CCCGG	CGGGCCC	G	CCA
- GGGCCCG	UA GCUC	AGCCAGGU--A	GAGC G	CCCGG	CUCAUA	CCGGG	UG-----	-----GUC	GGGGG	UUCAAAU	CCCCC	CGGGCCC	A	CCA

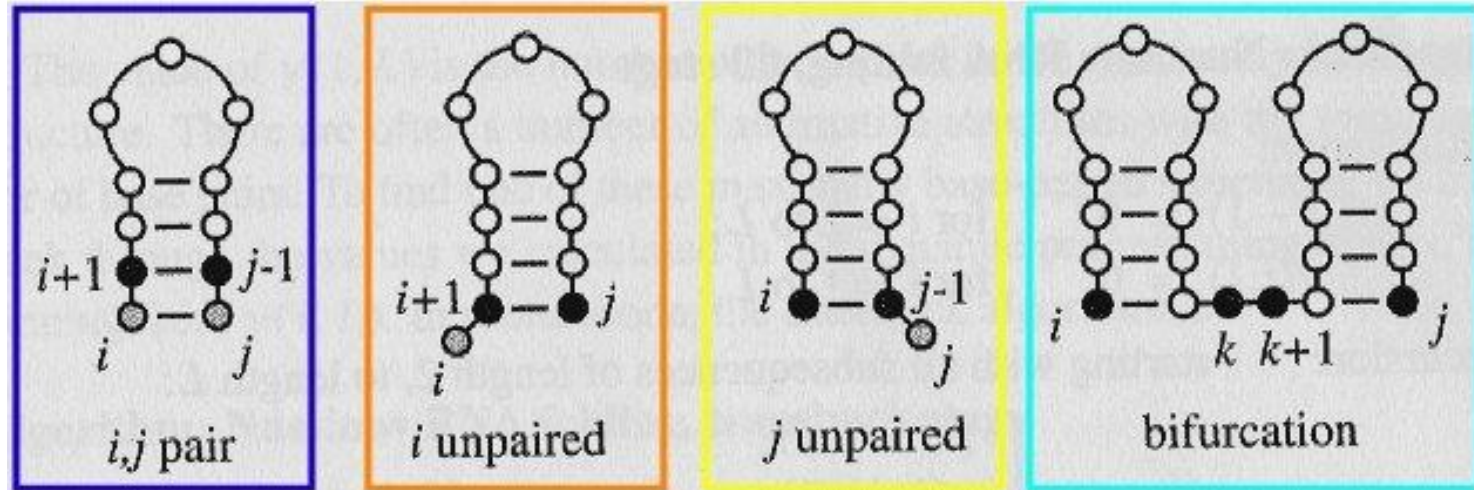
Opis implementacije

- nakon filtriranja, u skupu su ostale 432 sasvim korektne sekvence, koje su iskorišćene nadalje u obučavanju i proveru implementiranih gramatika
- modeli su trenirani na jednom delu skupa, koji je prethodno morao biti transformisan u odgovarajući skup stabala izvođenja, što je i učinjeno modifikovanom tehnikom rekurzivnog spusta
- gotovi modeli iskorišćeni su za predviđanje sekundarne strukture drugog dela podataka (ukupno 108 test instanci), što je poslužilo za evaluaciju



