

# Outline

RNA alignment problem

Structural alignment definition

NP-hard results

Algorithms

Results

## Ribonucleic Acid (RNA)

A ribonucleic acid (RNA) molecule is made up of a long chain of subunits – ribonucleotides – linked together.

Each ribonucleotide contains one of four possible bases, abbreviated A(adenine), C(cytosine), G(guanine), and U(uracil).

It is the sequence of bases that distinguishes one type of RNA from another.

This base sequence is called the primary structure of the RNA molecule.

Ribonucleic Acid (RNA) is an important molecule which performs a wide range of functions in biological system.

In particular it is RNA (not DNA) that contains genetic information of virus such as HIV and therefore regulates the functions of such virus.

It is well known that secondary and tertiary structural features of RNAs are important in the molecular mechanism involving their functions.

RNA has recently become the center of much attention because of its catalytic properties, leading to an increased interest in obtaining its structural information.

The secondary or tertiary structure of an RNA is a set of base-pairs (ribonucleotide pairs) which formed bonds between A·U, C·G, and G·U.

An RNA structure can be represented by  $R(P)$ , where  $R = r[1], r[2], \dots, r[n]$  is a sequence of nucleotides, and  $P \subset \{1, 2, \dots, n\}^2$  is a set of pairs of which each element  $(i, j)$  represents a base pair  $(r[i], r[j])$  in  $R$ .

We assume that base pairs in  $R(P)$  do not share participating bases. Formally any  $(i_1, j_1)$  and  $(i_2, j_2)$  in  $P$ ,  $j_1 \neq i_2$ ,  $i_1 \neq j_2$ , and  $i_1 = i_2$  if and only if  $j_1 = j_2$ .

For an RNA  $R(P)$ , we define  $p_r(\ )$  as follows.

$$p_r(i) = \begin{cases} i & \text{if } r[i] \text{ is an unpaired base} \\ j & \text{if } (r[i], r[j]) \text{ or } (r[j], r[i]) \text{ is a base pair in } P \end{cases}$$

$p_r(i) \neq i$  if and only if  $r[i]$  is a base in a base pair.

$p_r(i) = i$  if and only if  $r[i]$  is an unpaired base.

If  $p_r(i) \neq i$ , then  $p_r(i)$  is the base paired with base  $i$ .

