Algorithms in Bioinformatics: A Practical Introduction

RNA Secondary Structure Prediction

Functions of RNA

- Serves as the intermediary for transforming DNA to protein
- Functions as a catalyze
- Acts as information storage in viruses such as HIV

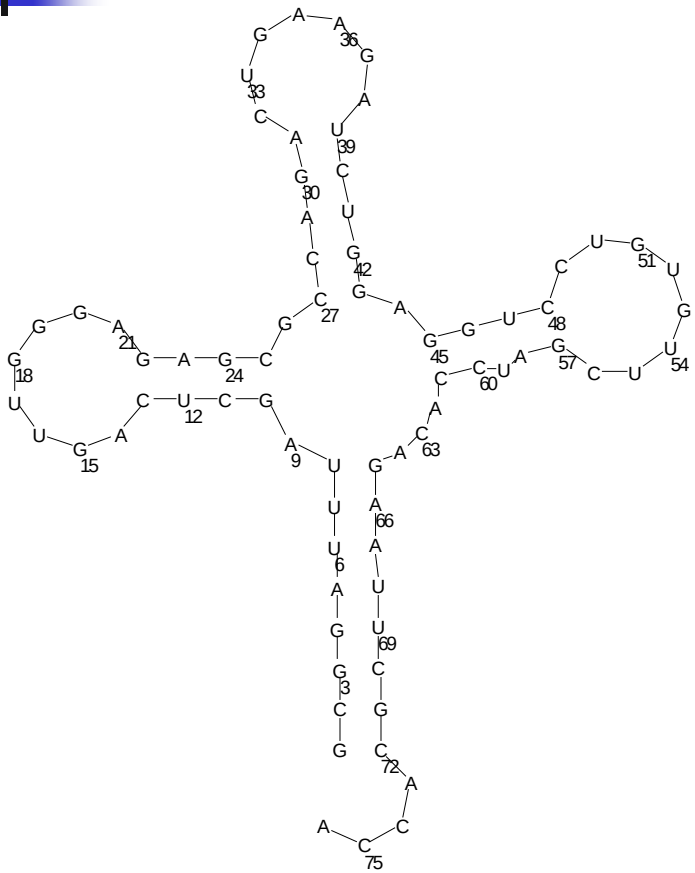
Why we study the structure of RNA?

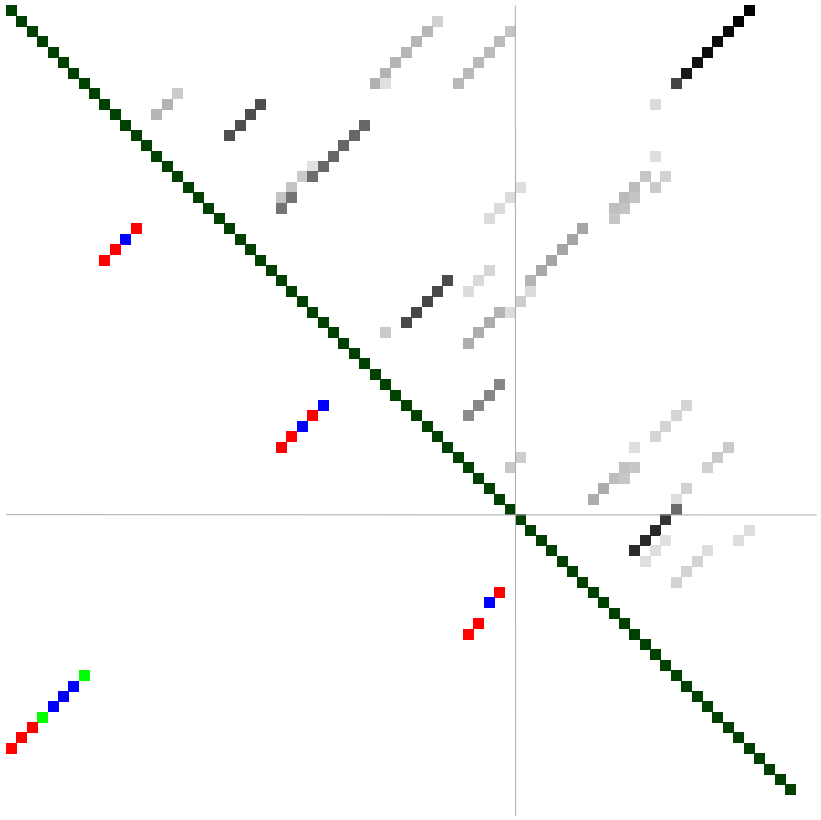
- RNA is the only known molecular which can act as information storage and as catalyze
- It seems that their functionality is quite related to their structure

RNA structure

- As RNA has an extra OH attaching to 2’ carbon, RNA forms extra hydrogen bond which enable it to have 3D structure
- RNA structure can be described in three levels
  - Primary structure
    - Just the sequence
  - Secondary structure
    - The base pairs
  - Tertiary structure
    - The 3-dimensional structure

