Prediction and identification of secondary structures

Nadia El-Mabrouk

University of Montreal,
Canada

Introduction

Challenge: Decode and interpret data of complete genomes.
Essential to develop algorithmic tools.

For linear genetic sequences $\longrightarrow$ BLAST, FASTA

RNA fundamental for the cell function. Function determined by secondary and tertiary structure

Problematics:

1. What is the structure of a given genetic element?
2. How to identify a gene characterized by its structure?

Plan

1. Introduction to RNAs
2. Secondary structure prediction
   - Energy minimization;
   - General dynamic algorithm;
   - Ignoring multiple loops;
3. Secondary structure identification
   - Approximate matching of context free grammar;
   - Approximate matching of secondary expression by pushdown automata

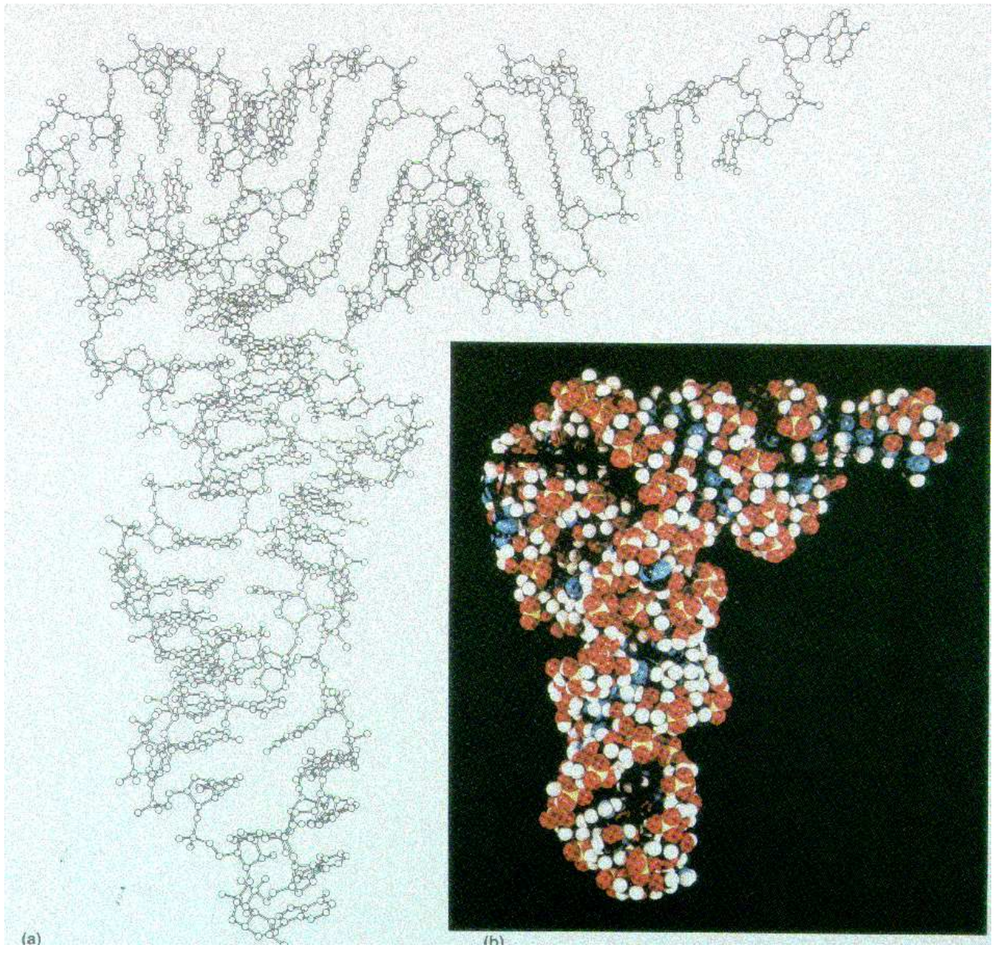
I.Introduction / RNAs

- Ribosomal RNAs (rRNA)
- Small nuclear RNAs (snRNA);
- Messenger RNAs (mRNA);
- Transfert RNAs (tRNA);
- RNA component of ribonuclease P (RNase P RNA);

Primary structure

A sequence of 4 ribonucleic acids: A,C,G,U. In DNA, U is T.

GCGUGGGUGAUCUAGUGGUUAUGAUGUCUGCUUUACACGCAGAACGUCGCGGGUUCGAA...



(a)

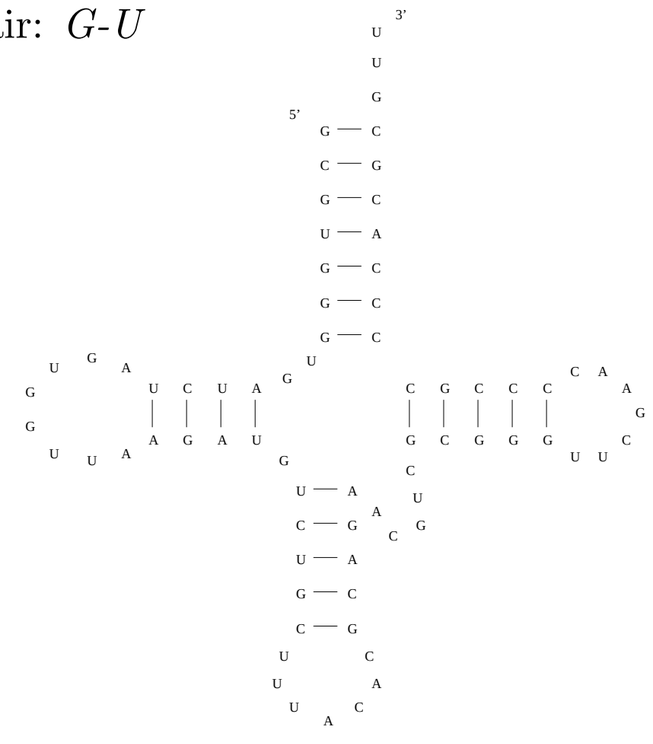
(b)

