Prediction of RNA Secondary Structures by Genetic Algorithms *

Ming-Cheng Lin, Chang-Biau Yang and Kuo-Si Huang
Department of Computer Science and Engineering,
National Sun Yat-sen University, Kaohsiung, Taiwan
E-mail: cbyang@cse.nsysu.edu.tw

ABSTRACT

Many methods can be used to predict the secondary structure of an RNA molecule. One of the methods is the dynamic programming approach. However, the dynamic programming approach usually takes too much time. Thus, it is not very practical to solve the problem of long sequences with dynamic programming. RAGA (RNA Sequence Alignment by the Genetic Algorithm) is a genetic algorithm to align two similar sequences where the structure of one of them, the master sequence, is known and the other (slave sequence) is unknown. We can predict the structure of an RNA molecule by analyzing several homologous sequence alignments. In this paper, we add a new operation in RAGA to mutate the residues of the base pairs in the master sequence and then realign the two sequences again. Our experimental results show that our new operation yields a big improvement over other traditional operations, such as crossover and mutation, in RAGA.

Keywords: computational biology, RNA, secondary structure, alignment, genetic algorithm

1 INTRODUCTION

RNA is a single strand of nucleotides composed of adenine (A), guanine (G), cytosine (C) and uracil (U) and it can fold back on itself to form its secondary structure with base pairs like $A \equiv U$, $G = C$, and $G - U$. However, an RNA sequence can fold to form several possible secondary structures. Determining which is the correct secondary structure is called the *RNA secondary structure prediction* problem [18].

One of the simple methods to predict the secondary structure of an RNA sequence is based on the number of base pairs. Waterman et al. designed two simple dynamic programming algorithms to find the maximum number of base pairs in an RNA sequence [17]. Some researchers tried to align several RNA sequences with the secondary structure information. Those methods align and fold sequences simultaneously [6, 13].

Some secondary structures with pseudoknots are more complex and difficult to analyze. In order to predict RNA secondary structures with pseudoknots, some researchers designed the model of tree grammars to represent these structures [7, 8, 16]. Akutsu used dynamic programming to solve the RNA secondary structure prediction problem with simple pseudoknots in $O(n^4)$ time [1]. Tabaska and Stormo aligned an RNA sequence to a structure with pseudoknots in polynomial time [14].

*This research work was partially supported by the National Science Council of the Republic of China under contract NSC-90-2213-E-110-015.

Unfortunately, these methods can predict only the structure of an RNA sequence with length no more than 200 in acceptable time. Corpet and Michot designed a heuristic algorithm to identify which portions of two sequences can be aligned without the structure information, and others portions are aligned by using a specialized dynamic programming algorithm. [2]. This method cannot predict structures with pseudoknots. Notredame et al. used a genetic algorithm (GA) to achieve the optimization of sequence alignment. It can solve the problem with pseudoknots and possibly predict the structure of an RNA sequence with length more than 2000 [12].

For secondary structure prediction, a genetic algorithm is more practical since it can solve a problem of large size. In this paper, we shall solve the problem of RNA secondary structure prediction with genetic algorithms. We shall try to add some useful operations (with the structure information) to increase accuracy of sequence alignment. Our experimental results show that our new operation gets a big improvement to other traditional operations, such as crossover and mutation.

The organization of this paper is as follows. In Section 2, we shall introduce the RNA sequence alignment by genetic algorithm (RAGA), proposed by Notredame et al. In Section 3, we shall present our method combined with RAGA. Our experimental results and conclusions will be given in Section 4 and Section 5, respectively.

2 RNA SEQUENCE ALIGNMENT BY THE GENETIC ALGORITHM

A genetic algorithm (GA) is an adaptive method for solving optimization problems according to its fitness value function [5, 10]. In GAs, there are several individuals called *chromosomes* and the set of these chromosomes is called a *population*. In a population, pairs of chromosomes with good fitness values will produce new chromosomes with some operations (reactions), such as *crossovers* or *mutations*. Some of these operations retain the most part of significant information in chromosomes and some can avoid missing some useful information during the production. After these operations are applied, new chromosomes will be selected to produce the next generation according to their *fitness values*. If the fitness value of one chromosome is higher, it usually produces more suitable offspring.

The genetic algorithm (GA) can be used to predict the secondary structure of a long RNA sequence by aligning another similar sequence which sequence has structured information, as shown in Figure 1. The genetic algorithm designed by Notredame et al. is called RAGA (*RNA sequence alignment by genetic algorithm*) [12]. They also used their previous package, SAGA (Sequence Alignment by Genetic Algorithm) to help them to develop this algorithm [11].

