RNA Secondary Structure Alignment Based on Stem Representation *

Meng-Yi Wu, Chang-Biau Yang† and Kuo-Si Huang
Department of Computer Science and Engineering
National Sun Yat-sen University, Kaohsiung, Taiwan
†cbyang@cse.nsysu.edu.tw

Abstract

The comparison methods for RNA or protein molecules are important and basic tools in molecular biology. So far, most comparison methods, such as sequence alignment, are only applicable to the primary structures of biomolecules. Indeed, the functions of biomolecules have close relationship in their structures. The RNA secondary structure alignment problem is to align two given RNA structures to get the structure similarity. In addition, it is also helpful to predict the functions of biomolecules and to classify them. In this paper, we design a dynamic programming method for aligning two RNA secondary structures that do not contain any pseudoknot. The time complexity of our algorithm is $O(N^4)$, where N is the number of blocks contained in the given RNAs. We also apply our algorithm to real biomolecules, the tRNAs of Homo sapiens mitochondrion, to evaluate the practicability of our algorithm. We take three tRNA genes, TRNG, TRNA and TRNV, to test the performance of our algorithm. From the viewpoint of human eyes, the structure of TRNG is more similar to TRNA. Our algorithm also gets this result.

1 Introduction

It is well-known that RNAs play an important role for the biochemical interactions, such as RNA transcription, reverse transcription, protein translation, and mRNA splicing. Some RNAs are also ribozymes, and some of them contain genetic information of viruses, such as HIV, SARS, and flu. The functions of RNAs are corresponding to their structures and RNA-protein interactions. Actually, it is not easy to get the exact structure of RNA, because the structure may be different in various environments, such as temperature and pressure. Because of the importance of RNA structure, some methods have been proposed to predict the RNA structures [7, 10, 13, 17, 20].

The functions of biomolecules have close relationship in their structures. Therefore, the structure comparison methods for RNA or protein molecules are important and they are basic tools in molecular biology. So far, most comparison methods are only applicable to the primary structures of biomolecules. In general, many RNA sequences have very high similarity, but their secondary structures or functions may not be very similar. In other words, many RNAs have similar secondary structures and functions, but their identities may not be high enough. Hence, this paper pays attention to the RNA secondary structure alignment problem. Our goal is to align two RNA molecules with given secondary structures to get the structure similarity.

For distinguishing different RNA structures, our important job is to construct the structure similarity evaluation method between RNAs. Since there are many sequence alignment methods, some researchers try to incorporate the secondary structure information into primary structure [1, 3, 9]. Lin et al. [9] proposed two novel operations, arc-altering and arc-breaking. In an RNA structure alignment, an arc-altering operation means that one nucleotide is deleted and another nucleotide is aligned to a nucleotide of the other RNA, and an arc-breaking operation means that two nucleotides identically are aligned to two nucleotides that are not a base pair of the other RNA. The above two operations make the problem harder, but the biological meanings of these operations may need more discussion.

In another way, an RNA structure may be represented as a tree. Several algorithms try to adopt

*This research work was partially supported by the National Science Council of the Republic of China under contract NSC-92-2213-E-110-005.

the methods for solving tree problems to reconstruct the alignment and to evaluate the similarity between two RNAs [4–6, 11, 14, 15, 18]. Some methods [5, 6, 14, 15] do not use RNA nucleotide information, because it is difficult to represent the stems and loops in a tree. Zhang [18] proposed a method to solve WLCS (weighted largest common sub-structure) problem on two RNA trees by applying the ordered tree edit distance algorithm [19].

Lu et al. proposed another measurement of edit distance between labeled trees [11], which can be applied on the comparison of RNA secondary structures. If RNA secondary structures can be represented as labeled trees, one may calculate the edit distances between these labeled trees for comparing and classifying RNA secondary structures. Their method is an alternative measurement of the similarity between RNA secondary structures.

Form the viewpoint of RNA structures with psudoknots, some researchers also studied the RNA structure alignment problem [2,8,12,16]. For two given RNAs, which one structure is known and the other structure is unknown, Lenhof et al. [8] proposed a polyhedral approach to align the two RNAs. The approach can be applied to two RNA structures which have tertiary interactions or pseudoknot. If the structures of two RNAs are known, there are many methods to calculate the similarity or to align these two RNAs [2, 12, 16]. Ma et al. [12] proved that calculating edit distance of two RNAs is an NP-hard problem, where the structures of RNAs contain psudoknots.

From the molecular biological or biomedical viewpoint, the stems and loops in an RNA structure may possess some functions or induce the interactions with other molecules, proteins and complexes. The stems and loops should be important characteristics for RNA structures. If we regard stems or loops as blocks, an RNA sequence can be represented as a sequence with stem blocks and loop blocks. The block-representing RNA sequences lead to a novel RNA structure similarity and alignment problem. In this paper, we propose a dynamic programming algorithm for calculating the similarity and alignment of two block-representing RNAs with stems and loops.

The rest of this paper is organized as follows. In Section 2, we introduce the RNA structures and related definitions. Our RNA secondary structure alignment algorithm based on stem representation is described in Section 3. We improve our algorithm to reduce the number of required indices in Section 4. Section 5 shows our experimental results and we conclude this paper in Section 6.

2 Preliminaries

RNA is a single strand, composed of nucleotides (bases) adenine (A), guanine (G), cytosine (C) and uracil (U). The sequence consisting of the nucleotides A, G, C and U is called the *primary structure* of RNA. The *secondary structure* of an RNA is formed by its primary structure which is folded back on itself. In a secondary structure of an RNA, there are three possible types of base pairs: G≡C, A=U, and G−U with triple, double and single hydrogen bond, respectively.

An RNA sequence is represented as a string of N_R elements $R = r_1 r_2 \cdots r_i \cdots r_{N_R}$, where $r_i \in \{\text{A, G, C, U}\}$, $1 \le i \le N_R$. (r_i, r_j) denotes a base pair of a secondary structure of R, where $1 \le i < j \le N_R$. For convenience, given two nucleotides x and y in an RNA sequence, if x is closer to $5'$ and y is closer to $3'$, we call x is in the *left position* of y and y is in the *right position* of x.