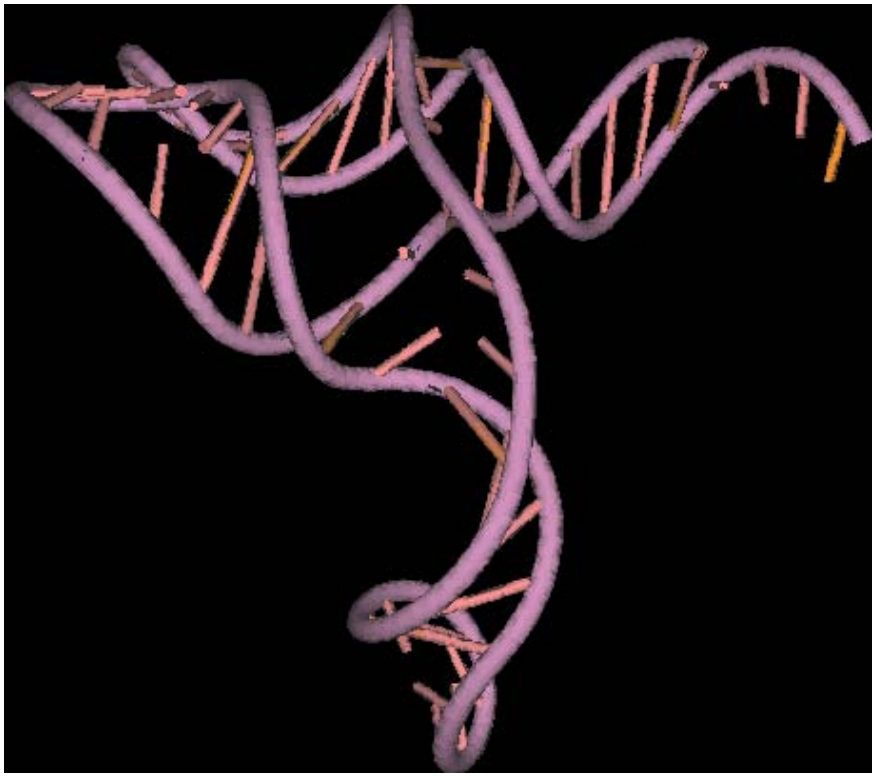
Example (Secondary structure for phenylalanyl-tRNA)

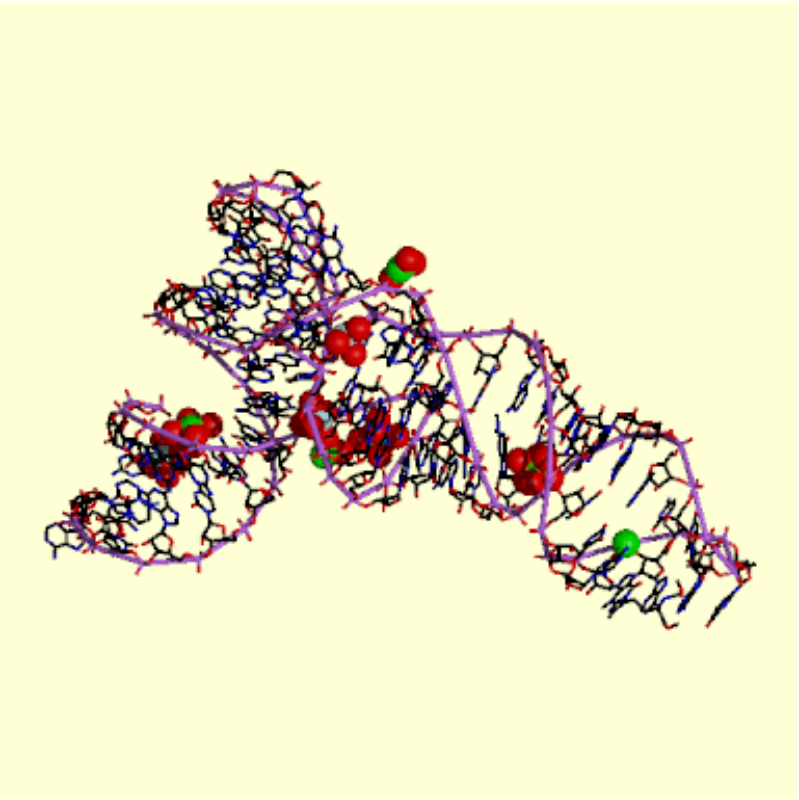




GCGGAUUUAGCUCAGUUGGGAGAGCGCCAGACUGAAGAUCUGGAGGUCC
UGUGUUCGAUCCACAGAAUUCGCACCA

Example (Tertiary structure for phenylalanyl-tRNA)





http://www.geocities.com/CollegePark/Hall/3826/interests.html
http://www.biochem.ucl.ac.uk/bsm/pdbsum/1ehz/tracel_r.html

Example (Function for phenylalanyl-tRNA)

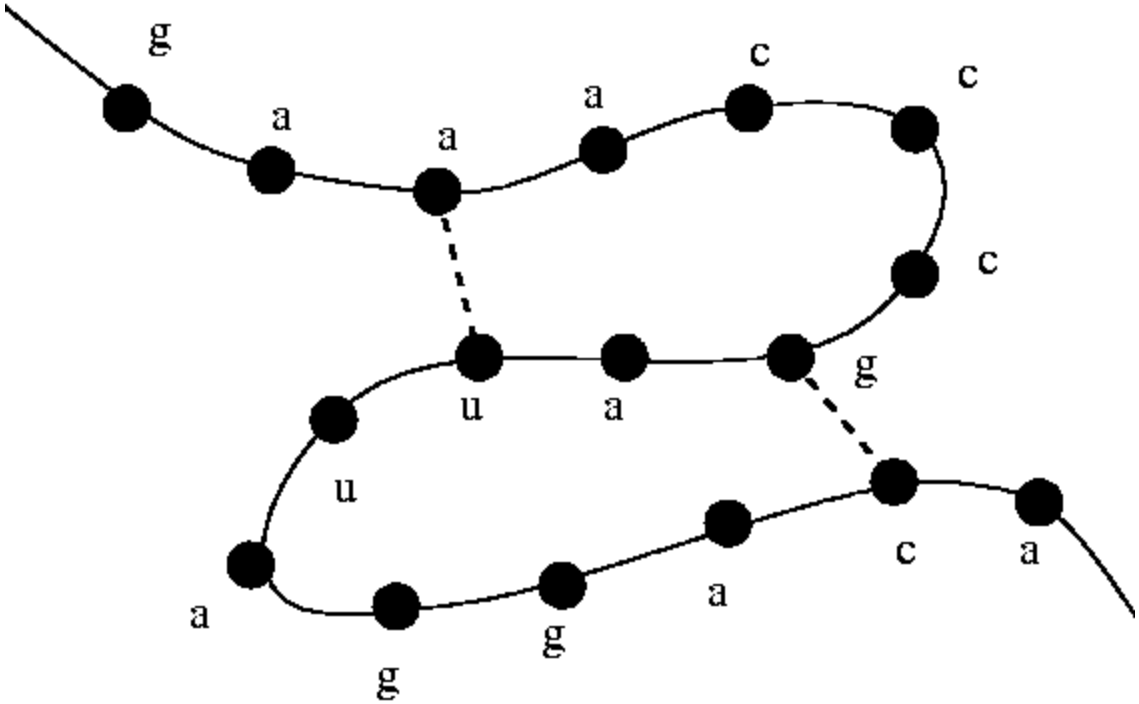
- The structure of phenylalanyl-tRNA is a cloverleaf.
- Its function is to translate a codon (3 bases) into an amino acid.
- Note that the cloverleaf structure is essential to its translation function.