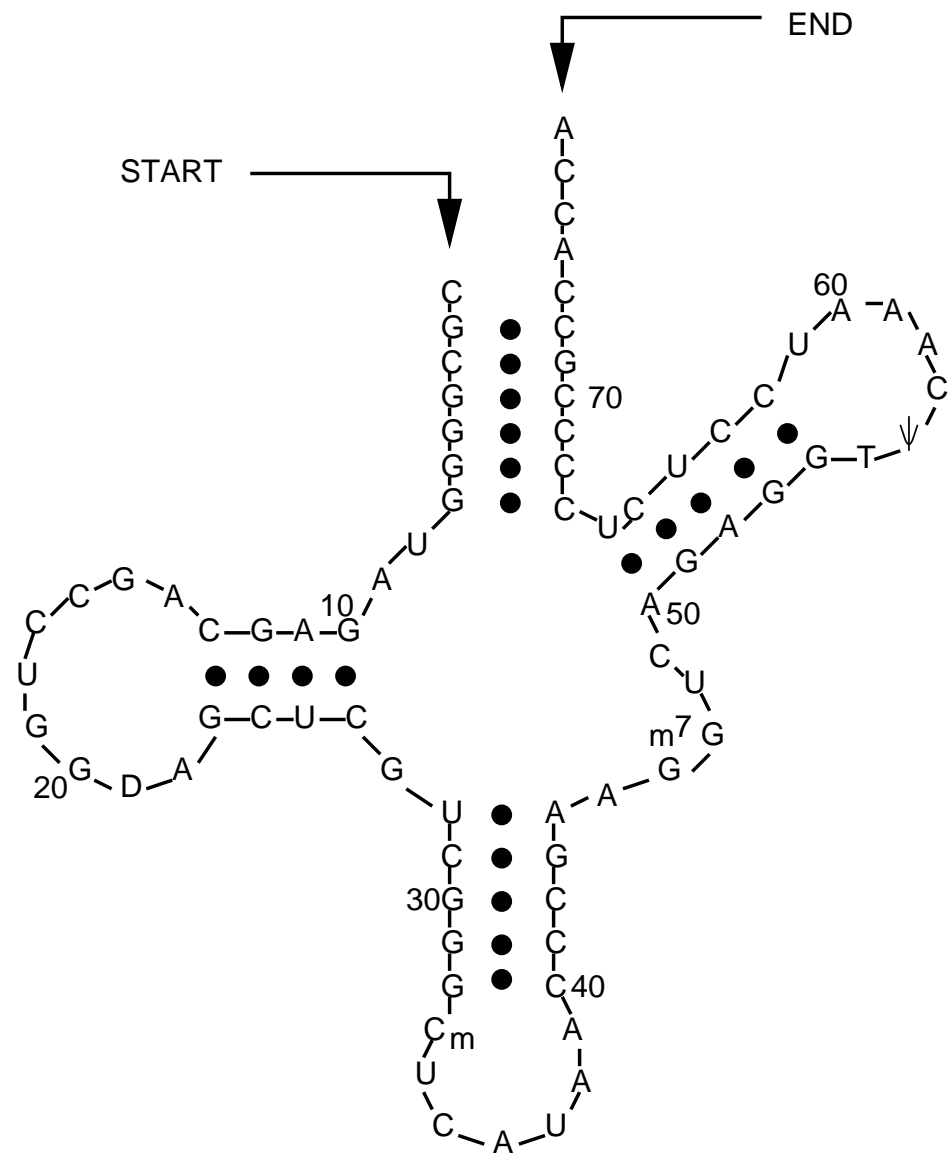
For two base pairs  $r = (r_1, r_2)$  and  $s = (s_1, s_2)$  in  $R(P)$ , we say  $r$  and  $s$  are crossing if  $r_1 < s_1 < r_2 < s_2$  or  $s_1 < r_1 < s_2 < r_2$ .

For secondary structure, these bonds have been traditionally assumed to be one-to-one and non-crossing. Therefore secondary structure can be represented as a tree (or forest)

For tertiary structure, there is no restriction of non-crossing. Tertiary structure can be considered as a special graph.

An RNA secondary structure can be described in a unique and natural way as made up of substructures called *stacked pairs, loops, and external single stranded regions* .

- Stacked pairs (also called Helical Stem).
- Loops:
  - Hairpin loop
  - Bulge loop
  - Interior loop
  - Multiple loop
- External single stranded regions.



The secondary structure of a transfer RNA molecule. Letters other than A, G, C, U indicate chemical modifications of these four basic units.

## Why consider alignment?

- search a database.
- inferring structure:  
from the primary structure of an RNA when a closely related RNA's structure is given.
- predicting secondary structure:  
from a set of closely related primary structures to predict common secondary structures.

## Similarity measures

Following the tradition in sequence comparison, we define three operations, relabel, delete, and insert, on RNA structure.

For a given RNA structure, each operation can be applied to either a base-pair or an unpaired base.

### **For unpaired bases:**

- Relabel an unpaired base is to replace it with another base.
- Delete an unpaired base is to delete the base from the sequence.
- Insert a base is to insert a new base into the sequence as an unpaired base.

## **For base-pairs:**

- Relabel: Relabel a base-pair is to replace one base-pair with another.  
This means that at the sequence level, two bases may be changed at the same time.
- Delete: Delete a base-pair is to delete the base-pair from the structure.  
At the sequence level, this means to delete two bases at the same time.
- Insert: Insert a base-pair is to insert a new base-pair into the structure.  
At the sequence level, this means to insert two bases at the same time.

We assume that there is a cost function associated with the operations.

We can now consider how to align one RNA structure with another using minimum number of weighted operations.