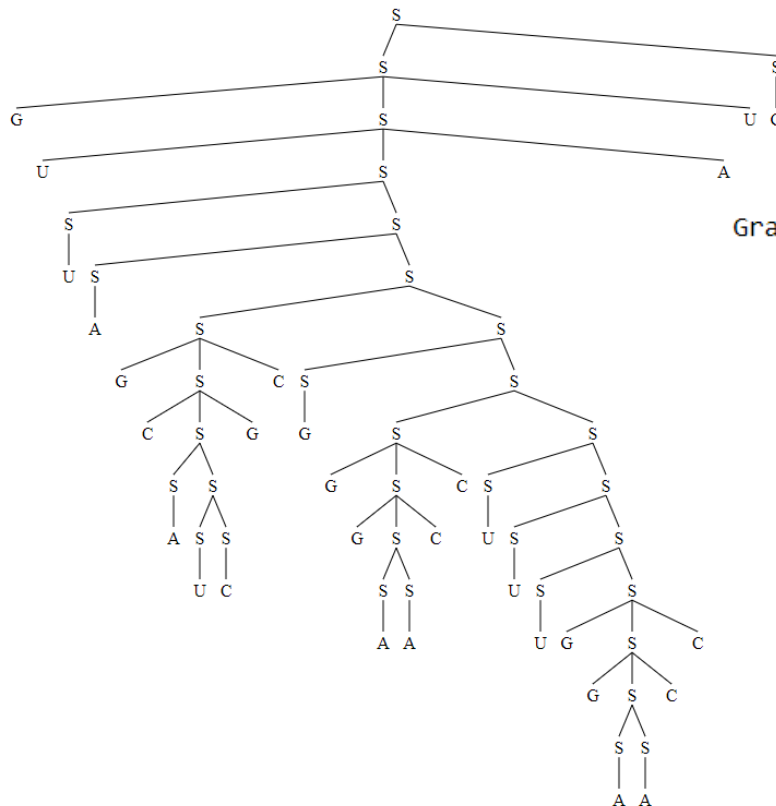
Opis implementacije

- prvi implementirani model je tipa **Nusinov**, koji se zasniva na gramatici sa pravilima izvođenja $S \rightarrow dSd$ (uparivanje) $| SS$ (grananje) $| d$ (neuparena baza)



Opis implementacije

- kako on maksimizuje broj uparivanja, a to najčešće ne odgovara stvarnosti, ovaj model ima samo teorijski značaj
- na slikama je prikazan primer stabla izvođenja i parametri obučene gramatike, a tako će biti i u nastavku



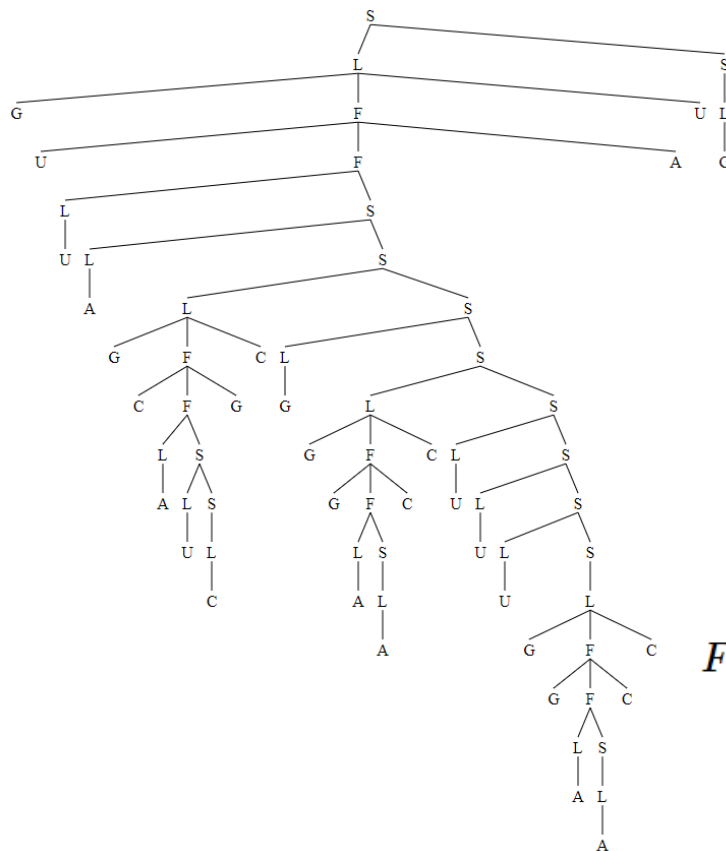
Grammar with 11 productions

```
S -> 'G' S 'C' [0.0905911]
S -> 'C' S 'G' [0.0617045]
S -> 'U' S 'A' [0.0303525]
S -> 'A' S 'U' [0.0269874]
S -> 'G' S 'U' [0.00922903]
S -> 'U' S 'G' [0.00496435]
S -> S S [0.382688]
S -> 'A' [0.112048]
S -> 'U' [0.110515]
S -> 'G' [0.0900913]
S -> 'C' [0.0808289]
```