5

Secondary structure

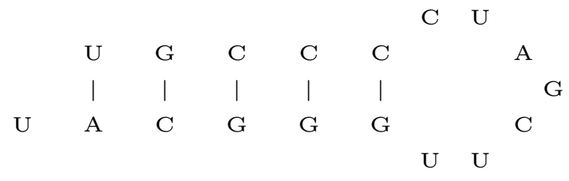
Watson-Crick base pairs: *G-C, A-U*

Wobble base pair: *G-U*

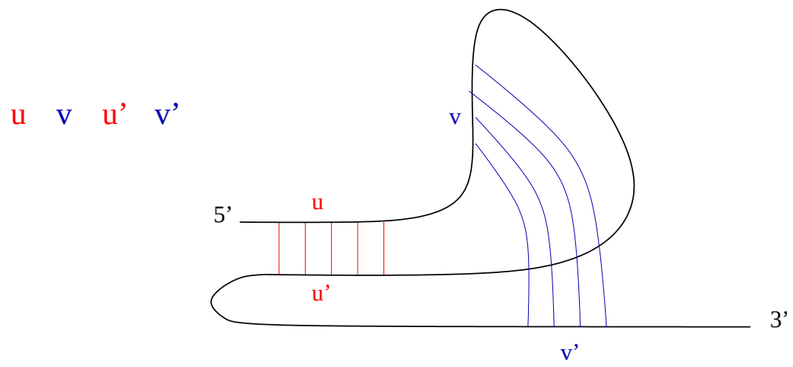


Secondary structure - Definition

1. Pairings: Watson-Crick, Wobble
2. No overlap of pairs: Each base paired with at most one base
3. No sharp turns: at least 3 bases in a loop



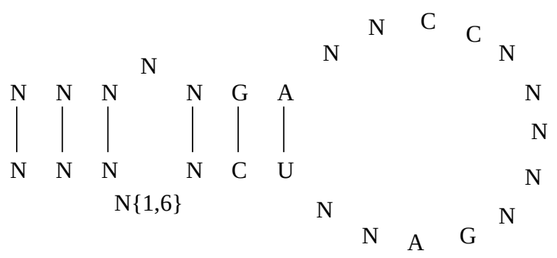
4. No pseudoknots



RNA sequence evolution constrained by structure

Homologous RNAs have common secondary structure BUT no significant sequence similarity

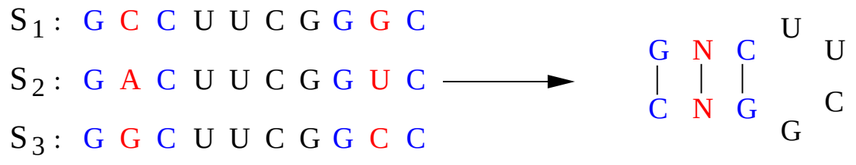
Region III of mitochondrial 5S RNAs:



II. RNA structure prediction

One sequence $\longrightarrow$ Many possible foldings

- Multiple alignments: Identify covarying residues (*Woese & Pace, 1993*)



- Stable structure = Structure minimizing the free energy

Energy minimization

Prevalent method: Dymanic programming

Sankoff 1976; Nussinov 1978; Waterman 1978; Zuker & Steigler 1981; Zuker & Sankoff 1984; Sankoff 1985

$O(N^3)$ time; $O(N^2)$ space

ViennaRNA software, *Schuster et. al., 1994*

MFOLD software, *Zuker & al., 1989*

Thermodynamic considerations

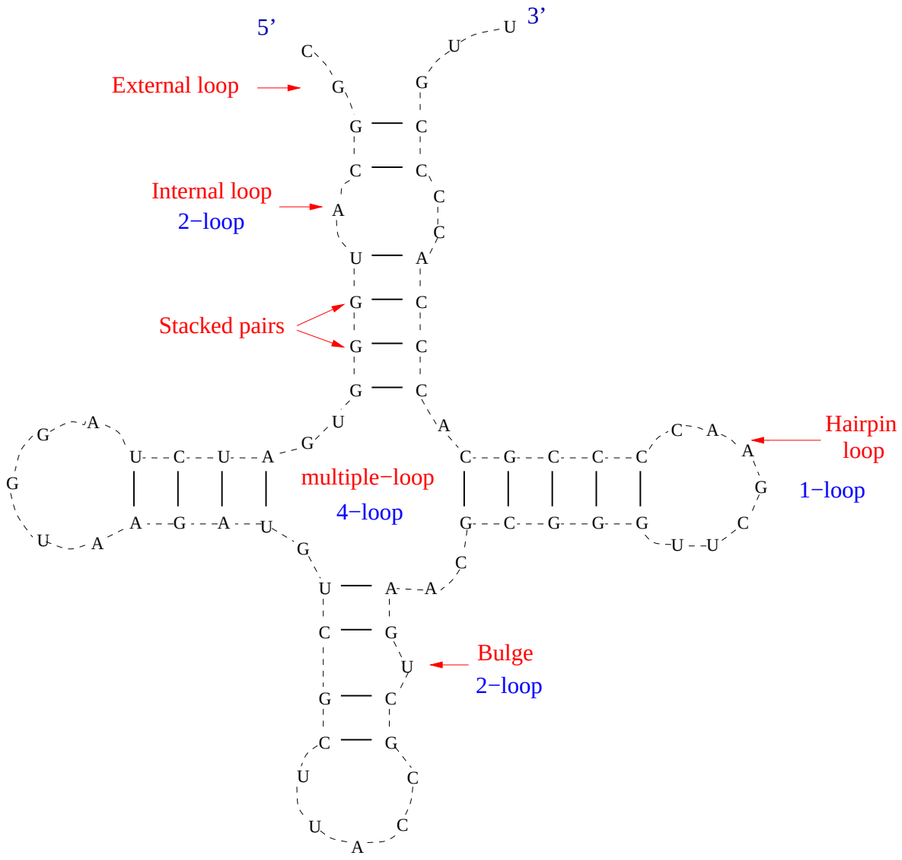
A structure S, loops and stacked pairs $s_1, s_2, \cdots s_t$.