### **Alignment between RNA structures**

Given two RNA structures  $R_1$  and  $R_2$ , a structural alignment of  $R_1$  and  $R_2$  is represented by  $(R'_1, R'_2)$  satisfying the following conditions.

- 1)  $R'_1$  and  $R'_2$  are  $R_1$  and  $R_2$  with some new symbol  $-$  inserted such that  $|R'_1| = |R'_2|$ .
- 2) If  $r'_1[i]$  is an unpaired base in  $R_1$ , then either  $r'_2[i]$  is an unpaired base in  $R_2$  or  $r'_2[i] = -$ .  
If  $r'_2[i]$  is an unpaired base in  $R_2$ , then either  $r'_1[i]$  is an unpaired base in  $R_1$  or  $r'_1[i] = -$ .
- 3) If  $(r'_1[i], r'_1[j])$  is a base pair in  $R_1$ , then either  $(r'_2[i], r'_2[j])$  is a base pair in  $R_2$  or  $r'_2[i] = r'_2[j] = -$ .  
If  $(r'_2[i], r'_2[j])$  is a base pair in  $R_2$ , then either  $(r'_1[i], r'_1[j])$  is a base pair in  $R_1$  or  $r'_1[i] = r'_1[j] = -$ .

## Properties of alignments

Alignments satisfy the following three conditions:

1. one-to-one;
2. base to base, base pair to base pair;
3. respects topological relationships.

Given an alignment  $(R'_1, R'_2)$ , we define  $S_M, S_D, S_I, P_M, P_D,$  and  $P_I$  as follows.

$$\begin{aligned}
S_M &= \left\{ i \mid \begin{array}{l} r'_1[i] \text{ is an unpaired base in } R_1, \\ r'_2[i] \text{ is an unpaired base in } R_2. \end{array} \right\} \\
S_D &= \left\{ i \mid \begin{array}{l} r'_1[i] \text{ is an unpaired base in } R_1, \\ r'_2[i] = -. \end{array} \right\} \\
S_I &= \left\{ i \mid \begin{array}{l} r'_2[i] \text{ is an unpaired base in } R_2, \\ r'_1[i] = -. \end{array} \right\} \\
P_M &= \left\{ (i, j) \mid \begin{array}{l} (r'_1[i], r'_1[j]) \text{ is a base pair in } R_1, \\ (r'_2[i], r'_2[j]) \text{ is a base pair in } R_2. \end{array} \right\} \\
P_D &= \left\{ (i, j) \mid \begin{array}{l} (r'_1[i], r'_1[j]) \text{ is a base pair in } R_1, \\ r'_2[i] = r'_2[j] = -. \end{array} \right\} \\
P_I &= \left\{ (i, j) \mid \begin{array}{l} (r'_2[i], r'_2[j]) \text{ is a base pair in } R_2, \\ r'_1[i] = r'_1[j] = -. \end{array} \right\}
\end{aligned}$$

Let  $\Gamma$  be a cost function for edit operations on base pairs and unpaired bases.

The cost of an alignment  $(R'_1, R'_2)$  is defined as follows.

$$\begin{aligned}
 \text{cost}((R'_1, R'_2)) = & \\
 & \sum_{i \in S_M} \Gamma(r'_1[i] \rightarrow r'_2[i]) \\
 + & \sum_{i \in S_D} \Gamma(r'_1[i] \rightarrow \lambda) \\
 + & \sum_{i \in S_I} \Gamma(\lambda \rightarrow r'_2[i]) \\
 + & \sum_{(i,j) \in P_M} \Gamma((r'_1[i], r'_1[j]) \rightarrow (r'_2[i], r'_2[j])) \\
 + & \sum_{(i,j) \in P_D} \Gamma((r'_1[i], r'_1[j]) \rightarrow \lambda) \\
 + & \sum_{(i,j) \in P_I} \Gamma(\lambda \rightarrow (r'_2[i], r'_2[j]))
 \end{aligned}$$

Given two RNA structures, Our goal is to find the alignment with minimum cost.

$$D(R_1(P_1), R_2(P_2)) = \min_{(R'_1, R'_2)} \{ \text{cost}((R'_1, R'_2)) \}$$

Let  $\Gamma$  be a cost function for edit operations on base pairs and unpaired bases and  $G$  be a gap opening cost.