We can cut an RNA secondary structure into a linear structure that contains the information of the secondary structure. In the linear structure, there are many blocks according to the information of the secondary structure. A *block* is defined to be a contiguous sequence (or string) that is composed of several nucleotides. Hence, the *block string* of N_B elements is denoted as $B = b_1 b_2 \cdots b_i \cdots b_{N_B}$, where b_i is a block, $1 \le i \le N_B$.

We define a *stem* as a group of stacked pairs, and the stem in an RNA secondary structure is contiguous among these stacked pairs. In a stem, the stacked pairs can be divided into two *stem blocks* according to their indices. The *left stem block* is closer to $5'$, and the *right stem block* is referred to as the block closer to $3'$. We use $< b_i, b_j >$ to denote the two stem blocks in the same stem, where b_i and b_j are the left and right stem blocks, respectively. Another group of contiguous unpaired nucleotides is called *loop block*. The block string B of an RNA secondary structure consists of loop blocks and stem blocks.

The following example illustrates an RNA structure.

$$R = \mathtt{A_1G_2G_3C_4C_5A_6A_7U_8U_9G_{10}C_{11}U_{12}G_{13}C_{14}A_{15}}\\ \mathtt{A_{16}G_{17}G_{18}C_{19}C_{20}U_{21}G_{22}C_{23}C_{24}C_{25}A_{26}C_{27}G_{28}}$$

As shown in Figure 1, $(\mathtt{G_2}, \mathtt{U_9})$ and $(\mathtt{G_3}, \mathtt{U_8})$ are base pairs in R, where base pairs $\mathtt{G_2}$ and $\mathtt{G_3}$ are contiguous, $\mathtt{U_8}$ and $\mathtt{U_9}$ are also contiguous. The set of stacked pairs $(\mathtt{G_2}, \mathtt{U_9})$ and $(\mathtt{G_3}, \mathtt{U_8})$ form one stem. The left stem block is composed of $\mathtt{G_2}$ and

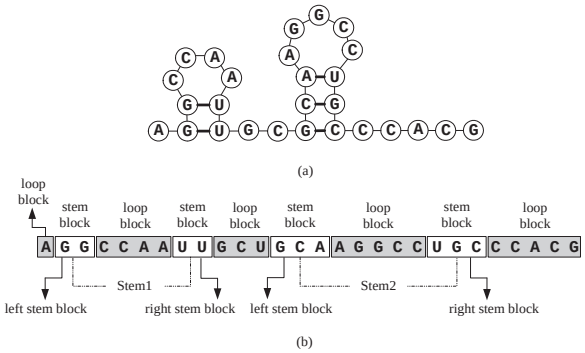

Figure 1: The secondary structure and the blocks of RNA sequence R = AGGCCAAUUGCUGCAAGGCCUGCCCACG. (a) The secondary structure of R. (b) All loop blocks and all stem blocks of R, where a stem has left stem block and right stem block.

G_3. The right stem block is composed U_8 and U_9. Similarly, the base pairs $(G_{13}, C_{23}), (C_{14}, G_{22})$ and (A_{15}, U_{21}) form another stem, where the left stem block is composed of G_{13}, C_{14} and A_{15}, and the right stem block is composed U_{21}, G_{22} and C_{23}. In addition, C_4, C_5, A_6 and A_7 are contiguous unpaired nucleotides. In fact, $C_4C_5A_6A_7$ is one of loop blocks in R.

The RNA secondary structure has two types named *non-pseudoknot* and *pseudoknot*. The *non-pseudoknot* type has no cross for each stem of an RNA secondary structure, the *pseudoknot* type may have crossing stems in an RNA secondary structure. For example, consider a string block $B = b_1b_2b_3b_4b_5b_6$, where b_2 and b_4 are loop blocks and other blocks are stem blocks. If $< b_1, b_6 >$ and $< b_3, b_5 >$ are two stems, we say B belongs to the non-pseudoknot type. If $< b_1, b_5 >$ and $< b_3, b_6 >$ are two stems, then they form a cross and the RNA B has a pseudoknot. Note that our algorithm only considers the RNA structures without any pseudoknot.

3 RNA Secondary Structure Alignment Algorithm

In this section, we shall propose a new dynamic programming method for performing an alignment on two RNA secondary structures. For convenience, we regard the two RNA sequences as the master sequence and the slave sequence. The *master* sequence is of length N_R, denoted as $R = r_1r_2 \cdots r_{N_R}$, and the *slave* sequence is of length N_{R^*}, denoted as $R^* = r_1^*r_2^* \cdots r_{N_{R^*}}^*$. If there are N_B blocks contained in the secondary structure of R, the N_B blocks form a block string $B = b_1b_2 \cdots b_{N_B}$. Similarly, $B^* = b_1^*b_2^* \cdots b_{N_{B^*}}^*$, where R^* contains N_{B^*} blocks.

An RNA secondary structure contains a set of stems, each consisting of two stem blocks. Suppose there are N_S and N_{S^*} stem blocks in the secondary structures of R and R^*, respectively. Let S denote a stem block string and $S = s_1s_2 \cdots s_i \cdots s_{N_S}$, where s_i is a stem block, $1 \le i \le N_S$. Let S^* denote another stem block string and $S^* = s_1^*s_2^* \cdots s_j^* \cdots s_{N_{S^*}}^*$, where s_j^* is a stem block, $1 \le j \le N_{S^*}$. Let $p(s_i)$ denote the *stem partner* of stem block s_i. That is, $< s_i, p(s_i) >$ and $< p(s_i), s_i >$ are stems. We define a *segment* to be a group of contiguous nucleotides which appear within a block or across several blocks. The symbol $[x_s, x_e]$ represents the segment starting at nucleotide x_s and ending at nucleotide x_e. Similarly, the symbols with subscripts s and e represent the starting and ending positions, respectively.