Opis implementacije

Grammar with 19 productions

$S \rightarrow L S$ [0.860373]
 $S \rightarrow L$ [0.139627]
 $L \rightarrow 'G' F 'C'$ [0.050552]
 $L \rightarrow 'C' F 'G'$ [0.0192615]
 $L \rightarrow 'G' F 'U'$ [0.0124096]
 $L \rightarrow 'U' F 'A'$ [0.00989722]
 $L \rightarrow 'A' F 'U'$ [0.00738485]
 $L \rightarrow 'U' F 'G'$ [0.00137038]
 $L \rightarrow 'A'$ [0.256033]
 $L \rightarrow 'U'$ [0.252531]
 $L \rightarrow 'G'$ [0.205862]
 $L \rightarrow 'C'$ [0.184697]
 $F \rightarrow 'G' F 'C'$ [0.305895]
 $F \rightarrow 'C' F 'G'$ [0.238017]
 $F \rightarrow 'U' F 'A'$ [0.116255]
 $F \rightarrow 'A' F 'U'$ [0.106133]
 $F \rightarrow 'U' F 'G'$ [0.0194999]
 $F \rightarrow 'G' F 'U'$ [0.0169693]
 $F \rightarrow L S$ [0.197231]



- realizovana je i gramatika **KH-99**, koja uspešno predviđa opštu formu uvijanja, ali ipak nedovoljno tačno

$S \rightarrow LS$ (nizanje elemenata) | L (poslednji element)

$L \rightarrow s$ (neuparena baza) | dFd (početak zavoynice)

$F \rightarrow dFd$ (nastavak zavoynice) | LS (unutrašnjost zavoynice)

Opis implementacije

- naposljetku je implementiran svojevrсни **model kovarijacije**, koji eksploatiše postojanje dobro očuvanog skeleta strukture u familiji koja se modeluje, što je u ovom slučaju familija tRNAK
- u njoj postoje tačno tri važne petlje, tačno četiri zavojnice i još neki dodatni elementi, ali takođe fiksnog sadržaja

$$G \rightarrow S \text{ (glava je tačno jedan nukleotid)}$$

$$R \rightarrow Gcca \text{ (glava i CCA rep)} \mid G \text{ (samo glava)}$$

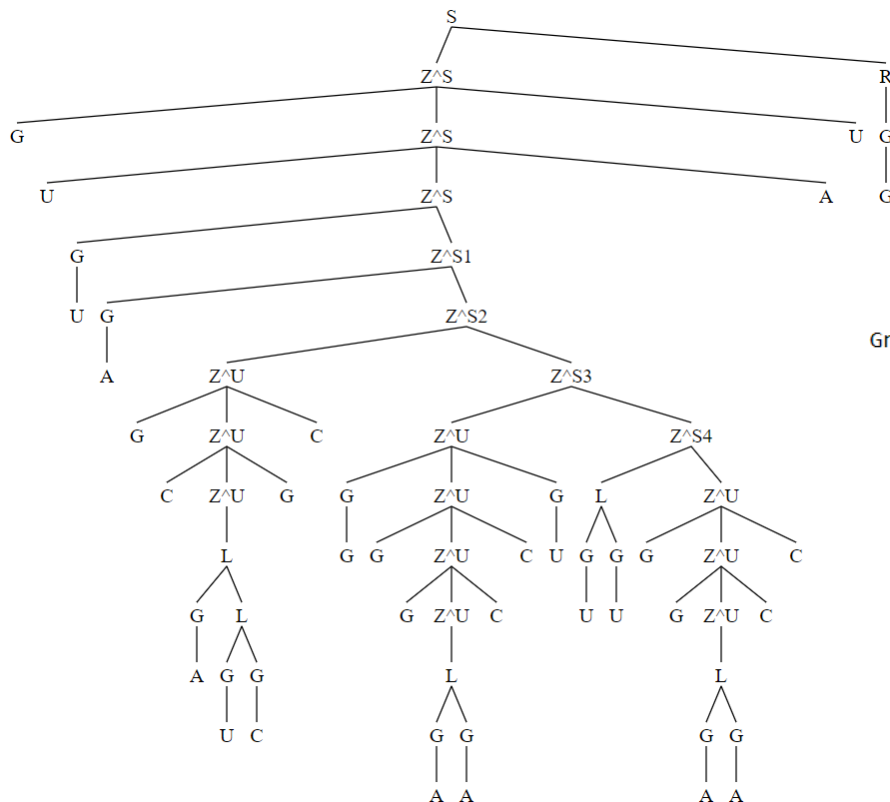
$$S \rightarrow GZ^S R \text{ (glava, zavojnica i rep)} \mid Z^S R \text{ (zavojnica i rep)}$$

$$Z^S \rightarrow dZ^S d \text{ (uparivanje)} \mid GZ^S G \text{ (promašaj)} \mid GGZ^U Z^U LZ^U \text{ (unutrašnjost)}$$

$$Z^U \rightarrow dZ^U d \text{ (uparivanje)} \mid GZ^U G \text{ (promašaj)} \mid L \text{ (unutrašnjost)}$$

$$L \rightarrow GL \text{ (nizanje baza)} \mid GG \text{ (dve baze)}$$

Opis implementacije



- ovakva gramatika se, očekivano, ponaša najbolje, budući da poznaje najviše konteksta, ali je i najsloženija i ograničena strogo na rad sa tRNK