The cost of an alignment  $(R'_1, R'_2)$  is defined as follows.

$$\begin{aligned}
 \text{cost}((R'_1, R'_2)) = & \\
 & G \times \#gaps \\
 & + \sum_{i \in S_M} \Gamma(r'_1[i] \rightarrow r'_2[i]) \\
 & + \sum_{i \in S_D} \Gamma(r'_1[i] \rightarrow \lambda) \\
 & + \sum_{i \in S_I} \Gamma(\lambda \rightarrow r'_2[i]) \\
 & + \sum_{(i,j) \in P_M} \Gamma((r'_1[i], r'_1[j]) \rightarrow (r'_2[i], r'_2[j])) \\
 & + \sum_{(i,j) \in P_D} \Gamma((r'_1[i], r'_1[j]) \rightarrow \lambda) \\
 & + \sum_{(i,j) \in P_I} \Gamma(\lambda \rightarrow (r'_2[i], r'_2[j]))
 \end{aligned}$$

Given two RNA structures, Our goal is to find the alignment with minimum cost.

$$D(R_1(P_1), R_2(P_2)) = \min_{(R'_1, R'_2)} \{ \text{cost}((R'_1, R'_2)) \}$$

## NP-hard result

Computing optimal alignment between two RNA tertiary structures is NP-hard. (It is also MAX SNP-hard.)

The reduction is from the 3-SAT problem.

Let  $S = C_1 \cdot C_2 \cdots C_n$ , where  $C_i = (v_{i_1} \cup v_{i_2} \cup v_{i_3})$ , be an instance of 3-SAT problem.

We can construct, in polynomial time, two RNA structures  $R_1$  and  $R_2$  as in Figure 1 and Figure 2.

Let the number of base pairs in  $R_1$  and  $R_2$  be  $N_1$  and  $N_2$ . Assuming that each operation has unit cost, we have the following lemma.

### **Lemma**

$S$  can be satisfied if and only if  
 $D(R_1, R_2) = N_2 - N_1$ .

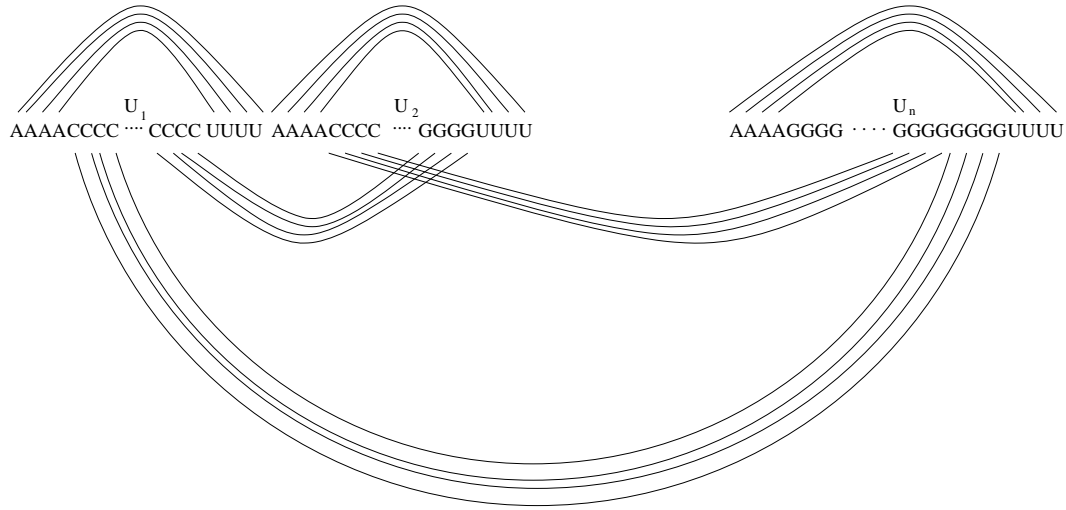


Figure 1: RNA structure 1.