We define a function $\rho(s_i, s_j)$ to decide whether two stem blocks s_i and s_j form a stem or not. If two stem blocks s_i and s_j form a stem, then $\rho(s_i, s_j) = 1$; otherwise, $\rho(s_i, s_j) = 0$. In addition, we denote the complement of $\rho(s_i, s_j)$ as $\overline{\rho(s_i, s_j)}$. For convenience, we regard $\rho(s_i, s_j)$ and $\overline{\rho(s_i, s_j)}$ as Boolean functions in our formulas. Let $stemalign(< s_i, s_j >, < s_k^*, s_l^* >)$ denote the alignment score function between stem $< s_i, s_j >$ of R and stem $< s_k^*, s_l^* >$ of R^*, where $1 \le i < j \le N_S$ and $1 \le k < l \le N_{S^*}$. Let $segmentalign(t, z)$ denote the segment alignment score function, where t is a segment of R and z is a segment of R^*. The operator $\oplus$ represents to find the next stem block and $\ominus$ represents to find the previous stem block.

Finally, let $f(x, y, u, v, \alpha, \beta, \gamma, \delta)$ define the alignment score function in our algorithm, where x and y are the stem blocks of R, u and v are the stem blocks of R^*, α and β are segments of R, and γ and δ are segments of R^*. Therefore, the initial indices of the alignment score function $f(x, y, u, v, \alpha, \beta, \gamma, \delta)$ of our algorithm are expressed in Figure 2.

Hence, the initialization of our algorithm is summarized as follows.

1. $x = s_1$ and $y = s_{N_S}$ in R.

2. $u = s_1^*$ and $v = s_{N_{S^*}}^*$ in R^*.

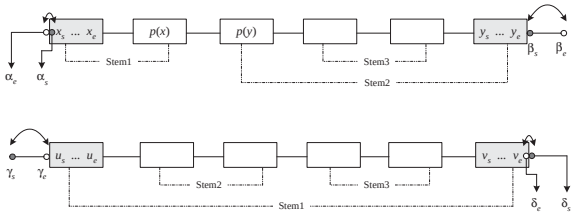

Figure 2: The illustration of initial indices of our algorithm.

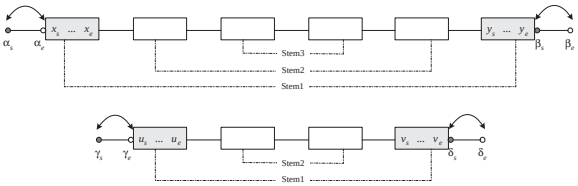

Figure 3: Illustration of Case 1.

3. In R, if α is not null (empty), then $\alpha = b_1$ according to B, $\alpha_s = (b_1)_s$ and $\alpha_e = (b_1)_e$, i.e., $[\alpha_s, \alpha_e] = [(b_1)_s, (b_1)_e]$. Otherwise, α is null, $\alpha_s = x_s$ and $\alpha_e = x_s - 1$, i.e., $[\alpha_s, \alpha_e] = [x_s, x_s - 1]$. It is similar for γ in R^*. If γ is not null, then $[\gamma_s, \gamma_e] = [(b_1^*)_s, (b_1^*)_e]$. Otherwise, γ is null, $[\gamma_s, \gamma_e] = [u_s, u_s - 1]$.

4. In R, if β is not null, then $\beta = b_{N_B}$ according to B, $\beta_s = (b_{N_B})_s$ and $\beta_e = (b_{N_B})_e$, i.e., $[\beta_s, \beta_e] = [(b_{N_B})_s, (b_{N_B})_e]$. Otherwise, β is null, $\beta_s = y_e + 1$ and $\beta_e = y_e$, i.e., $[\beta_s, \beta_e] = [y_e + 1, y_e]$. It is similar for δ in R^*. If δ is not null, then $[\delta_s, \delta_e] = [(b^*_{N_{B^*}})_s, (b^*_{N_{B^*}})_e]$. Otherwise, δ is null, $[\delta_s, \delta_e] = [v_e + 1, v_e]$.

In our algorithm, the input data are two RNA secondary structures without any pseudoknot. In other words, their secondary structures look like $\cdot(\cdot((\cdot(\cdot))\cdot)\cdot)$, where $''()''$ represents a stem and $''\cdot''$ represents a loop block. We divide our alignment algorithm into four cases according to the situations of stem blocks as follows.

Case 1: $\rho(x, y) = 1$ and $\rho(u, v) = 1$. Figure 3 illustrates Case 1. We can further divide the case into three subcases.

Case 1.1: Stem $< x, y >$ aligns with stem $< u, v >$. We must find a solution for stem blocks $x \oplus 1, y \ominus 1, u \oplus 1$ and $v \ominus 1$. Thus, $f(x, y, u, v, \alpha, \beta, \gamma, \delta) = f(x \oplus 1, y \ominus 1, u \oplus 1, v \ominus 1, [x_e + 1, (x \oplus 1)_s - 1], [(y \ominus 1)_e + 1, y_s - 1], [u_e + 1, (u \oplus 1)_s - 1], [(v \ominus 1)_e + 1, v_s - 1]) + stemalign(< x, y >, < u, v >) + segmentalign(\alpha, \gamma) + segmentalign(\beta, \delta)$.

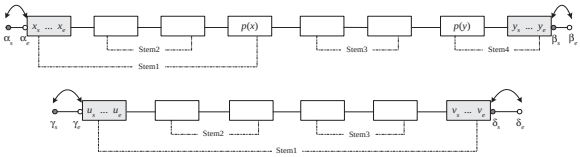

Figure 4: Illustration of Case 2.

Case 1.2: $< x, y >$ is not aligned with $< u, v >$, and $< x, y >$ of R becomes a part of a segment. We must find a solution for stem blocks $x \oplus 1, y \ominus 1, u$ and v. Thus, $f(x, y, u, v, \alpha, \beta, \gamma, \delta) = f(x \oplus 1, y \ominus 1, u, v, [\alpha_s, (x \oplus 1)_s - 1], [(y \ominus 1)_e + 1, \beta_e], \gamma, \delta)$.

Case 1.3: $< x, y >$ is not aligned with $< u, v >$, and $< u, v >$ of R^* becomes a part of a segment. We must find a solution for stem blocks $x, y, u \oplus 1$ and $v \ominus 1$. Thus, $f(x, y, u, v, \alpha, \beta, \gamma, \delta) = f(x, y, u \oplus 1, v \ominus 1, \alpha, \beta, [\gamma_s, (u \oplus 1)_s - 1], [(v \ominus 1)_e + 1, \delta_e])$.