Tinoco-Uhlenbeck hypothesis:

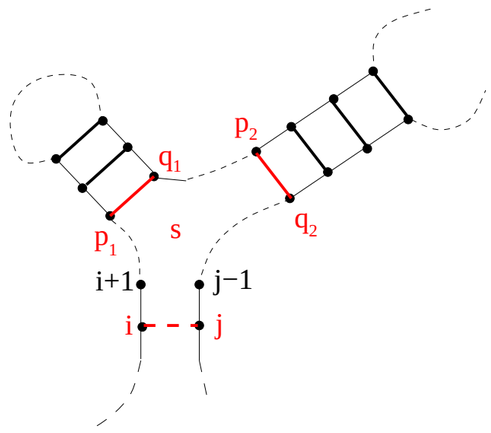
$$E(S) = e(s_1) + e(s_2) + \cdots e(s_t)$$

$e(s_i)$ experimentaly estimated. Depends on the cycle type.

Loops


5’
3’
External loop
Internal loop
2−loop
Stacked pairs
multiple−loop
4−loop
Hairpin
loop
1−loop
Bulge
2−loop

General dynamic programming algorithm



s: multiple loop from i to j.

$$E(S_{ij}) = \begin{cases} \sum_{h=1}^{k-1} E(S_{p_h,q_h}) & \text{if } [i,j] \text{ is not closed} \\ e(s) + \sum_{h=1}^{k-1} E(S_{p_h,q_h}) & \text{if } [i,j] \text{ is closed} \end{cases}$$

Key observation: S is optimal if any sub-structure of S is optimal.

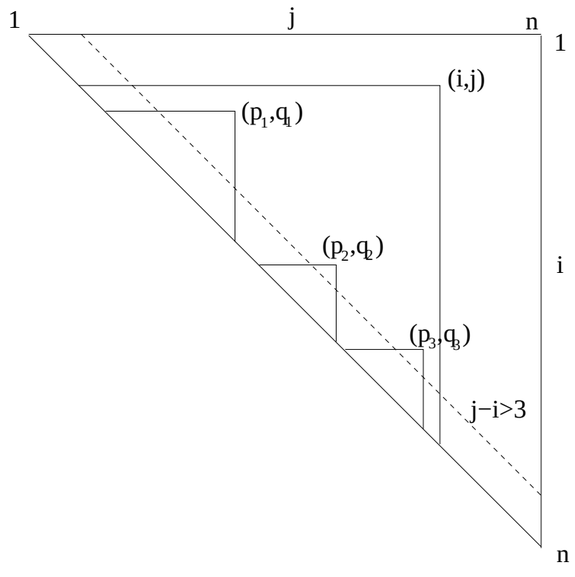
$E(i,j)$: Optimal folding of $[i,j]$. Find $E(1,n)$

$$E(i,j) = \min \begin{cases} 0 & (j-i<4) \\ C(i,j) & (\text{optimal structure with } (i,j) \text{ closed}) \\ \min_{i\le h\le j}[E(i,h)+E(h+1,j)] \end{cases}$$

$$C(i,j) = \min_{k\ge 1} \min_{\substack{s:k-cycle \\ from\ i\ to\ j}} \left[e(s) + \sum_{(p_h,q_h)} C(p_h,q_h) \right]$$

Basic algorithm

Square array, each cell (i, j) containing $F(i, j)$ and $C(i, j)$



Calculate each diagonal; Begin by the main diagonal

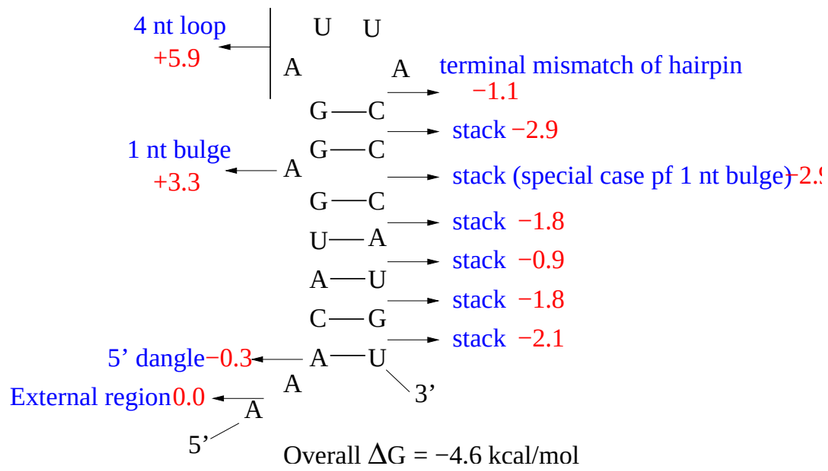
Keep pointers to retrace optimal structures

Complexity: Each cell $(j-i)^{2k} \implies n^{2k}$

Ignoring multiple loops

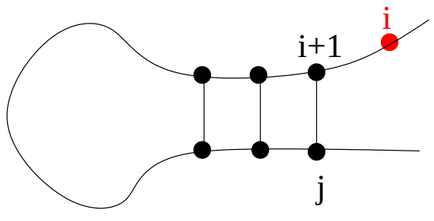
s multiple loop $\longrightarrow e(s) = 0$

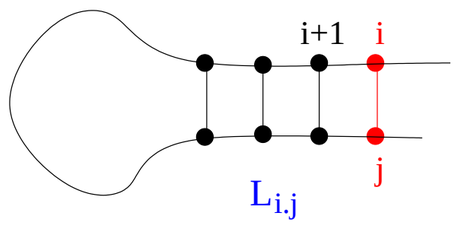
Experimental evaluation of $e(s)$ for simple loops and stacked pairs