Grammar with 30 productions

Z^U -> 'G' Z^U 'C' [0.315679]	Z^S -> 'U' Z^S 'A' [0.110768]
Z^U -> 'C' Z^U 'G' [0.220624]	Z^S -> 'G' Z^S 'U' [0.0301042]
Z^U -> 'U' Z^U 'A' [0.108909]	Z^S -> 'U' Z^S 'G' [0.0208414]
Z^U -> 'A' Z^U 'U' [0.0864609]	Z^S -> G Z^S G [0.00887688]
Z^U -> G Z^U G [0.0462995]	Z^S -> G Z^S 1 [0.125048]
Z^U -> 'G' Z^U 'U' [0.0349]	S -> G Z^S R [0.037037]
Z^U -> 'U' Z^U 'G' [0.0166608]	S -> Z^S R [0.962963]
Z^U -> L [0.170467]	R -> G 'C' 'C' 'A' [0.978395]
Z^S4 -> L Z^U [1.0]	R -> G [0.0216049]
Z^S3 -> Z^U Z^S4 [1.0]	L -> G L [0.838424]
Z^S2 -> Z^U Z^S3 [1.0]	L -> G G [0.161576]
Z^S1 -> G Z^S2 [1.0]	G -> 'U' [0.305379]
Z^S -> 'G' Z^S 'C' [0.353146]	G -> 'A' [0.280368]
Z^S -> 'C' Z^S 'G' [0.229255]	G -> 'G' [0.249011]
Z^S -> 'A' Z^S 'U' [0.121961]	G -> 'C' [0.165241]

Opis implementacije

- rezultati su upoređeni kako vizuelno, tako i upotrebom numeričkih **mera uspešnosti** karakterističnih za istraživanje podataka: udeo tačno predviđenih oznaka i sasvim tačnih struktura, odziv, preciznost
- druge dve mere dobijene su tako što je problem predviđanja sekundarne strukture shvaćen kao problem **pretraživanja informacija**, pri čemu se informacijom (dokumentom) smatra podatak da su dve baze uparene

$$\text{recall} = \frac{|\{\text{relevant documents}\} \cap \{\text{retrieved documents}\}|}{|\{\text{relevant documents}\}|}$$

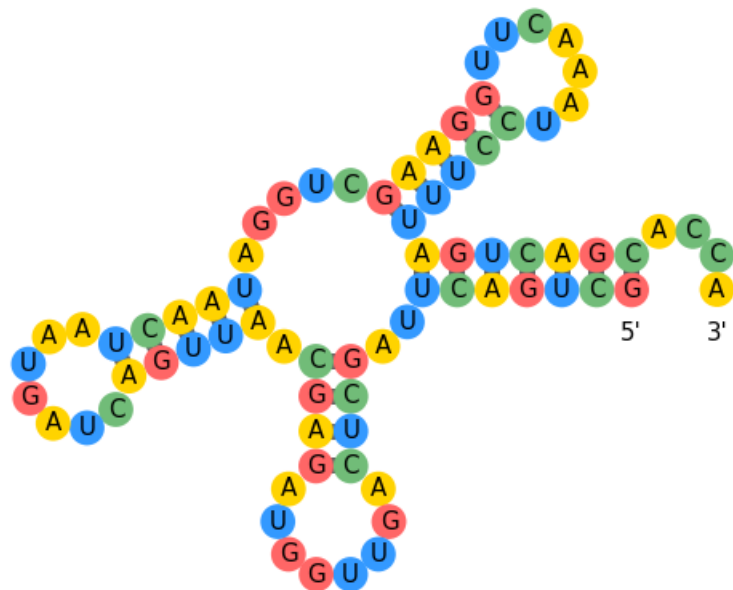
$$\text{precision} = \frac{|\{\text{relevant documents}\} \cap \{\text{retrieved documents}\}|}{|\{\text{retrieved documents}\}|}$$