Case 2: $\rho(x, y) = 0$ and $\rho(u, v) = 1$. Figure 4 illustrates Case 2. This case can be further divided into three subcases as follows.

Case 2.1: Find the partner $p(y)$ of stem block y. We must find a solution for stem blocks $x, p(y) \ominus 1, u$ and v. Thus, $f(x, y, u, v, \alpha, \beta, \gamma, \delta) = f(x, p(y) \ominus 1, u, v, \alpha, [(p(y) \ominus 1)_e + 1, \beta_e], \gamma, \delta)$.

Case 2.2: Find $p(x)$ of x. We must find a solution for stem blocks $p(x) \oplus 1, y, u$ and v. Thus, $f(x, y, u, v, \alpha, \beta, \gamma, \delta) = f(p(x) \oplus 1, y, u, v, [\alpha_s, (p(x) \oplus 1)_s - 1], \beta, \gamma, \delta)$.

Case 2.3: Stem $< u, v >$ of R^* is a part of a segment. We must find a solution for stem blocks $x, y, u \oplus 1$ and $v \ominus 1$. Thus, $f(x, y, u, v, \alpha, \beta, \gamma, \delta) = f(x, y, u \oplus 1, v \ominus 1, \alpha, \beta, [\gamma_s, (u \oplus 1)_s - 1], [(v \ominus 1)_e + 1, \delta_e])$.

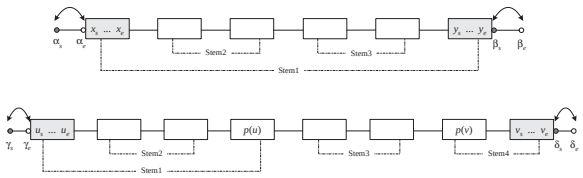

Figure 5: Illustration of Case 3.

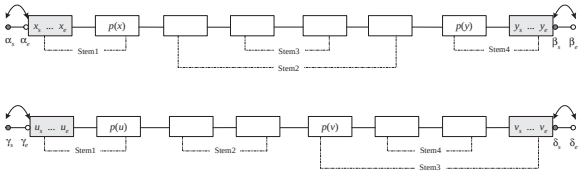

Figure 6: Illustration of Case 4.

Case 3: $\rho(x,y) = 1$ and $\rho(u,v) = 0$. Figure 5 illustrates Case 3. We can further divide the case into three subcases.

Case 3.1: Find $p(v)$ of v. We must find a solution for stem blocks x, y, u and $p(v) \ominus 1$. Thus, $f(x,y,u,v,\alpha,\beta,\gamma,\delta) = f(x,y,u,p(v) \ominus 1, \alpha, \beta, \gamma, [(p(v) \ominus 1)_e + 1, \delta_e])$.

Case 3.2: Find $p(u)$ of u. We must find a solution for stem blocks $x, y, p(u) \oplus 1$ and v. Thus, $f(x,y,u,v,\alpha,\beta,\gamma,\delta) = f(x,y,p(u) \oplus 1, v, \alpha, \beta, [\gamma_s, (p(u) \oplus 1)_s - 1], \delta)$.

Case 3.3: Stem $< x, y >$ of R is a part of a segment. We must find a solution for stem blocks $x \oplus 1, y \ominus 1, u$ and v. Thus, $f(x,y,u,v,\alpha,\beta,\gamma,\delta) = f(x \oplus 1, y \ominus 1, u, v, [\alpha_s, (x \oplus 1)_s - 1], [(y \ominus 1)_e + 1, \beta_e], \gamma, \delta)$.

Case 4: $\rho(x,y) = 0$ and $\rho(u,v) = 0$. Figure 6 illustrates Case 4. We can further divide the case into six subcases as follows.

Case 4.1: Find $p(y)$ of y. We must find a solution for stem blocks $x, p(y) \ominus 1, u$ and v. Thus, $f(x,y,u,v,\alpha,\beta,\gamma,\delta) = f(x, p(y) \ominus 1, u, v, \alpha, [(p(y) \ominus 1)_e + 1, \beta_e], \gamma, \delta)$.

Case 4.2: Find $p(v)$ of v. We must find a solution for stem blocks x, y, u and $p(v) \ominus 1$. Thus, $f(x,y,u,v,\alpha,\beta,\gamma,\delta) = f(x,y,u,p(v) \ominus 1, \alpha, \beta, \gamma, [(p(v) \ominus 1)_e + 1, \delta_e])$.

Case 4.3: Find $p(y)$ of y and $p(v)$ of v. Thus, we divide this subcase into two mapping groups. The first group is for stem blocks $x, p(y) \ominus 1, u$ and $p(v) \ominus 1$. The second group is for stem blocks $p(y), y, p(v)$ and v. We must find the solution for both groups. Thus, $f(x,y,u,v,\alpha,\beta,\gamma,\delta) = f(x, p(y) \ominus 1, u, p(v) \ominus 1, \alpha, [(p(y) \ominus 1)_e + 1, p(y)_s - 1], \gamma, [(p(v) \ominus 1)_e + 1, p(v)_s - 1]) + f(p(y), y, p(v), v, [p(y)_s, p(y)_s - 1], \beta, [p(v)_s, p(v)_s - 1], \delta)$.

Case 4.4: Find $p(x)$ of x. We must find a solution for stem blocks $p(x) \oplus 1, y, u$ and v. Thus, $f(x,y,u,v,\alpha,\beta,\gamma,\delta) = f(p(x) \oplus 1, y, u, v, [\alpha_s, (p(x) \oplus 1)_s - 1], \beta, \gamma, \delta)$.

Case 4.5: Find $p(u)$ of u. We must find a solution for stem blocks $x, y, p(u) \oplus 1$ and v. Thus, $f(x,y,u,v,\alpha,\beta,\gamma,\delta) = f(x,y,p(u) \oplus 1, v, \alpha, \beta, [\gamma_s, (p(u) \oplus 1)_s - 1], \delta)$.