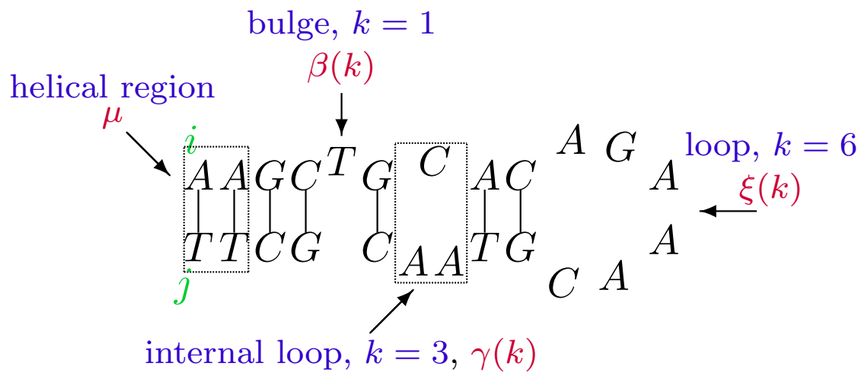


Freir rules (1986) at 37^{o}C. Example from *Durbin, Eddy et. al.* book.

$$E(S_{i,j}) = \min \begin{cases} E(S_{i+1,j}) \\ E(S_{i,j-1}) \\ \min_{i<k<j}\{E(S_{i,k}) + E(S_{k+1},j)\} \\ E(L_{i,j}) \end{cases}$$







$E(S_{i,j})$:

$$\alpha(i,j) + \xi(j-i-1) \quad \text{loop}$$

$$\alpha(i,j) + \mu + E(S_{i+1,j-1}) \quad \text{helical region}$$

$$\min_{k\geq 1}\{\alpha(i,j) + \beta(k) + E(S_{i+k+1,j-1})\} \quad \text{bulge at } i$$

$$\min_{k\geq 1}\{\alpha(i,j) + \beta(k) + E(S_{i+1,j-k-1})\} \quad \text{bulge at } j$$

$$\min_{k_1,k_2\geq 1}\{\alpha(i,j) + \gamma(k_1+k_2) + E(S_{i+1+k_1,j-1-k_2})\} \quad \text{internal loop}$$

$$\min_{i+1<k<j-2}\{E(i+1,k) + E(k+1,j-1)\} \quad \text{multiple loop}$$

Basic algorithm

Square array, each cell (i, j) containing $E(S_{i,j})$

	j	$j-1$	$j-2$	$\cdots$
i	α			
$i+1$		μ	β	
$i+2$		β	γ	
.				
.				

Complexity:

- Loop or helical region: Constant time for each $(i, j) \Longrightarrow O(n^2)$
- Bulge: $O(n)$ for each $(i, j) \Longrightarrow O(n^3)$
- Internal loop: $O(n^2)$ for each $(i, j) \Longrightarrow O(n^4)$

III. Secondary structure identification

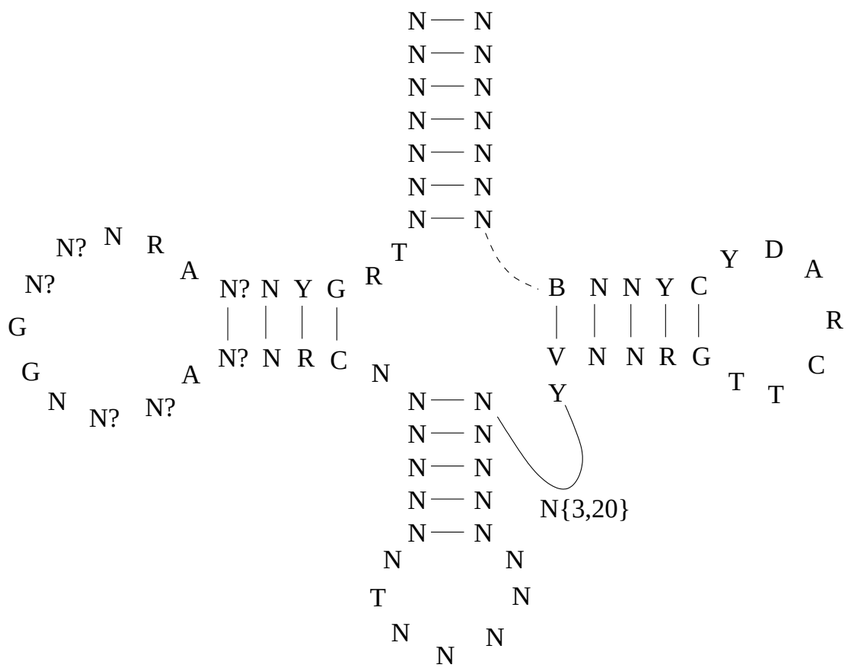
Input: A characterization of the secondary structure of an RNA family.

Goal: Identify such RNAs in a genome.

Use weights, scores or number of errors.

Characterization of an RNA structure

A possible characterization of a tRNA:



Existing methods

Tailor-made:

- FAStRNA for tRNAs (*N. El-Mabrouk and F. Lisacek, 1996*),
- CITRON for group I introns (*F. Lisacek and Y. Diaz and F. Michel, 1994*),
- SNOSCAN for snoRNAs (*T. Lowe and S. Eddy, 1999*), ...