In  $R_1$ , there are  $n$  segments each of which is enclosed by four base pairs. These base pairs are all A·U pairs.

And each segment is connected to every other segment by four base pairs of C·G type. Let the number of base pairs in  $R_1$  be  $N_1$

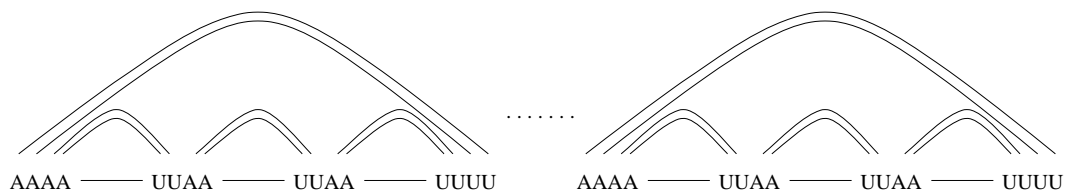


Figure 2: RNA structure 2.

In  $R_2$ , for each  $v_{i_k}$ , there is a corresponding segment which is enclosed by two base pairs of A·U type.

Each clause  $C_i$  is then represented by segments of  $v_{i_1}, v_{i_2}, v_{i_3}$  and is enclosed by another two base pairs of A·U type. Let the number of base pairs in  $R_2$  be  $N_2$

Each  $v_{i_k}$  is connected to  $v_{j_l}$ , by C·G pairs, if they are not complement of each other.

## Algorithms

When both RNAs are secondary structures, since there is no crossing, we can represent RNA structures as ordered forests and then use the tree edit distance algorithm to solve this problem.

To deal with tertiary structures, we now consider an algorithm where aligned elements in the alignment have no crossing.

An extension of our algorithm can handle the case where both RNAs are tertiary structures with only H-type pseudo-knots, one stem crosses with at most one other stem. (A stem in an RNA is a set of stack pairs of maximum size.)

## Notation

We use  $\Gamma(\ )$  to define  $\gamma(i, j)$ .

If  $i = p_{r_1}(i)$  and  $j = p_{r_2}(j)$ ,

$$\begin{aligned}\gamma(i, 0) &= \Gamma(r[i] \rightarrow \lambda) \\ \gamma(0, j) &= \Gamma(\lambda \rightarrow r_2[j]) \\ \gamma(i, j) &= \Gamma(r_1[i] \rightarrow r_2[j])\end{aligned}$$

If  $i' = p_{r_1}(i) < i$  and  $j' = p_{r_2}(j) < j$ ,

$$\begin{aligned}\gamma(i', 0) &= \Gamma((r_1[i'], r_1[i]) \rightarrow \lambda)/2 \\ \gamma(i, 0) &= \Gamma((r_1[i'], r_1[i]) \rightarrow \lambda)/2 \\ \gamma(0, j') &= \Gamma(\lambda \rightarrow (r_2[j'], r_2[j]))/2 \\ \gamma(0, j) &= \Gamma(\lambda \rightarrow (r_2[j'], r_2[j]))/2 \\ \gamma(i, j) &= \Gamma((r_1[i'], r_1[i]) \rightarrow (r_2[j'], r_2[j]))\end{aligned}$$

We use  $D(i_1, i_2; j_1, j_2)$  to represent the cost of an optimal alignment between  $R_1[i_1..i_2]$  and  $R_2[j_1..j_2]$ .