Case 4.6: Find $p(x)$ of x and $p(u)$ of u. Thus, we divide the subcase into two mapping groups. The first group is for stem blocks $x, p(x), u$ and $p(u)$. The second group is for stem blocks $p(x) \oplus 1, y, p(u) \oplus 1$ and v. We must find a solution for the two groups. Thus, $f(x,y,u,v,\alpha,\beta,\gamma,\delta) = f(x, p(x), u, p(u), \alpha, [p(x)_e + 1, p(x)_e], \gamma, [p(u)_e + 1, p(u)_e]) + f(p(x) \oplus 1, y, p(u) \oplus 1, v, [p(x)_e + 1, (p(x) \oplus 1)_s - 1], \beta, [p(u)_e + 1, (p(u) \oplus 1)_s - 1], \delta)$.

4 Improvement on Space

The function $f(x,y,u,v,\alpha,\beta,\gamma,\delta)$ has an 8-dimensional argument, and the maximum range of each argument in one dimension is the maximum number of blocks in the block strings (i.e., $max(N_B, N_{B^*})$). One can easily observe that both time complexity and space complexity of our algorithm are $O((max(N_B, N_{B^*}))^8)$.

For practical implementation in computers, the formula is infeasible when the number of RNA blocks is large. Accordingly, we have to try to reduce the required memory space. That is, we

will try to decrease the dimensions (i.e., the number of indices) of the dynamic programming formula $f(x, y, u, v, \alpha, \beta, \gamma, \delta)$. It will lead to the design of a new dynamic programming formula $\widehat{f}(i, j, k, l)$ whose function is the same as that of $f(x, y, u, v, \alpha, \beta, \gamma, \delta)$.

Before introducing formula $\widehat{f}(i, j, k, l)$, we need modify the symbols, terms and functions defined in formula $f(x, y, u, v, \alpha, \beta, \gamma, \delta)$, and define some more. Let $I(x, S, B)$ denote the block in the block string B which is corresponding to the stem block x of stem block string S, where $1 \leq x \leq N_S$ and $1 \leq I(x, S, B) \leq N_B$. In other words, if $s_x = b_i$, then $I(x, S, B) = i$. Let $I(x, B, S)$ denote the stem block of S corresponding to block x of B, but $I(x, B, S) = \texttt{null}$ if we can not find the corresponding stem block. Let $L(x, i, j)$ denote the maximal block between blocks i and x, where $I(L(x, i, j), B, S) \neq \texttt{null}$ and $i \leq p(L(x, i, j)) \leq j$. In other words, $L(x, i, j) = \max_{i \leq y \leq x}\{y | y$ is a block of B, $I(y, B, S) \neq \texttt{null}$ and $i \leq p(y) \leq j\}$. Similarly, $R(x, i, j) = \min_{x \leq y \leq j}\{y | y$ is a block of B, $I(y, B, S) \neq \texttt{null}$ and $i \leq p(y) \leq j$ $\}$. In addition, we define $LN(x, i, j) = \max_{i \leq y \leq x}\{y | y$ is a block of B, $I(y, B, S) \neq \texttt{null}$ and $i \leq p(y) \leq j\}$. Similarly, $R(x, i, j) = \min_{x \leq y \leq j}\{y | y$ is a block of B, $I(y, B, S) \neq \texttt{null}$ and $i \leq p(y) \leq j$ $\}$. $LN(x, i, j)$ and $RN(x, i, j)$ are used for finding the next stems. The above functions for S^* and B^* are defined similarly.

In the new dynamic programming algorithm, the initial values of the variables in $\widehat{f}(i, j, k, l)$ are set that i and j are the block indices in B, and $i = 1$ and $j = N_B$ in R; k and l are the block indices in B^*, and $k = 1$ and $l = N_B^*$ in R^*. Then formula $\widehat{f}(i, j, k, l)$ is given in Figure 7.

$segmentalign(i, j)$ represents the segment alignment score function, where i is a segment of R and j is a segment of R^*. The calculation of $segmentalign(i, j)$ is to find the minimal length between segments i and j. Indeed, one can easily modify the function $segmentalign(i, j)$ to align segments i and j. We do not consider whether the aligned nucleotides are the same or not, because the RNA secondary structure alignment appreciates the alignment of the structural shape. We shall consider only the segment length here. Hence, $segmentalign(i, j)$ is given as follows.

$$segmentalign(i, j) = min(n_R(i), n_{R^*}(j)) \times \phi, \text{ where } \phi \text{ is the score for aligning two nucleotides and it is a constant, and } n_R(i) \text{ and } n_{R^*}(j) \text{ are the lengths of segments } i \text{ and } j, \text{ respectively.} \quad (8)$$

For example, consider two segments $i =$ CGAAGUC and $j =$ AAUGAGCUG, where segments i and j have 7 and 9 nucleotides, respectively. If $\phi = 1$ in Equation 8, then $segmentalign(i, j) = min(7, 9) \times 1 = 7$. When $\phi = 0$, our algorithm only aligns the stem part of RNA secondary structures. Here, we will set ϕ to 1 when discussing the performance of our algorithm.

In the stem alignment problem, we consider only base pairs here. That is, we only align the base pairs in both stems. Let x be one stem of the secondary structure of R and $n_{bp}(x)$ be the number of base pairs in x. Similarly, let y be one stem of the secondary structure of R^* and $n_{bp^*}(y)$ be the number of base pairs in y. Hence, $stemalign(x, y)$ is given as follows.

$$stemalign(x, y) = (min(n_{bp}(x), n_{bp^*}(y)))^{\lambda} \times \tau, \text{ where } \lambda \text{ and } \tau \text{ are constants.} \quad (9)$$

For example, suppose two stems $x =$ AAG $\|$ CUU and $y =$ AGGCC $\|$ GGCCU, stem x has 3 base pairs and stem y has 5 base pairs. If $\lambda = 2$ and $\tau = 4$ in Equation 9, then $stemalign(x, y) = (min(3, 5))^2 \times 4 = 36$.

We can see that there are four indices in the improved formula $\widehat{f}(i, j, k, l)$, and the number of all cases are fixed. Therefore, it is trivial that the time complexity of our improved dynamic programming algorithm is $O(N^4)$, where N is the number of blocks contained in the given RNAs (i.e. $N = max(N_B, N_{B^*})$).