General:

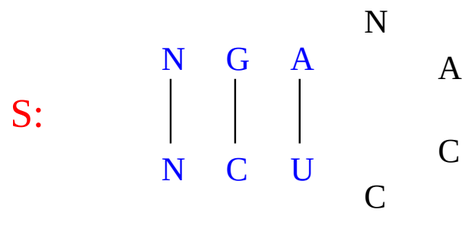
- RNAMOTIF (*T. Macke et al., 2001*),
- SCFG-based RNA models (*S. Eddy and R. Durbin, 1994*, *Sakakibara et al. 1994*)
- BioSmatch (*N. El-Mabrouk and M. Raffinot, 2003*)

Approximate matching problem

A genome T

A secondary structure S

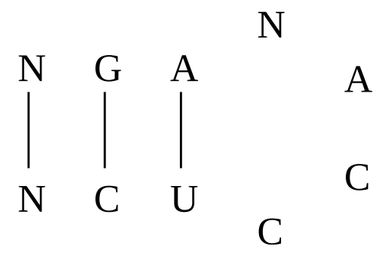
$\longrightarrow$ all positions in T corresponding to occurrences of S, with at most k errors, or with a minimum score



T: C A C C U C A G G A A U C U C G C T G G G A T A C

N G A N A C C U C N

I. Approximate matching of a context-free grammar



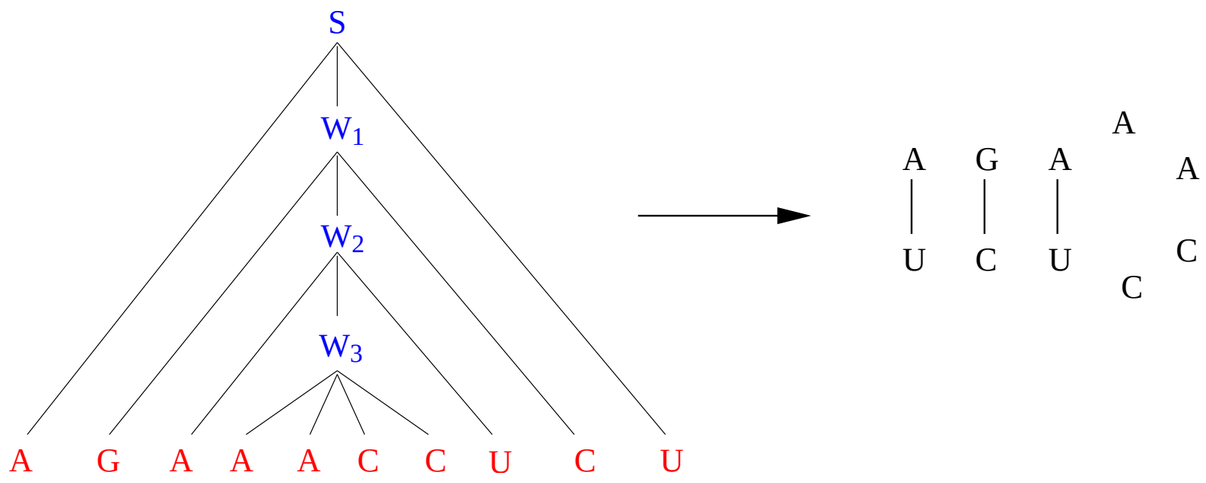
$$
\begin{aligned}
S &\rightarrow A\,W_1\,U \mid C\,W_1\,G \mid G\,W_1\,C \mid U\,W_1\,A \\
W_1 &\rightarrow G\,W_2\,C \\
W_2 &\rightarrow A\,W_3\,U \\
W_3 &\rightarrow AACC \mid CACC \mid GACC \mid UACC
\end{aligned}
$$

S, W_1, W_2, W_3: non-terminals

A, C, G, T: terminals

Parse Tree

$$
\begin{aligned}
S &\rightarrow A\,W_1\,U \mid C\,W_1\,G \mid G\,W_1\,C \mid U\,W_1\,A \\
W_1 &\rightarrow G\,W_2\,C \\
W_2 &\rightarrow A\,W_3\,U \\
W_3 &\rightarrow AACC \mid CACC \mid GACC \mid UACC
\end{aligned}
$$



Approximate matching of context-free language

Restriction of *G. Myers 1995* to acyclic grammars of form:

$$X \to a\,Y\,\overline{a}\,, \quad X \to a\,Y\,, \quad X \to Y\,a\,, \quad X \to a$$

Goal: Given a DNA sequence, find an alignment of the grammar with maximum score.

- $\delta(a, b)$: Score of aligning a with b
- $\delta(a)$: Score of deleting or inserting a
- $\rho(a)$: Score of a correct pairing $(a \bullet \overline{a})$

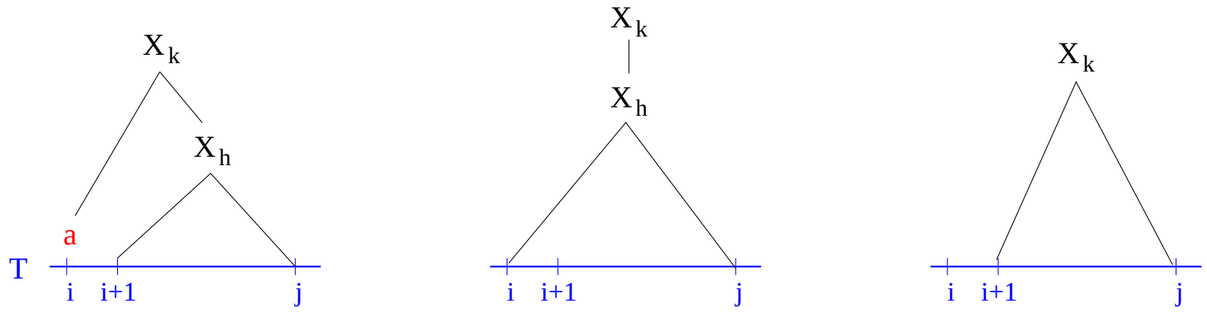
Dynamic programming algorithm

A matrix M_k $n \times n$ for each non-terminal X_k

$(M_k)_{i,j}$: Max score of an alignment of X_k with $[i, j]$. Find $(M_S)_{1,n}$.

Example: $X_k \rightarrow a\, X_h$

$(M_k)_{i,j}$: Max over all X_h of three values



a aligned with T[i] | Deletion of a | Deletion of T[i]

Complexity: sn^2, s number of non-terminals.