**How to compute  $D(i_1, i_2 ; j_1, j_2)$ ?**

**If both  $r_1[i]$  and  $r_2[j]$  are single base,**

$$D(i_1, i ; j_1, j) = \min \begin{cases} D_1(i_1, i - 1 ; j_1, j) & + \gamma(i, 0) \\ D_1(i_1, i ; j_1, j - 1) & + \gamma(0, j) \\ D_1(i_1, i - 1 ; j_1, j - 1) & + \gamma(i, j) \end{cases}$$

**If both  $r_1[i]$  and  $r_2[j]$  are a base in a base pair and  $i_1 \leq p_{r_1}(i) < i$ ,  $j_1 \leq p_{r_2}(j) < j$ ,**

$$D(i_1, i ; j_1, j) = \min \begin{cases} D(i_1, i - 1 ; j_1, j) + \gamma(i, 0) \\ D(i_1, i ; j_1, j - 1) + \gamma(0, j) \\ D(i_1, p_{r_1}(i) - 1 ; j_1, p_{r_2}(j) - 1) \\ \quad + D(p_{r_1}(i) + 1, i - 1 ; p_{r_2}(j) + 1, j - 1) + \gamma(i, j) \end{cases}$$

**Otherwise,**

$$D(i_1, i ; j_1, j) = \min \begin{cases} D(i_1, i - 1 ; j_1, j) + \gamma(i, 0) \\ D(i_1, i ; j_1, j - 1) + \gamma(0, j) \end{cases}$$

This includes the following cases.

- $r_1[i]$  is a single base and  $r_2[j]$  is a base in a base pair,
- $r_1[i]$  is a base in a base pair and  $r_2[j]$  is a single base,
- $r_1[i]$  is a base in a base pair and  $p_{r_1}(i) < i_1$ ,
- $r_2[j]$  is a base in a base pair and  $p_{r_2}(j) < j_1$ ,
- $r_1[i]$  is a base in a base pair and  $p_{r_1}(i) > i$ ,
- $r_2[j]$  is a base in a base pair and  $p_{r_2}(j) > j$ .

## Algorithm

From the above formulas, we can compute optimal alignment between  $D(1, |R_1| ; 1, |R_2|)$  using bottom up approach.

Moreover, it is clear that we do not need to compute all  $D(i_1, i_2 ; j_1, j_2)$ .

We only need to compute these  $D(i_1, i_2 ; j_1, j_2)$  such that  $(i_1 - 1, i_2 + 1)$  is a base pair in  $R_1$  and  $(j_1 - 1, j_2 + 1)$  is a base pair in  $R_2$ .

Given  $R_1[1..|R_1|]$  and  $R_2[1..|R_2|]$ , we can first determine base pair lists for  $R_1$  and  $R_2$ .

For each pair of base pairs,  $(i_1, i_2)$  and  $(j_1, j_2)$ , we compute  $D(i_1, i_2 ; j_1, j_2)$ .

Finally we compute  $D(1, |R_1| ; 1, |R_2|)$ .

## Complexity

The time to compute  $D(i_1, i_2 ; j_1, j_2)$  is bounded by  $O((i_2 - i_1) \times (j_2 - j_1))$ .

Since  $(i_2 - i_1) \leq |R_1|$  and  $(j_2 - j_1) \leq |R_2|$ , the time complexity of the algorithm is  $O(\text{base\_pairs}(R_1) \times \text{base\_pairs}(R_2) \times |R_1| \times |R_2|)$ .

This time complexity can be improved to  $O(\text{stem}(R_1) \times \text{stem}(R_2) \times |R_1| \times |R_2|)$ . (Recall that a stem is a set of stack pairs of maximum size.)

The space complexity of the algorithm is  $O(|R_1| \times |R_2|)$ .

Note that when one of the RNA is secondary structure, this algorithm compute the optimal solution of the problem.

This algorithm can be modified to handle the case where the input RNAs are tertiary structures with only H-type pseudo-knots (a stem crosses with at most one other stem).

Since the number of tertiary interactions is relatively small compared with the number of secondary interactions, we can also use this algorithm to compute the similarity when both structures are tertiary structures.

## Tertiary structures

Essentially the algorithm tries to find the best secondary structures to align and delete tertiary interactions.

Although this is not an optimal solution, in practice it would produce a reasonable result by aligning most of the base pairs.