5 Experimental Results

In this section, we will show our experimental results and analyze the performance of our algorithm. Our algorithm is implemented by C++ using Borland C++ Compiler 5.5 on PC with AMD Duron^{TM} processor 800 MHz and 256 MB RAM. Our testing data are obtained from NCBI (http://www.ncbi.nlm.nih.gov/). There are 22 tRNAs in Homo sapiens mitochondrion. In our experiments, we use the method in [7] to predict the secondary structures of tRNAs.

$$\widehat{f}_1(i,j,k,l) = \max_{\substack{RN(i,i,j) \le x \le LN(j,i,j) \\ RN(k,k,l) \le y \le LN(l,k,l)}} \left\{ \widehat{f}(i,x-1,k,y-1) + \widehat{f}(x,p(x),y,p(y)) + \widehat{f}(p(x)+1,j,p(y)+1,l) \right\} \quad (1)$$

$$\widehat{f}_2(i,j,k,l) = \begin{array}{l} segmentalign([i_s,(R(i,i,j)-1)_e],[k_s,(R(k,k,l)-1)_e]) \\ \quad +segmentalign([(L(j,i,j)+1)_s,j_e],[(L(l,k,l)+1)_s,l_e]) \\ \quad +stemalign(< R(i,i,j),L(j,i,j) >,< R(k,k,l),L(l,k,l) >) \\ \quad +\widehat{f}(R(i,i,j)+1,L(j,i,j)-1,R(k,k,l)+1,L(l,k,l)-1) \end{array} \quad (2)$$

$$\widehat{f}_3(i,j,k,l) = \max \begin{cases} \widehat{f}(i,R(i,i,j)-1,k,R(k,k,l)-1) + \widehat{f}(R(i,i,j),LN(p(L(j,i,j)),i,j),R(k,k,l),L(l,k,l)) \\ \quad +\widehat{f}(LN(p(L(j,i,j)),i,j)+1,j,L(l,k,l)+1,l) \\ \widehat{f}(i,RN(p(R(i,i,j)),i,j)-1,k,R(k,k,l)-1) + \widehat{f}(RN(p(R(i,i,j)),i,j),L(j,i,j),R(k,k,l),L(l,k,l)) \\ \quad +\widehat{f}(L(j,i,j)+1,j,L(l,k,l)+1,l) \\ \widehat{f}(i,R(i,i,j)-1,k,RN(R(k,k,l),k,l)-1) + \widehat{f}(R(i,i,j),L(j,i,j),RN(R(k,k,l),k,l),LN(L(l,k,l),k,l)) \\ \quad +\widehat{f}(L(j,i,j)+1,j,LN(L(l,k,l),k,l)+1,l) \end{cases} \quad (3)$$

$$\widehat{f}_4(i,j,k,l) = \max \begin{cases} \widehat{f}(i,R(i,i,j)-1,k,R(k,k,l)-1) + \widehat{f}(R(i,i,j),L(j,i,j),R(k,k,l),LN(p(L(l,k,l)),k,l)) \\ \quad +\widehat{f}(L(j,i,j)+1,j,LN(p(L(l,k,l)),k,l)+1,l) \\ \widehat{f}(i,R(i,i,j)-1,k,RN(p(R(k,k,l)),k,l)-1) + \widehat{f}(R(i,i,j),L(j,i,j),RN(p(R(k,k,l)),k,l),L(l,k,l)) \\ \quad +\widehat{f}(L(j,i,j)+1,j,L(l,k,l)+1,l) \\ \widehat{f}(i,RN(R(i,i,j),i,j)-1,k,R(k,k,l)-1) + \widehat{f}(RN(R(i,i,j),i,j),LN(L(j,i,j),i,j),R(k,k,l),L(l,k,l)) \\ \quad +\widehat{f}(LN(L(j,i,j),i,j)+1,j,L(l,k,l)+1,l) \end{cases} \quad (4)$$

$$\widehat{f}_5(i,j,k,l) = \max \begin{cases} \widehat{f}(i,R(i,i,j)-1,k,R(k,k,l)-1) + \widehat{f}(R(i,i,j),LN(p(L(j,i,j)),i,j),R(k,k,l),L(l,k,l)) \\ \quad +\widehat{f}(LN(p(L(j,i,j)),i,j)+1,j,L(l,k,l)+1,l) \\ \widehat{f}(i,R(i,i,j)-1,k,R(k,k,l)-1) + \widehat{f}(R(i,i,j),L(j,i,j),R(k,k,l),LN(p(L(l,k,l)),k,l)) \\ \quad +\widehat{f}(L(j,i,j)+1,j,LN(p(L(l,k,l)),k,l)+1,l) \\ \widehat{f}(i,p(L(j,i,j))-1,k,p(L(l,k,l))-1) + \widehat{f}(p(L(j,i,j)),j,p(L(l,k,l)),l) \\ \widehat{f}(i,RN(p(R(i,i,j)),i,j)-1,k,R(k,k,l)-1) + \widehat{f}(RN(p(R(i,i,j)),i,j),L(j,i,j),R(k,k,l),L(l,k,l)) \\ \quad +\widehat{f}(L(j,i,j)+1,j,L(l,k,l)+1,l) \\ \widehat{f}(i,R(i,i,j)-1,k,RN(p(R(k,k,l)),k,l)-1) + \widehat{f}(R(i,i,j),L(j,i,j),RN(p(R(k,k,l)),k,l),L(l,k,l)) \\ \quad +\widehat{f}(L(j,i,j)+1,j,L(l,k,l)+1,l) \\ \widehat{f}(i,p(R(i,i,j)),k,p(R(k,k,l))) + \widehat{f}(p(R(i,i,j))+1,j,p(R(k,k,l))+1,l) \end{cases} \quad (5)$$

$$\widehat{f}_6(i,j,k,l) = \begin{cases} segmentalign([i_s,j_e],[k_s,l_e]) & \text{if } i=j \text{ and } k \le l \quad \text{or } i \le j \text{ and } k=l \\ 0 & \text{if } i>j \text{ or } k>l \end{cases} \quad (6)$$