Formal definition of RNA secondary structure

- Given a RNA $s=s_1s_2\ldots s_n$ where $s_i \in \{a, c, g, u\}$.
- For $1 \leq i < j \leq n$, if s_i and s_j form a base pair via hydrogen bond, we say (i, j)
  - Normally, a base pair is c-g or a-u.
  - Occasionally, we have g-u pair.
- A secondary structure of a RNA s is a set S of base pairs such that each base is paired at most once.

Pseudoknot

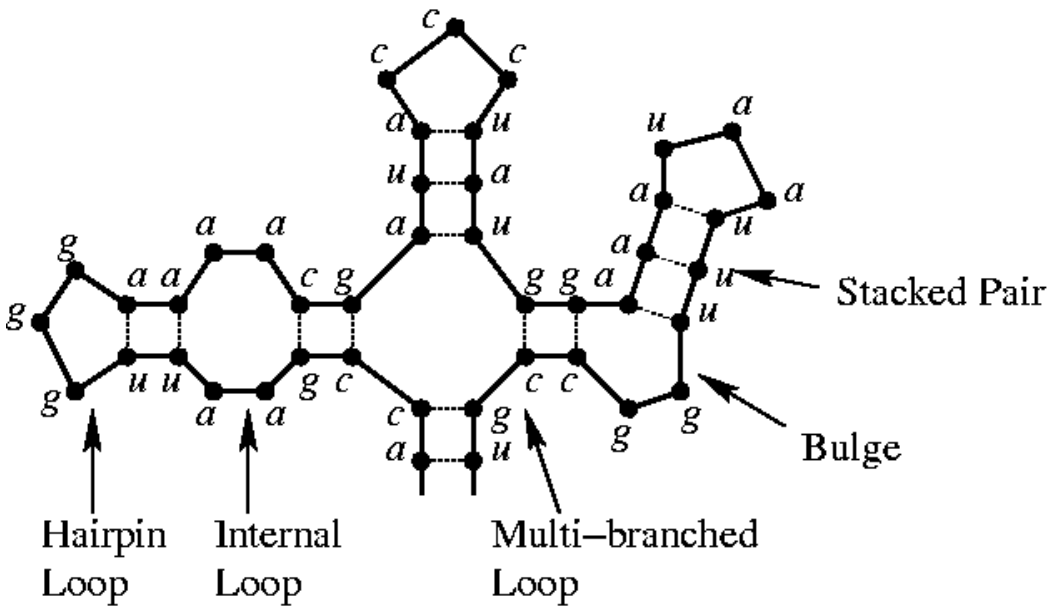
- A pseudoknot is two base pairs (i,j) and (i’,j’) such that i<i’<j<j’



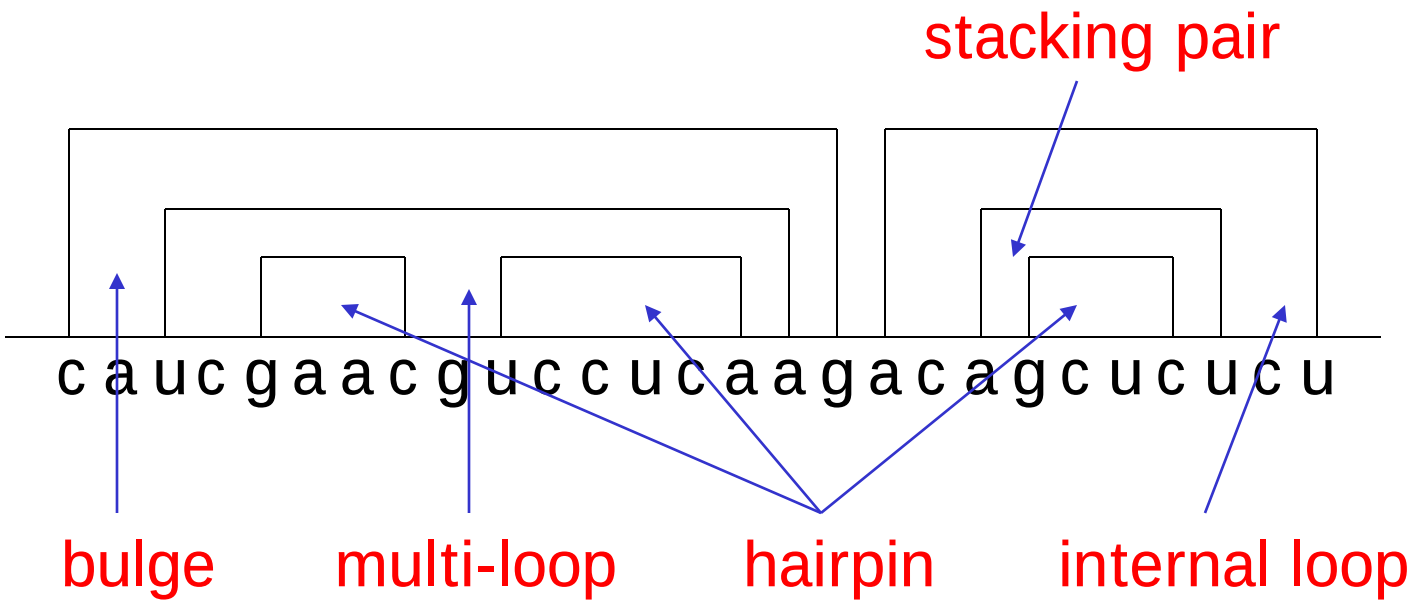
Loops

- Suppose there is no pseudoknot!
- Loops are regions enclosed by backbone and base pairs.