A post-processing step can be applied to add some matching tertiary interactions.

We use a two-pass approach: first, find the best secondary structure matching, and second, add in the tertiary interactions, using the results of the first pass as *constraints* on the tertiary computation.

How do we compute a constrained alignment?

Given two RNAs  $R_1$  and  $R_2$  and a mapping  $M$  between them, we want to find the optimal alignment between them given the constraint that for any  $(r, s) \in M$ ,  $r$  is forced to align to  $s$ .

$$D(R_1, R_2, M) =$$

$$\min_{M'} \left\{ \text{cost}(M') \mid \begin{array}{l} M' \text{ is an alignment between} \\ R_1 \text{ and } R_2, \text{ and } M \subset M' \end{array} \right\}$$

How do we do this?

Does it work?

The input usually contains secondary and tertiary interactions of the form  $R(S \cup T)$ , where  $S$  is a set of non-crossing base pairs and  $T$  is a set of tertiary base pairs which cross  $S$ .

The size of  $T$  is usually very small compared to  $S$ , and in addition when comparing  $R_1(S_1 \cup T_1)$  and  $R_2(S_2 \cup T_2)$  it is safe to assume that base pairs in  $S_1$  should be aligned to base pair in  $S_2$  and base pairs in  $T_1$  should be aligned to base pair in  $T_2$ .

## **Our experiments**

- 1 Input:  $R_1(S_1 \cup T_1)$  and  $R_2(S_2 \cup T_2)$ .
- 2 compute  $D(R_1(S_1), R_2(S_2))$ .
- 3 Let  $M$  be the set of aligned bases from step 2.  
compute  $D(R_1(S_1 \cup T_1), R_2(S_2 \cup T_2), M)$ .

## Results

Given  $R_1(S_1 \cup T_1)$  and  $R_2(S_2 \cup T_2)$ :

$$D(R_1(S_1), R_2(S_2)) \leq D(R_1(S_1 \cup T_1), R_2(S_2 \cup T_2)).$$

$$D(R_1(S_1 \cup T_1), R_2(S_2 \cup T_2)) \leq D(R_1(S_1 \cup T_1), R_2(S_2 \cup T_2), M).$$

The following testing data is from The RNase P Database, <http://www.mbio.ncsu.edu/RNaseP/>.

Alcaligenes-eutrophus-pb-b

Anacystis-nidulans-cb

Agrobacterium-tumefaciens-pb-a

Bacillus-brevis-gpb-low

Borrelia-burgdorferi-spi

Bacteroides-thetaiotaomicron-bd

Chlorobium-limicola-gsb

232.0 236.0, 211.0 222.0, 282.0 286.0, 248.0 249.0, 216.0 216.0, 243.0 243.0

207.0 226.0, 300.0 314.0, 194.0 195.0, 194.0 197.0, 241.0 244.0

302.0 314.0, 210.0 211.0, 215.0 228.0, 215.0 217.0

313.0 328.0, 294.0 303.0, 320.0 328.0

193.0 198.0, 228.0 240.0

225.0 225.0

## RNA structure alignment with gap penalty

The alignment produced this way may have the problem of creating many small gaps.

We can improve our alignment algorithm by introducing a gap opening penalty.

The main difficulty is that with a deletion of a base pair, there might be two places with gap openings.