$$\widehat{f}(i,j,k,l) = \begin{cases} \max \begin{cases} \widehat{f}_1(i,j,k,l) \\ (\rho(i,j)\times\rho(k,l)+\rho(R(i,i,j),L(j,i,j)) \\ \quad \times\rho(R(k,k,l),L(l,k,l)))\times\widehat{f}_2(i,j,k,l) \\ (\overline{\rho(i,j)}\times\rho(k,l)+\overline{\rho(R(i,i,j),L(j,i,j))} \\ \quad \times\rho(R(k,k,l),L(l,k,l)))\times\widehat{f}_3(i,j,k,l) \\ (\rho(i,j)\times\overline{\rho(k,l)}+\rho(R(i,i,j),L(j,i,j)) \\ \quad \times\overline{\rho(R(k,k,l),L(l,k,l))})\times\widehat{f}_4(i,j,k,l) \\ (\overline{\rho(i,j)}\times\overline{\rho(k,l)}+\overline{\rho(R(i,i,j),L(j,i,j))} \\ \quad \times\overline{\rho(R(k,k,l),L(l,k,l))})\times\widehat{f}_5(i,j,k,l) \\ segmentalign([i_s,j_e],[k_s,l_e]) \end{cases} & \text{if } i<j \text{ and } k<l \\ \widehat{f}_6(i,j,k,l) & \text{otherwise} \end{cases} \quad (7)$$

Figure 7: The new dynamic programming formula $\widehat{f}(i,j,k,l)$.

Some parameters are used to measure the LCS (longest common subsequence) identity of the two sequences. $Min(L)$ denotes the alignment length of the minimal LCS alignment. The LCS identity is defined as follows.

$$LCS\ identity = \frac{\text{LCS length}}{Min(L)} \times 100\%$$

Then, some parameters are used to measure the *structure similarity* of the secondary structures of R and R^*. $Score(R, R^*)$ denotes the score of the secondary structure alignment of R and R^* by using our algorithm. $Max(R, R^*)$ denotes the maximal score between the two scores which R and R^* are aligned with themselves, respectively. That is, $Max(R, R^*) = Max\{Score(R, R), Score(R^*, R^*)\}$. Then the structure similarity is defined as follows.

$$structure\ similarity = \frac{Score(R, R^*)}{Max(R, R^*)} \times 100\%$$

We shall change values of some parameters in Equation 9. λ advances (makes) that the alignment score of base pairs between two stems is nonlinear enhancement and τ denotes the stem score contributed by aligning two base pairs in two stems. We test structure similarities of each pair of 22 tRNAs by using our algorithm, where $\lambda = 1, 1.5$ and 2, and $\tau = 2, 3, 4, 5$ and 6. Consequently, we have come to the conclusion that λ can be set to 1.5 and τ can be set to 4.

Though many RNA sequences share very high identity, their secondary structures or functions may not be very similar. Meantime, the secondary structures and functions of many RNA sequences are very similar, but they may not share high sequence identity. Hence, we will discuss the performance of our algorithm according to the similarities of sequences and structures.

Table 1 shows the result based on $\lambda = 1.5$ and $\tau = 4$, where the input data is TRNG and TRNA for Case A, and TRNG and TRNV for Case B. They are tRNAs of Homo sapiens mitochondrion. In Table 1, we can see that LCS identity is 47.62 % and 54.43 % of Case A and Case B, respectively. That is, the LCS identity of Case B is higher than Case A.

The RNA secondary structures of Case A are predicted by using a dynamic programming method [7]. Figure 8 and Figure 9 show the secondary structures of Case A and Case B, respectively. In Figure 8 and Figure 9, we can see that the two secondary structures of Case A are more similar than those of Case B.

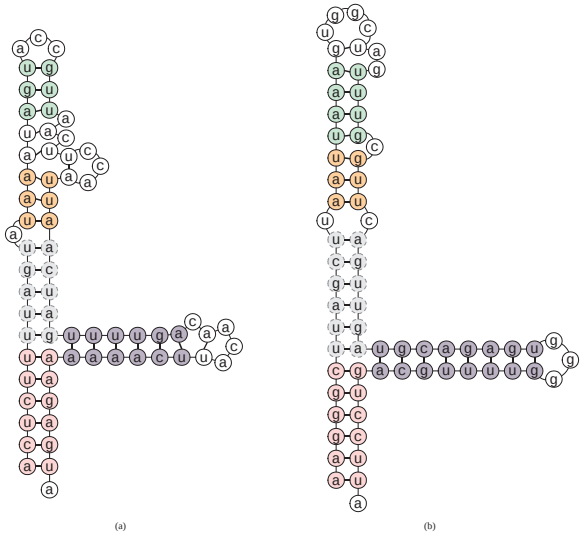

Figure 8: The optimal alignment for Case A by using our algorithm and $\lambda = 1.5$ and $\tau = 4$. (a) The secondary structure of TRNG. (b) The secondary structure of TRNA. Note: The secondary structures are predicted by using a dynamic programming method [7].

In Table 1, we can see that the structure similarity of Case A is 76.21 % and structure similarity of Case B is 50.44 %. In other words, the structure similarity of Case A is higher than Case B by applying our algorithm, but the LCS identity of Case B is higher than Case A. Hence, our algorithm is a good approach for RNA secondary structure alignment.

6 Conclusion

In this paper, we propose an RNA secondary structure alignment algorithm to evaluate the secondary structure similarity of two RNAs based on blocks for dealing with stems and loops. For the practical implementation in computers, the eight indices of our original formula make the program infeasible when the number of RNA blocks is large. We improve the formula into a four-index one. We can see that there are four indices in the improved formula, both time complexity and space complexity of our improved dynamic programming algorithm are reduced to $O(N^4)$, where N is the number of blocks contained in the given RNAs.

For the experiment of the biological data, we

Table 1: The experimental results of Case A and Case B based on $\lambda = 1.5$ and $\tau = 4$. Case A: TRNG and TRNA. Case B: TRNG and TRNV.