- Hairpin: loop contains exactly one base pair
- stacked pair: loop formed by base pairs (i,j), (i+1,j-1)
- Internal loop: loop contains two base pairs
- Bulge: internal loop with two adjacent bases.
- Multi-loop: loop contains three or more base pairs



Another view of loops



How to obtain RNA secondary structure?

Different ways to obtain RNA secondary structure.

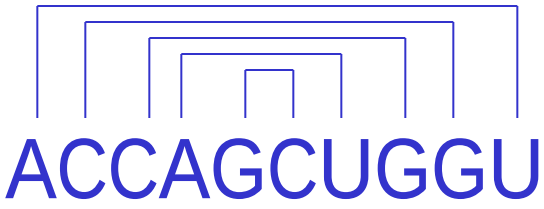
1. By experiment
   - X-ray Crystallography
   - NMR Spectroscopy
2. Phylogenetic approach
   - Given a sufficient number of related RNA sequences, infers the RNA structure
3. Prediction
   - For secondary structure, based on the current best solution, on average, we can correctly predict 73% of known base-pairs when sequence of fewer than 700 bases are folded

Overview

- In this lecture, we focus on RNA secondary structure prediction.
  - RNA secondary structure prediction problem (without pseudoknot)
    - Define thermodynamic model
    - Dynamic programming solution
    - Speedup
  - RNA secondary structure prediction problem (with pseudoknot)

RNA secondary structure prediction problem

- Nussinov folding algorithm
  - Idea: maximize the number of base pairs
- Example: ACCAGCUGGU



Nussinov folding algorithm (I)

- Let S[1..n] be the RNA sequence
- Let V(i,j) be the maximum number of base pairs in S[i..j].
- Base case:
  - V(i,i)=0 since the sequence has only one base!
  - V(i+1,i)= 0 since the sequence is empty!

Nussinov folding algorithm (II)

- When $i<j$, we have four cases:
  1. No base pair attached to j
     - $V(i, j) = V(i, j-1)$
  2. No base pair attached to i
     - $V(i,j) = V(i+1, j)$
  3. (i, j) form a base pair
     - $V(i, j) = V(i+1, j-1) + \delta(S[i], S[j])$
       where $\delta(x, y)=1$ if $(x,y)\in\{(a,u), (u,a), (c,g), (g,c), (g,u), (u,g)\}$; and 0, otherwise
  4. Both I and j attached to some base pairs both (i,j) is not a base pair
     - $V(i, j) = \max_{i\le k<j}\{V(i,k)+V(k+1,j)\}$

- Note: cases 1 and 2 are subcase of case 4!

Nussinov folding algorithm (III)

- Therefore, we have:
  - Base case:
    - V(i,i)=0, V(i+1,i)=0
  - Recursive case (i<j):
    - $$V(i,j) = \max \begin{cases} V(i+1, j-1) + \delta(S[i], S[j]) \\ \max_{i \le k < j} \{V(i,k) + V(k+1, j)\} \end{cases}$$

Example: base case

- S[1..7]=ACCAGCU

	1	2	3	4	5	6	7
1	0						
2	0	0					
3		0	0				
4			0	0			
5				0	0		
6					0	0	
7						0	0

Example: recursive case (I)

- S[1..7]=ACCAGCU

C	1	2	3	4	5	6	7
1	0	0	0				
2	0	0	0	0			
3		0	0	0			
4			0	0	0		
5				0	0	1	
6					0	0	0
7						0	0

V (3,5)=max number of base pairs in S[3..5].

By the recursive formula,
$V(3,5)=\max\{V(4,4)+\delta(S[3],S[5]),$
$\max_{3\le k<5}V(3,k)+V(k+1,5)\} =$
$\max\{V(4,4)+1, V(3,3)+V(4,5),$
$V(3,4)+V(5,5)\} = 1$

Example: recursive case (II)

- S[1..7]=ACCAGCU

C	1	2	3	4	5	6	7
1	0	0	0	0			
2	0	0	0	0	1		
3		0	0	0	1	1	
4			0	0	0	1	
5				0	0	1	1
6					0	0	0
7						0	0

V (4,7)=max number of base pairs in S[4..6].

By the recursive formula,
$V(4,7)=\max\{V(5,6)+\delta(S[4],S[7]),$
$\max_{4\le k<7}V(4,k)+V(k+1,7)\} =$
$\max\{V(5,6)+1, V(4,4)+V(5,7),$
$V(4,5)+V(6,7), V(4,6)+V(7,7)\} = 2$

Example: recursive case (III)

- S[1..7]=ACCAGCU

C	1	2	3	4	5	6	7
1	0	0	0	0	1	1	2
2	0	0	0	0	1	1	2
3		0	0	0	1	1	2
4			0	0	0	1	2
5				0	0	1	1
6					0	0	0
7						0	0

Nussinov folding algorithm (IV)

- Time analysis:
  - We need to fill-in $O(n^2)$ V(i,j) entries
  - Each V(i,j) entry can be computed in O(n) time.
  - Thus, Nussinov algorithm can be solved in $O(n^3)$ time.