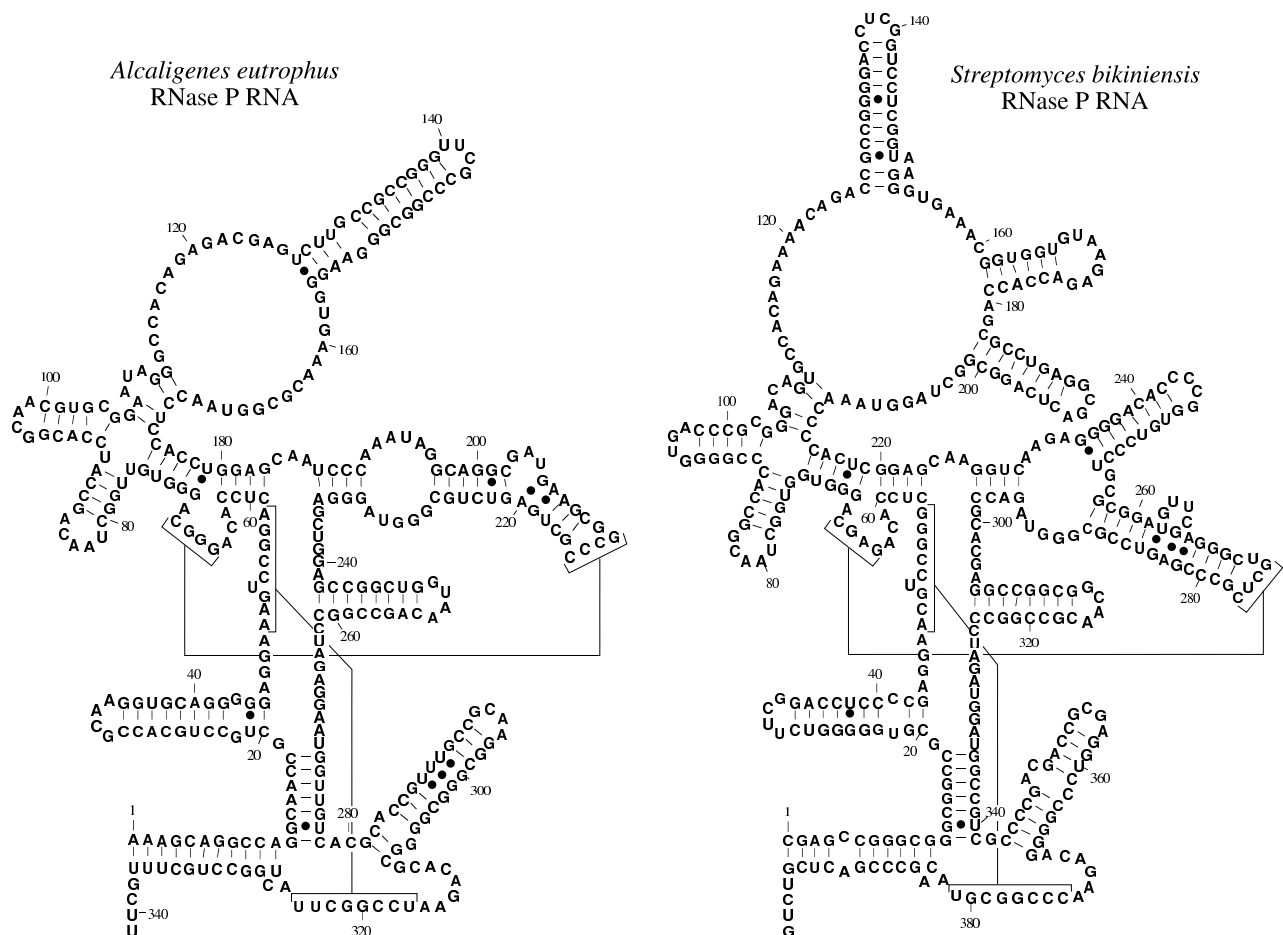
The algorithm is much more complicated.

However, the algorithm is of the same complexity (with a larger constant).

Testing data for alignment with affine gap penalty is again from The RNase P Database.

*Alcaligenes eutrophus*: Beta Purple Bacteria RNase P sequences & structures.

*Streptomyces bikiniensis*: High G+C Gram positive RNase P sequences & structures.



*Alcaligenes eutrophus* and *Streptomyces bikiniensis* from the RNase P database.







## Summary

RNA structure alignment problem.

NP-hard for general tertiary structure alignment.

Algorithm that can handle special cases and provide practical solutions.

A software tool is being built for solving this problem.

[http://www.csd.uwo.ca/~kzhang/rna/rna\\_align.html](http://www.csd.uwo.ca/~kzhang/rna/rna_align.html)