λ	τ	Case	LCS length	$Min(L)$	LCS identity	$Score(R, R^*)$	$Max(R, R^*)$	Structure similarity
1.5	4	A	40	84	47.62 %	205	269	76.21 %
		B	43	79	54.43 %	114	226	50.44 %

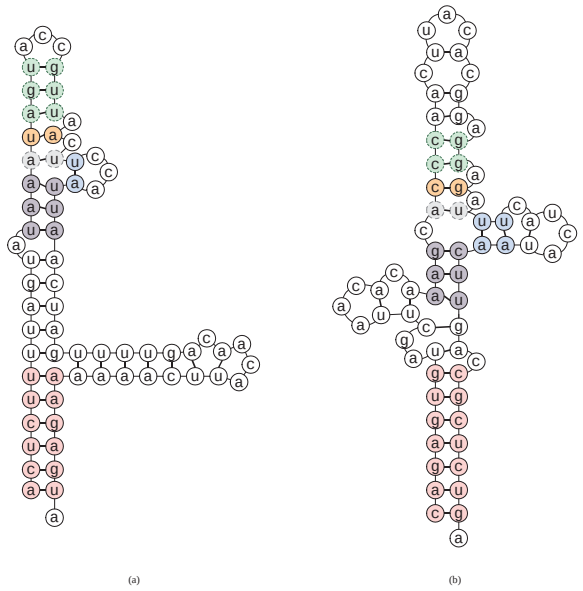

Figure 9: The optimal alignment for Case B by using our algorithm and $\lambda = 1.5$ and $\tau = 4$. (a) The secondary structure of TRNG. (b) The secondary structure of TRNV. Note: The secondary structures are predicted by using a dynamic programming method [7].

take three of tRNAs, that are TRNG, TRNA, and TRNV, as the input data of our algorithm. In the result, we see that the structure similarity of TRNG and TRNA is higher, but their LCS identity is lower than TRNG and TRNV. Hence, our block concept and our algorithm should be a good approach and method for RNA secondary structure alignment. In addition, it is also helpful to predict the functions of biomolecules and to classify them. In the future, we will test more real data to verify the practicality of our algorithm. The psudoknots are not considered in our algorithm, it is a worth way to develop new RNA structure alignment method to deal with psudoknots.

References

[1] V. Bafna, S. Muthukrishnan, and R. Ravi, "Computing similarity between RNA strings," *Proceedings of the 6th Annual Symposium on Combinatorial Pattern Matching*, pp. 1–16, 1995.

[2] G. D. Collins, S. Le, and K. Zhang, "A new method for computing similarity between RNA structures," *Information Sciences*, Vol. 139, pp. 59–77, 2001.

[3] F. Corpet and B. Michot., "RNAlign program: Alignment of RNA sequences using both primary and secondary structures," *Computer Applications in the Biosciences*, Vol. 10, No. 4, pp. 389–399, 1994.

[4] T. Jiang, L. Wang, and K. Zhang, "Alignment of trees - an alternative to tree edit," *Theoretical Computer Science*, Vol. 143, No. 1, pp. 137–148, 1995.

[5] S. Y. Le, R. Nussinov, and J. V. Maizel, "Tree graphs of RNA secondary structures and their comparisons," *Computers and Biomedical Research*, Vol. 22, pp. 461–473, 1989.

[6] S. Y. Le, J. Owens, R. Nussinov, J. H. Chen, B. A. Shapiro, and J. V. Maizel, "RNA secondary structures: Comparison and determination of frequently recurring substructures by consensus," *Computer Applications in the Biosciences*, Vol. 5, No. 3, pp. 205–210, 1989.

[7] R. C. T. Lee, "Computational biology." http://www.csie.ncnu.edu.tw/, Department of Computer Science and Information Engineering, National Chi-Nan University, 2001.

[8] H. P. Lenhof, K. Reinert, and M. Vingron, "A polyhedral approach to RNA sequence structure alignment," *Journal of Computational Biology*, Vol. 5, No. 3, pp. 517–530, 1998.

[9] G. H. Lin, B. Ma, and K. Zhang, "Edit distance between two RNA structures," *Proceedings of the fifth Annual International Conference on Computational Biology*, pp. 211–220, 2001.

[10] M. C. Lin, C. B. Yang, and K. S. Huang, "Prediction of RNA secondary structures by genetic algorithms," *Proceedings of the 6th World Multiconference on Systemics, Cybernetics and Informatics, SCI 2002*, Vol. 12, pp. 439–444, 2002.

[11] C. L. Lu, Z. Y. Su, and C. Y. Tang, "A new measure of edit distance between labeled trees," *Proceedings of the 7th Annual International Computing and Combinatorics Conference (COCOON 2001)*, Vol. 2108, pp. 338–348, 2001.

[12] B. Ma, L. Wang, and K. Zhang, "Computing similarity between RNA structures," *Theoretical Computer Science*, Vol. 276, pp. 111–132, 2002.

[13] C. Notredame, E. O'Brien, and D. G. Higgins, "RAGA: RNA sequence alignment by genetic algorithm," *Nucleic Acids Research*, Vol. 25, No. 22, pp. 4570–4580, 1997.

[14] B. A. Shapiro, "An algorithm for comparing multiple RNA secondary structures," *Computer Applications in the Biosciences*, Vol. 4, No. 3, pp. 387–393, 1988.

[15] B. A. Shapiro and K. Zhang, "Comparing multiple RNA secondary structures using tree comparisons," *Computer Applications in the Biosciences*, Vol. 6, No. 4, pp. 309–318, 1990.

[16] Z. Wang and K. Zhang, "Alignment between two RNA structures," *Proceedings of the 26th International Symposium on Mathematical Foundations of Computer Science*, pp. 690–702, 2001.

[17] M. S. Waterman and T. F. Smith, "RNA secondary structure: A complete mathematical analysis," *Mathematical Bioscience*, Vol. 42, pp. 257–266, 1978.

[18] K. Zhang, "Computing similarity between RNA secondary structures," *In Proceedings of IEEE International Joint Symposia on Intelligence and Systems*, pp. 126–132, 1998.

[19] K. Zhang and D. Shasha, "Simple fast algorithms for the editing distance between trees and related problems," *SIAM Journal on Computing*, Vol. 18, No. 6, pp. 1245–1262, 1989.

[20] M. Zuker and D. Sankoff, "RNA secondary structures and their prediction," *Mathematical Bioscience*, Vol. 46, pp. 591–621, 1984.