Predicting RNA secondary structure by energy minimization

- The best solution is energy minimization (thermodynamic model) based on dynamic programming
  - Idea:
    - bases that are bonded tend to stabilize the structure
    - unpaired bases which form loops tend to destabilize the structure

Software

- This dynamic programming solution has been implemented in two important RNA folding softwares
  - Zuker MFOLD algorithm
    - http://bioinfo.math.rpi.edu/~zukerm/rna/
  - Vienna package
    - http://www.tbi.univie.ac.at/~ivo/RNA/

Thermodynamic energy model

- Assume there is no pseudoknot.
- Thermodynamic model says

1. Every loop's energy is independent of the other loops.
2. Energy of a secondary structure is the sum of the energies of all loops

Loop energy

- eS(i, j): free energy of the stacking pair consists of base pairs (i, j) and (i+1,j-1). Stacking pair stabilizes the structure and has a negative energy
- eH(i, j): free energy of the hairpin closed by the base pair (i, j)
- eL(i,j,i',j'): free energy of an internal loop or bulge enclosed by (i, j) and consists of 2 base pairs.
- eM(i,j,i_1,j_1,…,i_k,j_k): free energy of a multi-loop enclosed by (i, j) and consists of k+1 base pairs.

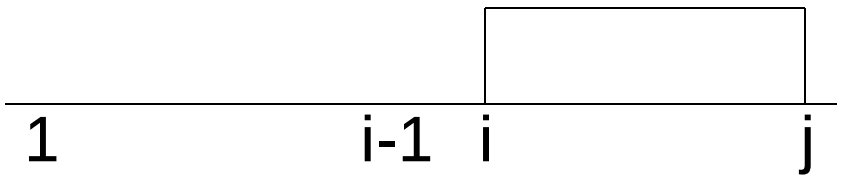
How to find the minimum energy secondary structure?

- Similar to finding optimal alignment, we use dynamic programming
- W(j): energy of the optimal secondary structure for S[1..j]
- V(i, j): energy of the optimal secondary structure for S[i..j] with (i, j) forms a base pair
- VBI(i, j): energy of the optimal secondary structure for S[i..j] with (i, j) closes a bulge or internal loop
- VM(i, j): energy of the optimal secondary structure for S[i..j] with (i, j) closes a multi-loop

W(j)

W(j) find the free energy of the optimal secondary structure for S[1..j]

- W(0) = 0
- For j>0,
  - $$W(j) = \min \begin{cases} W(j-1), & \text{j is free} \\ \min\limits_{1 \le i < j} \{V(i,j) + W(i-1)\} & \text{j pairs with i} \end{cases}$$



V(i, j)

V(i, j) find the free energy of the optimal secondary structure for S[i..j] with (i, j) forms a base pair.

- If i≥j, V(i, j) is undefined.
- If i<j,
  - $$V(i,j) = \min \begin{cases} eH(i,j) & \text{Hairpin} \\ eS(i,j) + V(i+1, j-1) & \text{Stacking pair} \\ VBI(i,j) & \text{Bulge/Internal loop} \\ VM(i,j) & \text{Multi-loop} \end{cases}$$

VBI(i, j)

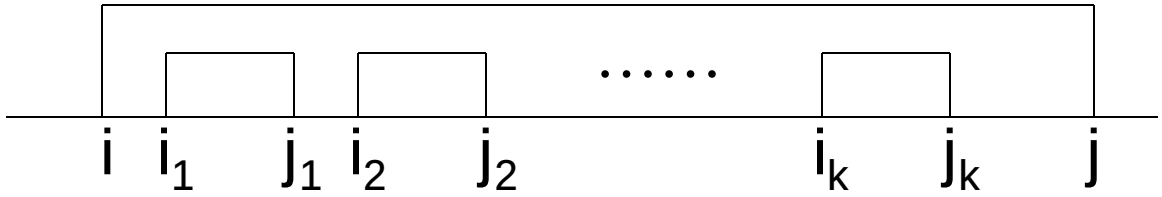
VBI(i, j) finds the free energy of the optimal secondary structure for S[i..j] with (i, j) closes a bulge or internal loop

- $$VBI(i, j) = \min_{\substack{i', j' \\ i<i'<j'<j}} \{eL(i, j, i', j') + V(i', j')\}$$

VM(i, j)

VM(i, j) finds the free energy of the optimal secondary structure for S[i..j] with (i, j) closes a multi-loop

- $$VM(i,j) = \min_{\substack{k, i_1, j_1, \ldots, i_k, j_k \\ i < i_1 < j_1 < \ldots < i_k < j_k < j}} \left\{ eM(i, j, i_1, j_1, \ldots, i_k, j_k) + \sum_{h=1}^{k} V(i_h, j_h) \right\}$$



Time analysis

- W(i): n entries, each requires finding minimum of n terms. In total, $O(n^2)$ time.
- V(i, j): n^2 entries, each requires finding minimum of 4 terms. In total, $O(n^2)$ time.
- VBI(i, j): n^2 entries, each requires finding minimum of $O(n^2)$ terms. In total, $O(n^4)$ time.
- VM(i, j): n^2 entries, each requires finding minimum of exponential terms. In total, exponential time.

- Total time is exponential!

Speedup

- Multi-loop: approximate it with affline linear function
  - Execution time: $O(n^3)$
- Internal loop: ninio equation
  - Execution time: $O(n^3)$

- We will go through the multi-loop speed-up.

Approximating free energy for multi-loop

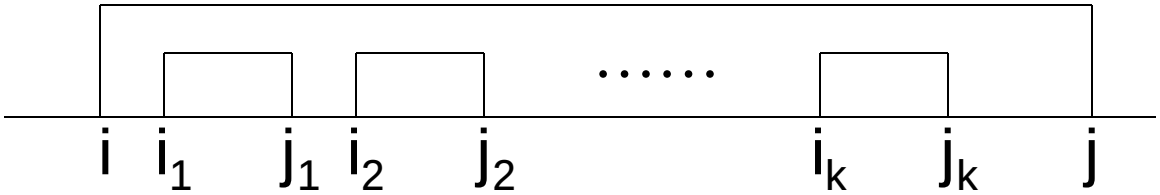
- Bottleneck is VM.
- To reduce the time, we approximate free energy for multi-loop using an affine linear function.