II. Secondary expression

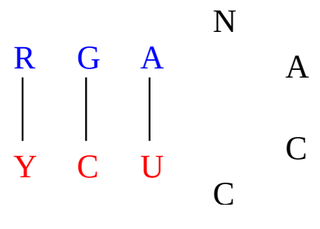
N. El-Mabrouk and M. Raffinot, 2003

Network expression

1. Any character (A, C, G, T),
2. $E_1 \mid E_2$ and $E_1 E_2$

$$(A \mid C)\,(\,A|\,G) \longrightarrow \{AA,\ AG,\ CA,\ CG\}$$

Complement (of E): $\overline{A} = T$, $\overline{T} = A$, $\overline{C} = G$ and $\overline{G} = C$,

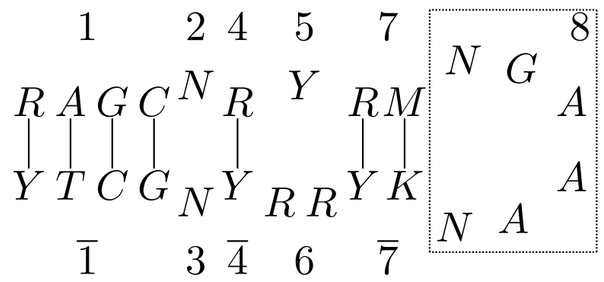


$$(A \mid G)\,G\,A \longrightarrow U\,C\,(T \mid C)$$

Secondary expression (of E)

1. E network expression $\rightarrow S = (E, p)$ is a secondary expression,
2. $E_1, E_2, E_3 \rightarrow$
   $S = (E_1, p)(E_2, sl)\ S'\ (\overline{E_2}, sr)(E_3, p)$ is a secondary expression.

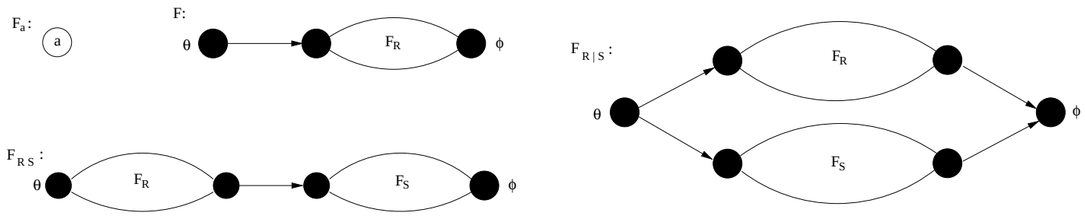
Group II intron's domain V:



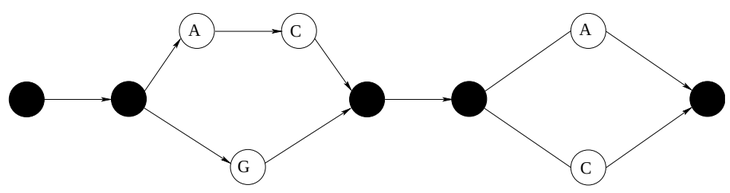
$S = (1, sl)(2, p)(4, sl)(5, p)(7, sl)(8, p)(\overline{7}, sr)(6, p)(\overline{4}, sr)(3, p)(\overline{1}, sr)$

Thompson construction (1968)

Black states: ε labeled states



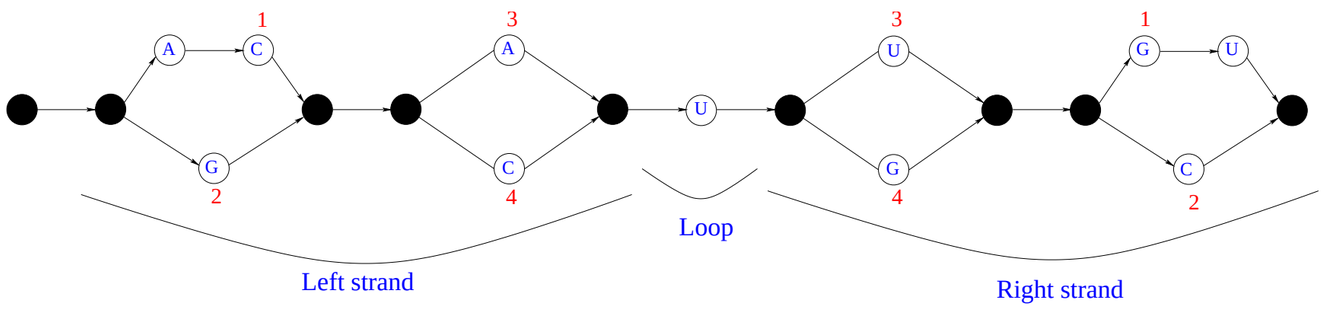
$$E_1 = ((A\,C)\mid G)\ (A\mid C) \longrightarrow \{ACA, ACC, GA, GC\}$$



Marking the states

$S = (E_1, sl)(E_2, p)(\overline{E_1}, sr)$, with $E_1 = ((AC)\,|G)\,(A|C)$, $E_2 = U$.

$E = E_1\ E_2\ \overline{E_1}$



Pushdown automaton 1

Non-deterministic, state-labeled pushdown automaton ε-NFPA:

$\mathcal{A} =< N, \Gamma, V, E, \lambda, \gamma, \theta, \phi, I >$,

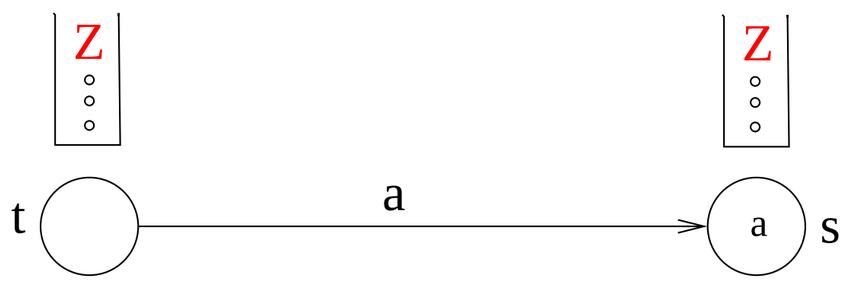
1. an input alphabet $N = \{A, C, G, T\}$
2. a stack alphabet Γ: all possible marks
3. a set V of vertices called states
4. a set E of directed edges between states
5. a fonction λ assigning a label in $N \cup \{\varepsilon\}$ to each state
6. a transition function γ

Our Pushdown automaton transition rules (1)

Notation:

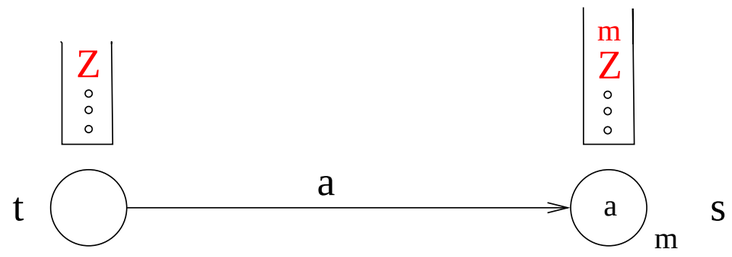
a should be equal to $\lambda(s)$.