Opis implementacije

>tdbR00000433|Mycoplasma_capricolum|2095|Thr|AGU
GCUGACUUAGCUCAGUUGGUAGAGCAAUUGACUAGUAAUCAUAGGUCGAAGGUUCAAUCCUUUAGUCAGCACCA

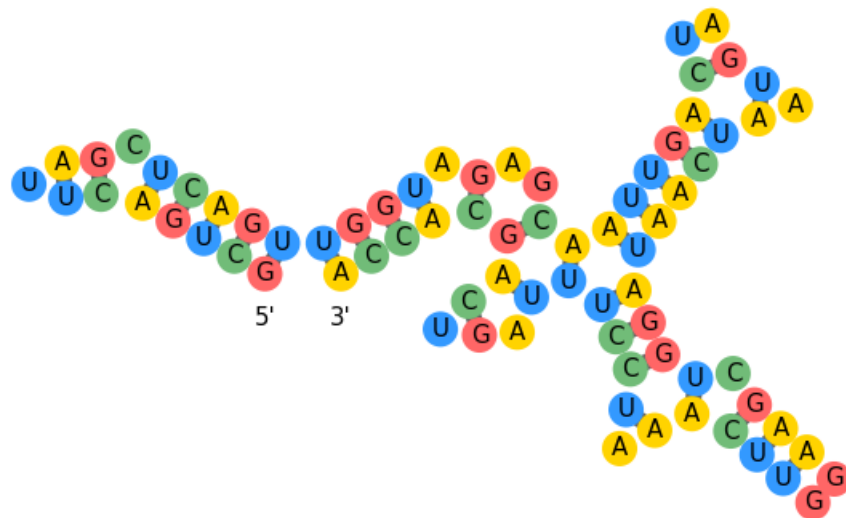
Stvarna struktura

(((((((.....))))).((((.....))))).((((.....)))))).....



Nussinov struktura

[jednakost: 46%] [odziv: 24%] [preciznost: 16%]
(((((((.....))))).((((.....))))).((((.....)))))).....



Opis implementacije

>tdbR00000433|Mycoplasma_capricolum|2095|Thr|AGU
GCUGACUUAGCUCAGUUGGUAGAGCAAUUGACUAGUAAUCAAUAGGUCGAAGGUUCAAUCCUUUAGUCAGCACCA

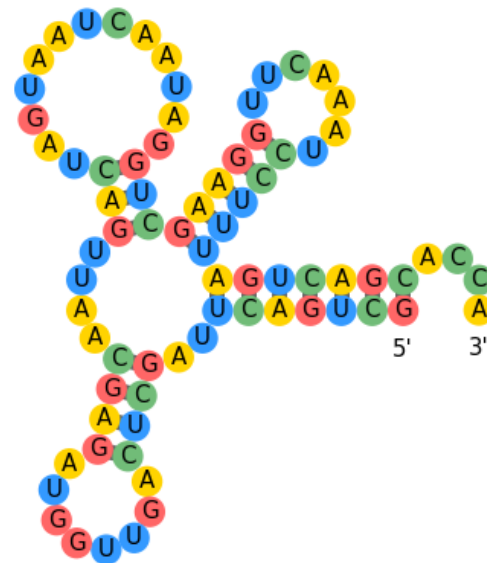
Stvarna struktura

(((((((.....))))).((((.....)))).....((((.....)))))).....



KH-99 struktura

[jednakost: 84%] [odziv: 76%] [preciznost: 84%]
(((((((.....))))).((((.....)))).....((((.....)))))).....