$$eM(i, j, i_1, j_1, \ldots, i_k, j_k) = a + bk + c\left((i_1 - i - 1) + (j - j_k - 1) + \sum_{h=1}^{k-1} (i_{h+1} - j_h - 1) \right)$$

where a, b, c are constant

Number of base-pairs

Number of free base



Speedup for multi-loop

- WM(i, j):free energy of a subregion i..j of the multi-loop region.

$$WM(i,j)=\min\begin{cases} WM(i,j-1)+c, & \text{j is free} \\ WM(i+1,j)+c, & \text{i is free} \\ V(i,j)+b, & \text{(i, j) is a pair} \\ \min\limits_{i<k\le j}\{WM(i,k-1)+WM(k,j)\} & \text{i and j are not free and (i, j) is not a pair} \end{cases}$$

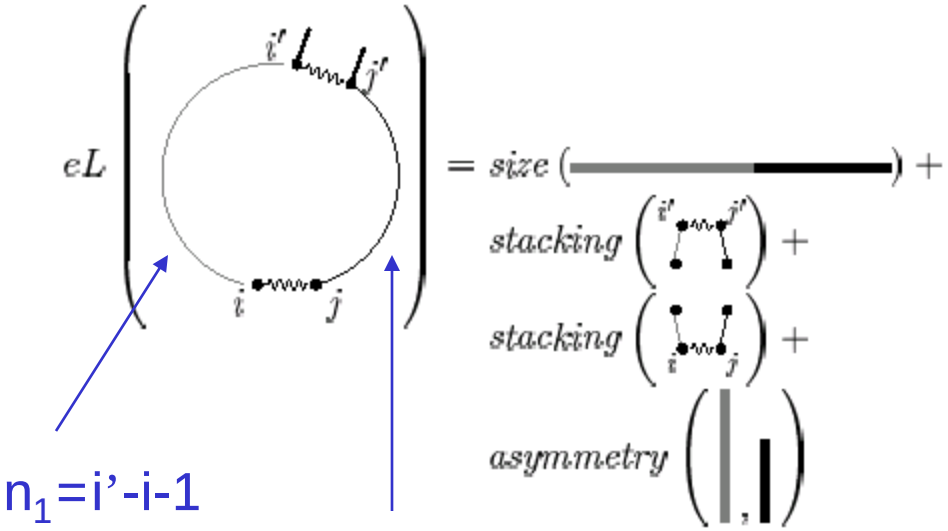
$$VM(i,j)=WM(i+1,j-1)+a$$

Time analysis

- WM(i, j): n^2 entries, each can be computed in O(n) time. In total, $O(n^3)$ time.
- VM(i, j): n^2 entries, each can be computed in O(n) time. In total, $O(n^3)$ time.

Assumption for internal loop/ bulge free energy

- eL(i, j, i', j') = size(n_1+n_2) + stacking(i, j) + stacking(i', j') +asymmetry(n_1, n_2)



n_1=i'-i-1

n_2=j-j'-1

- size( ): energy depends on loop size
- stacking( ): energy for the mismatched base pair adjacent to the base pair
- asymmetry( ): asymmetry penalty

Asymmetry Function Assumption

- We further assume that when $n_1, n_2 > c$, asymmetry(n_1, n_2) is only depend on the difference of n_1 and n_2. In other word,
  - asymmetry(n_1, n_2) = asymmetry(n_1-1, n_2-1) when n_1, n_2 >c

- Currently, we use Ninio equation, which is
  - asymmetry(n_1, n_2) = min{K, $|n_1-n_2|f(m)$}
    where m=min {n_1, n_2, c}, K and c are constants.
  - Note that asymmetry(n_1, n_2) satisfies the above assumption.
  - c is proposed to be 1 and 5 in two literatures.

Refined equation

- Let n_1=i'-i-1, n_2=j-j'-1, l=n_1+n_2.
- For n_1>c and n_2>c, we have

$$eL(i, j, i', j') - eL(i+1, j-1, i', j')$$
$$= size(l) - size(l-2) + stacking(i, j) - stacking(i+1, j-1)$$

- Proof:

$$\begin{aligned} & eL(i,j,i',j') - eL(i+1,j-1,i',j') \\ = \; & [size(\ell) + stacking(i,j) + stacking(i',j') + asymmetry(n_1,n_2)] - \\ & [size(\ell-2) + stacking(i+1,j-1) + stacking(i',j') + asymmetry(n_1-1,n_2-1)] \\ = \; & size(\ell) - size(\ell-2) + stacking(i,j) - stacking(i+1,j-1) \end{aligned}$$

VBI”

$$VBI''(i,j,l) = \min_{\substack{i<i'<j'<j \\ i'-i-1+j-j'-1=l \\ i'-i-1,\, j-j'-1>c}} \{eL(i,j,i',j') + V(i',j')\}$$

- By previous slide, we have

$$VBI''(i,j,l) = VBI''(i+1,j-1,l)$$
$$+ size(l) - size(l-2) + stacking(i,j) - stacking(i+1,j-1)$$

- For running time, there are O(n^3) entries for VBI”(i,j,l). Each entry can be computed in constant time.
  - Hence, all entries in VBI” can be computed in O(n^3) time.

Speedup for internal loop

$$VBI(i,j) = \min_{0<l\le n} \begin{cases} VBI''(i,j,l) \text{ if } l > c \\ \min_{1\le d\le c} V(i+d, j-l-d) + eL(i,j,i+d, j-l+d-2) \\ \min_{1\le d\le c} V(i+l+d, j-d) + eL(i,j,i+l+d+2, j-d) \end{cases}$$

- There are $O(n^2)$ entries for the table VBI. Each entry can be computed in O(n) time.
- In total, the RNA secondary structure prediction problem can be solved in $O(n^3)$ time.