Tr_1 If s is a p-state or an unmarked state, $\gamma(t, \lambda(s), Z) = (s, Z)$

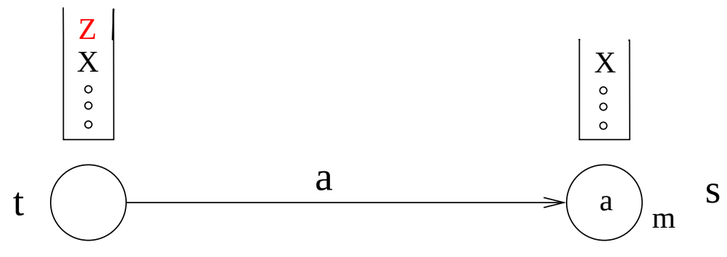


Our Pushdown automaton transition rules (2)

Tr_2 If s is a marked sl-state, $\gamma(t, \lambda(s), Z) = (s, mZ)$

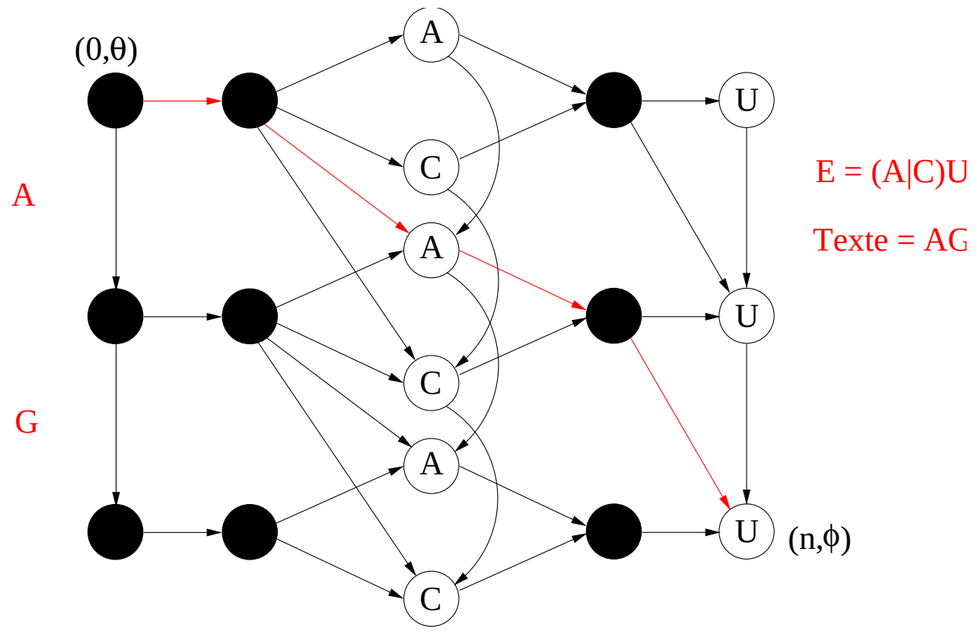


Tr_3 If s is an sr-state such that $m = Z$, $\gamma(t, \lambda(s), Z) = (s, \varepsilon)$



Alignment graph 1

Alignment of a network exp. E with a text T. Weighted directed graph: $n+1$ copies of the automaton recognizing E.



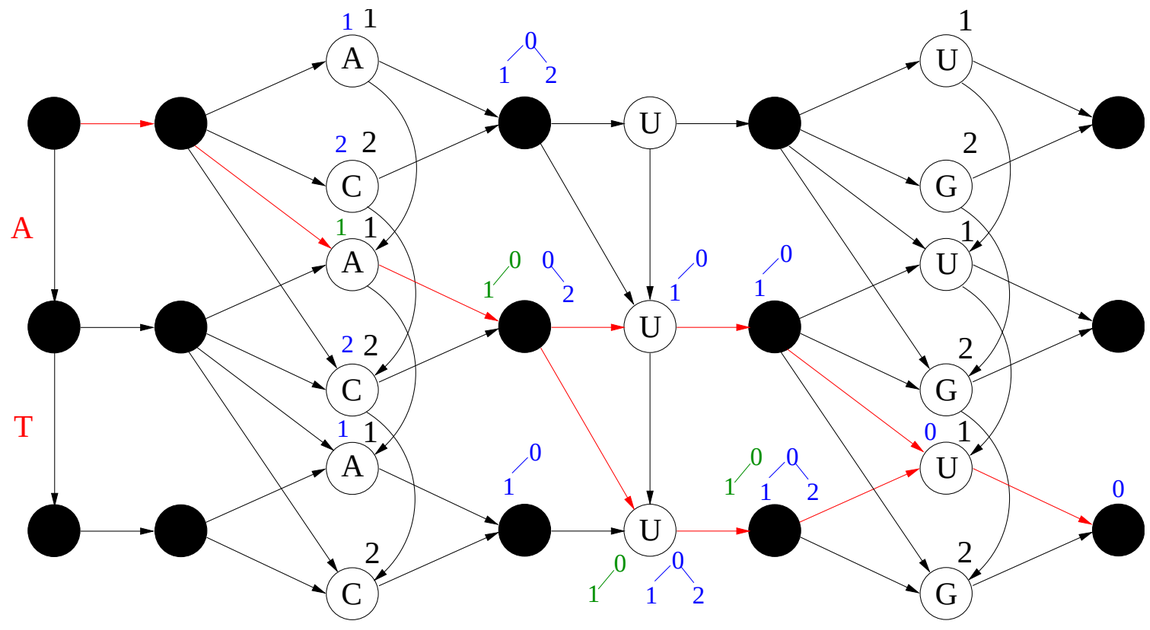
Best alignment of E with $T \rightarrow$ least cost path

Alignment graph 2

$S = \{(A|C), sl\}\{T, p\}\{(T|G), sr\}$, reading AT.