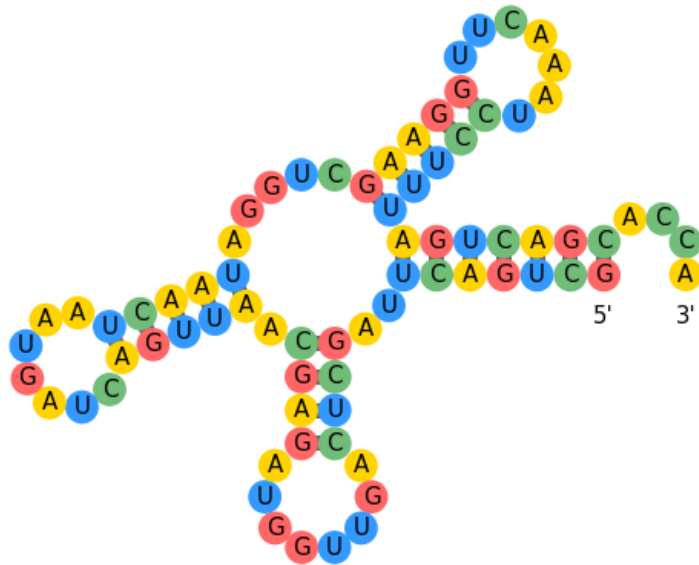


Opis implementacije

>tdbR00000433|Mycoplasma_capricolum|2095|Thr|AGU
GCUGACUUAGCUCAGUUGGUAGAGCAAUUGACUAGUAAUCAUAGGUCGAAGGUUCAAUCCUUUAGUCAGCACCA

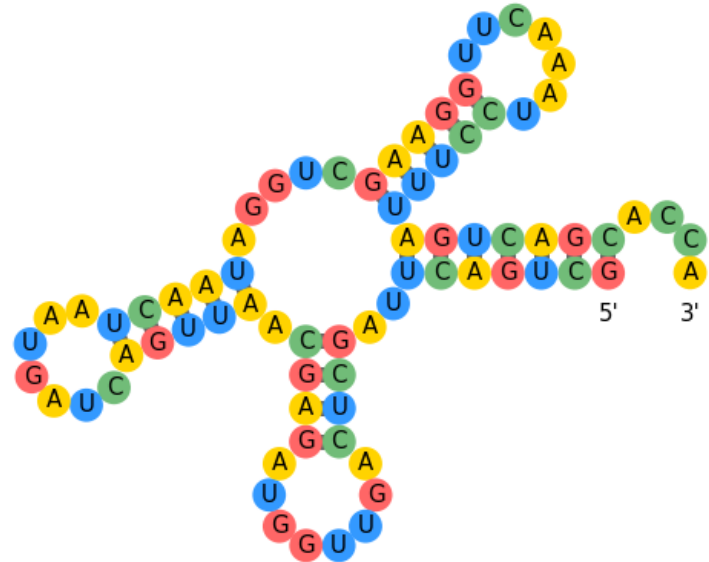
Stvarna struktura

(((((((.....))))).((((.....))))).((((.....)))))).....



CM struktura

[jednakost: 100%] [odziv: 100%] [preciznost: 100%]
(((((((.....))))).((((.....))))).((((.....)))))).....









Opis implementacije

- u tabeli su predstavljeni modeli i njihova uspešnost na skupu za proveru
- primera radi, može se primetiti da je model kovarijacije ubedljivo najuspešniji: sasvim tačno predviđa više od pola (54%) struktura, dok su mu ostale mere blizu maksimuma (>90%)
- uočljivo je i da znatno brže raste broj parametara od udela pogodaka, pa verovatno ne bi bilo preterano efikasno dalje usložnjavati modele

Gramatika		Udeo tačnih		Uparivanja	
Tip	Parametri	Oznake	Strukture	Odziv	Preciznost
Nussinov	11	48%	0%	27%	18%
KH-99	19	89%	38%	86%	86%
CM	26	96%	54%	93%	93%

Literatura

-  Daphne Koller, Nir Friedman (2009) *Probabilistic Graphical Models: Principles and Techniques – Adaptive Computation and Machine Learning*. The MIT Press.
-  R. Durbin, S. Eddy, A. Krogh, G. Mitchison (1998) *Biological Sequence Analysis: Probabilistic Models of Proteins and Nucleic Acids*. Cambridge University Press.
-  Zhao Q, Zhao Z, Fan X, Yuan Z, Mao Q, et al. (2021) *Review of machine learning methods for RNA secondary structure prediction*. PLOS Computational Biology 17(8).
-  Lazar Vasović, Nevena Ćirić (2022) *Sekundarna struktura tRNK*. GitHub repozitorijum: <https://github.com/matfija/Sekundarna-struktura-tRNK>
-  Lazar Vasović (2021) *Skriveni Markovljevi modeli u bioinformatici*. GitHub repozitorijum: <https://github.com/matfija/HMM-u-bioinformatici>
-  Mina Aleksandra Konaković (2014) *Stohastičke kontekst slobodne gramatike i primene*. eBiblioteka: <http://elibrary.matf.bg.ac.rs/handle/123456789/3857>