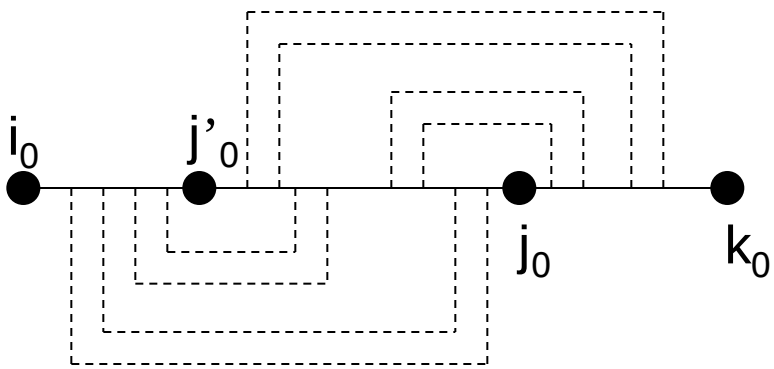
RNA secondary structure prediction with pseudoknots

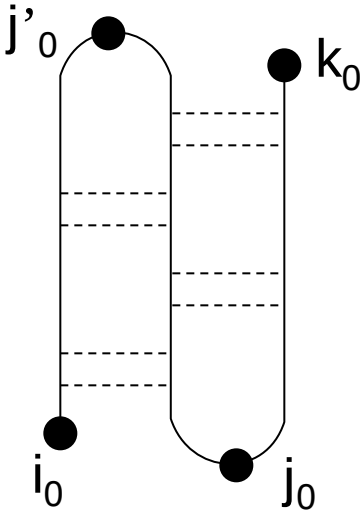
- Up to now, there is no good way to predict RNA secondary structure with pseudoknots.
- In fact, predicting RNA secondary structure with pseudoknots is a NP-hard problem.
- This section considers RNA secondary structure prediction with a particular kind of pseudoknot --- simple pseudoknot!

Simple Pseudoknot

A set of base pairs $M_{i0,k0}$ is a simple pseudoknot if there exist $i_0<j'_0<j_0<k_0$ such that

- Each endpoint i appear in $M_{i0,k0}$ once.
- Each $(i, j) \in M_{i0,k0}$ satisfies either $i_0 \leq i < j'_0 < j \leq j_0$ or $j'_0 \leq i < j_0 < j \leq k_0$
- If pairs (i, j) and (i', j') in $M_{i0,k0}$ satisfies either $i < i' < j'_0$ or $j'_0 \leq i < i'$, then $j > j'$.





RNA secondary structure with simple pseudoknots

- A set of base pairs M is called an RNA secondary structure with simple pseudoknots if
  - $M = M' \cup M_1 \cup M_2 \ldots \cup M_t$
  - M_h is a simple pseudoknot for $S[i_h..k_h]$ where $1 \leq i_1 < k_1 < i_2 < k_2 < \ldots < i_t < k_t \leq n$
  - M' is secondary structure without pseudoknots for string S' where S' is obtained by deleting all $S[i_h..k_h]$

Problem

- Input: an RNA sequence S[1..n]
- Output: an RNA secondary structure with simple pseudoknots
  - maximizing the number of base pairs

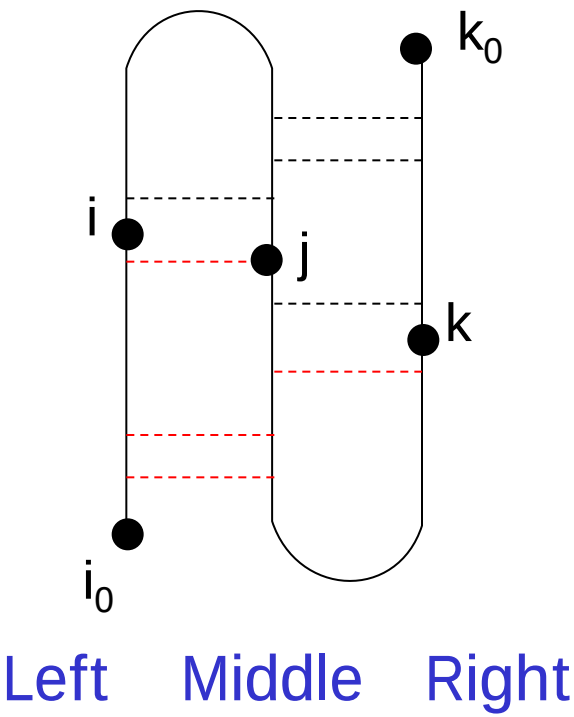
Dynamic programming for Simple Pseudoknot

- V(i, j): maximum number of base pairs in S[i..j]
- V_{pseudo}(i, j): maximum number of base pairs of a pseudoknot in S[i..j]
- $$V(i,j) = \max \begin{cases} V_{pseudo}(i,j) \\ V(i+1, j-1) + \delta(S[i], S[j]) \\ \max\limits_{i<k\le j} \{V(i,k-1) + V(k,j)\} \end{cases}$$
- V(i, i) = 0 for any i
- Note: δ(S[i], S[j]) is 1 if S[i] and S[j] are complement and 0, otherwise.
- Suppose, for all i and j, V_{pseudo}(i, j) are available. The table V can be filled in using $O(n^3)$ time.

Terminology

- What remain is to compute $V_{pseudo}(i_0, k_0)$.

- Given a set of base pairs in a simple pseudoknot for $S[i_0, k_0]$,
  - A base pair is said to be below the triplet (i, j, k) if they are the red edges.