At most $k = 1$ error.

Blue stacks for $k = 1$; Green stacks for $k = 0$.

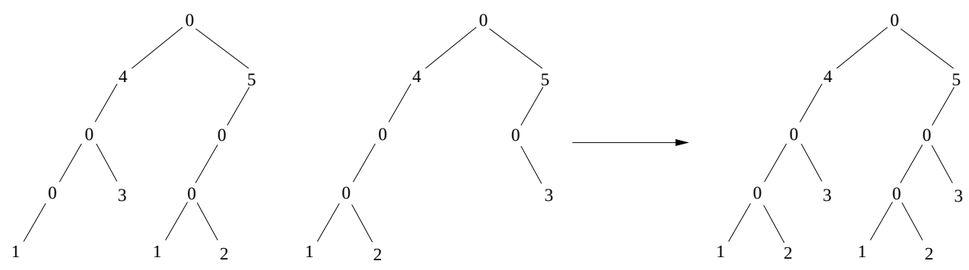


Representing the stacks

A binary tree for each error level

Node P: integer $P.val$, ($P.left$, $P.right$).

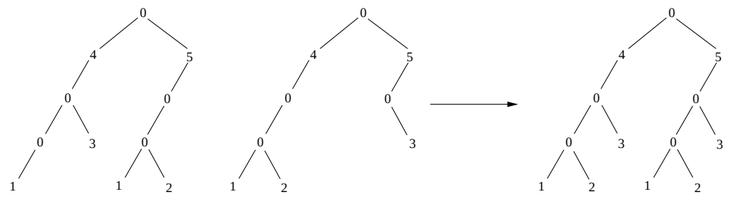
- INSERT(P, num), new node;
- REMOVE(P) removes the top element.
- COMBINE(P_1, P_2), new node
  $P.val = 0$, $P.left = P_1$ and $P.right = P_2$.
- MERGE(P_1, P_2) recursively merges the two trees.



Complexity

n: Text, p: Secondary expression E, r: Number of OR "|" in E.

INSERT, REMOVE, COMBINE are $O(1)$, and MERGE is $O(r)$ in the worst case



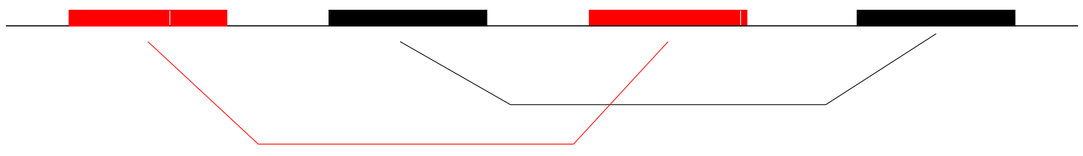
$\longrightarrow$ Upgrading one tree is $O(r)$.

For each node, at most $k+1$ trees $\longrightarrow$ $O(kr)$.

$O(pn)$ nodes in the alignment graph

Final complexity: $O(krpn)$

Extension 1: Pseudo-knots

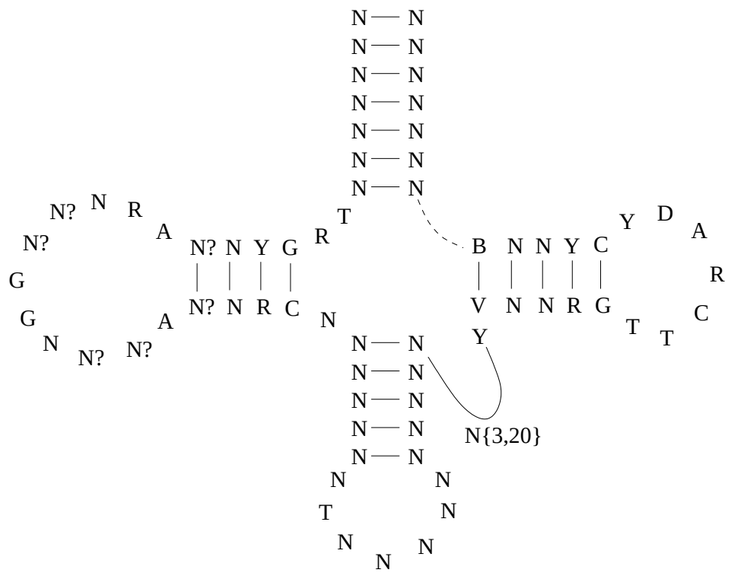


Definition: *A pseudo-knot is an expression of the form* $S_l^1\ S_l^2\ S_r^1\ S_r^2$, *where* S^1 *and* S^2 *are two secondary expressions.*

Two blocks of stacks: one for $S_l^1\ S_r^1$, and one for ,$S_l^2\ S_r^2$

At a node (i, s), if s belongs to $S_l^1\ S_r^1$, update the block of $S_l^1\ S_r^1$ only; otherwise update the block of $S_l^2\ S_r^2$ only

Extension 2: Generalized secondary expressions



Each helix has its own left and right strand in the automaton representing a GSE.

Conclusion

Approximate matching of secondary expressions:

- Worst-time complexity: $O(krpn)$
- Extendable to pseudo-knots and multi-loops structures.

Approximate matching of a context free grammar:

- Worst-time complexity $O(sln)$
- Not extendable to pseudo-knots

Practical problems:

- Stack management is practically time-costly
- Big variable loop