Computing $V_{pseudo}(i_0, k_0)$ (I)

- For $i_0<i<j<k<k_0$, we define
  - $V_L(i,j,k)$ be the maximum number of base pairs below the triplet (i, j, k) in a pseudoknot for $S[i_0..k_0]$ with (i, j) is a base pair
  - $V_R(i,j,k)$ be the maximum number of base pairs below the triplet (i, j, k) in a pseudoknot for $S[i_0..k_0]$ with (j, k) is a base pair
  - $V_M(i,j,k)$ be the maximum number of base pairs below the triplet (i, j, k) in a pseudoknot for $S[i_0..k_0]$ with both (i, j) and (j, k) are not a base pair

- Note: $\max\{V_L(i,j,k), V_M(i,j,k), V_R(i,j,k)\}$ is the maximum number of base pairs below the triplet (i, j, k) in a pseudoknot for $S[i_0..k_0]$

Computing $V_{pseudo}(i_0, k_0)$ (II)

$$V_{pseudo}(i_0, k_0) = \max_{i_0 \le i < j < k \le k_0} \{V_L(i, j, k), V_M(i, j, k), V_R(i, j, k)\}$$

$V_L(i, j, k)$

- $$V_L(i, j, k) = \delta(S[i], S[j]) + \max\begin{Bmatrix} V_L(i-1, j+1, k) \\ V_M(i-1, j+1, k) \\ V_R(i-1, j+1, k) \end{Bmatrix}$$

- $V_L(i, j, k)$ means (i, j) is a pair.
- Thus, $V_L(i, j, k)$ is equal to $\delta(S[i], S[j])$ plus the maximum number of base pairs below (i-1, j+1, k)

$V_R(i, j, k)$

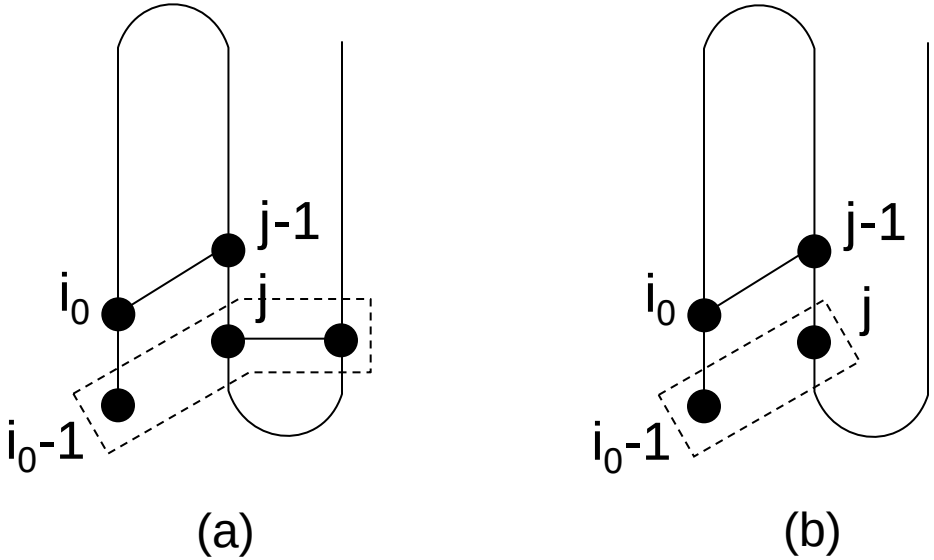
- Similarly, we have
- $$V_R(i, j, k) = \delta(S[j], S[k]) + \max\begin{Bmatrix} V_L(i, j+1, k-1) \\ V_M(i, j+1, k-1) \\ V_R(i, j+1, k-1) \end{Bmatrix}$$

$V_M(i, j, k)$

$$V_M(i, j, k) = \max\left\{\begin{array}{l} V_M(i-1, j, k), V_M(i, j+1, k), V_M(i, j, k-1), \\ V_L(i-1, j, k), V_L(i, j+1, k), \\ V_R(i, j+1, k), V_R(i, j, k-1) \end{array}\right\}$$

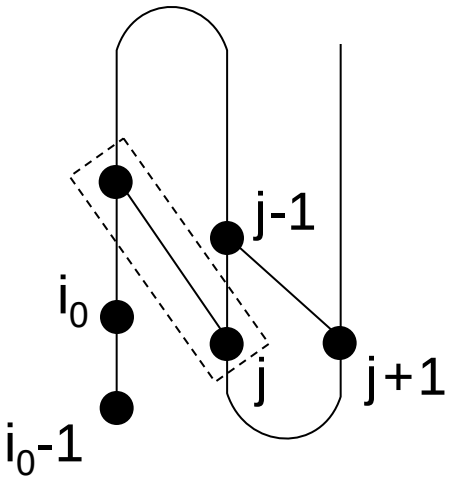
Basis

- V_R
  - $V_R(i_0\text{-}1,j,k)=0$ if $k=j$ or ($k=j+1$ and $S[j]$ and $S[k]$ does not form a base pair) (see figure (b))
  - $V_R(i_0\text{-}1,j,j+1)=1$ if $S[j]$ and $S[j+1]$ forms a base pair (see figure (a))



Basis

- V_L
  - $V_L(i,j,j) = \delta(S[i], S[j])$ for all $i<j$
    - (see figure)
  - $V_L(i_0-1,j,k)=0$ if $k=j$ or $k=j+1$
- V_M
  - $V_M(i_0-1,j,k)=0$ if $k=j$ or $k=j+1$



(a)

Time complexity for computing $V_{pseudo}(i_0, k_0)$ (I)

- For a fixed i_0, k_0, the basis can be computed in $O(n)$ time
- V_L, V_R, V_M can be computed in $O(n^3)$ time.
- Thus, for every i_0,k_0, $V_{pseudo}(i_0,k_0)$ can be computed in $O(n^3)$ time.
- It takes $O(n^5)$ time to compute $V_{pseudo}(i_0,k_0)$ for all $i_0<k_0$

Time complexity for computing $V_{pseudo}(i_0, k_0)$ (II)

- Can we further improve it?
- Note that the basis only depends on i_0
- Thus, for a fixed i_0, for any k_0,
  - the values of table V_R, V_L, V_M are the same.
  - We can compute $V_{pseudo}(i_0,k_0)$ for a fixed i_0 and for any k_0 in $O(n^3)$ time.
- In total, it takes $O(n^4)$ time to compute $V_{pseudo}(i_0,k_0)$ for all $i_0<k_0$

Conclusion

- The table V_{pseudo} can be filled in using $O(n^4)$
- The table V can be filled in using $O(n^3)$
- Thus, the RNA secondary structure problem with simple pseudoknots can be solved in $O(n^4)$ time.