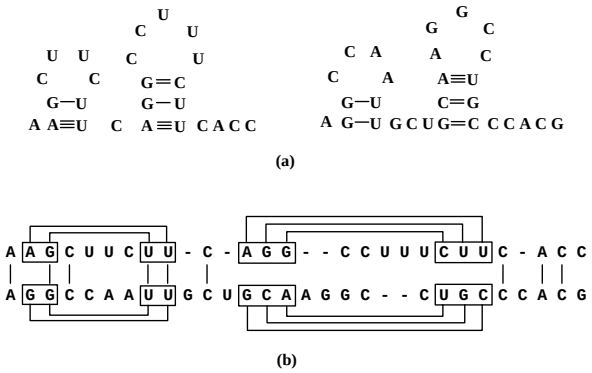

Figure 1: A correct RNA sequence alignment. (a) The real secondary structures of two RNA sequences. (b) Two-sequence alignment with the structure information of both of them.

In RAGA, the score function is designed by Corpet and Michot [2]. This score function is combined by three different scores as follows. (1) Pri: the primary score; (2) Sec: the secondary score; and (3) $GapS$: the gap penalty score.

Suppose we are given two similar RNA sequences $A = a_1a_2\cdots a_m$ and $B = b_1b_2\cdots b_n$. In addition, suppose the secondary structure of A is known. In other words, the set of base pairs in A is also given. The three scores are as follows.

Pri: If a_i is matched with b_j, i.e. $a_i = b_j$, for $a_i \in A$, $b_i \in B$, the aligned score is 1, otherwise it is 0. Pri is the sum of the values of all positions in the sequence alignment.

Sec: Suppose there is a base pair (a_p, a_q) in A and (b_i, b_j) are two nucleotides in B aligned with (a_p, a_q). If (b_i, b_j) also forms a base pair, according to equation 1, the sum of all pairing scores of (b_i, b_j)'s is Sec. The pairing score is defined as follows:

$$(b_i, b_j) = \begin{cases} 2 & \text{if } (b_i, b_j) \in \{(\text{G, C})\}, \\ 1 & \text{if } (b_i, b_j) \in \{(\text{A, U}), (\text{G, U})\}, \\ 0 & \text{other pairs.} \end{cases} \quad (1)$$

$GapS$: Sometimes, there exist gaps in loops or non-structured regions after sequence alignment. In order to reflect these two different situations in a chromosome, they use two specific gap penalties in the model of Corpet and Michot [2]. The penalty for opening a gap in a loop is called $GPLO$, which means these gaps exist between two stacked pairs in stems. The penalty for opening a gap in non-structured regions is called GP. And the third penalty ($GPLE$) depends on the length of the gaps in non-structured regions.

In the RNA sequence alignment, the total gap penalty is given as follows:

$$gap\ penalty = na \times GPLO + nb \times GP + nc \times GPLE, \quad (2)$$

where na is the number of gaps in a loop, nb is the number of other gaps except terminal gaps and nc is the total length of all gaps excluding terminal gaps.

Now the total alignment score is given as follows:

$$complete\ alignment\ score = Pri + \alpha \times Sec - gap\ penalty, \quad (3)$$

where α is a parameter associated with the structure information. If $\alpha = 0$, the alignment score is the same as that of the traditional sequence alignment (without the secondary structure information).

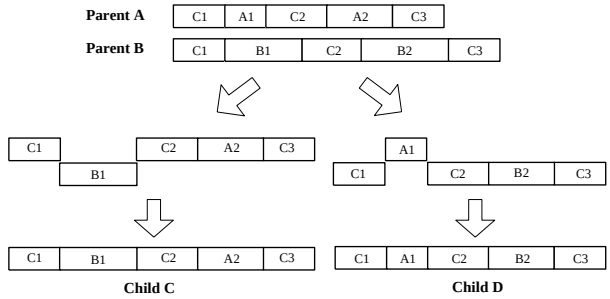

Figure 2: Two individuals produced by uniform crossover, in which C1, C2 and C3 are consistent blocks.

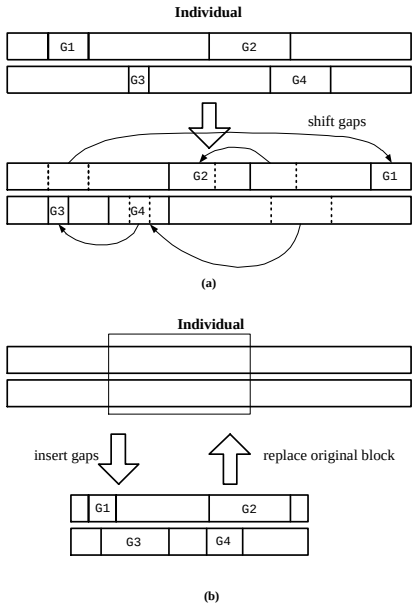

Figure 3: A gap shifting and insertion to form a new individual. (a) Gap shifting. (b) Gap insertion.

In RAGA, the two operations *crossovers* and *mutations* are as follows [12]:

Crossover: The uniform crossover is implemented in RAGA by mapping the consistent blocks from the parents. And the inconsistent blocks are randomly or deterministically chosen for mapping into the children. An example is shown in Figure 2.

Gap shifting: This operation randomly selects a gap from an alignment and shifting it in one direction until there is no further improvement.

Gap insertion: When aligning or realigning the regions of two sequences, this operation adds random size of gaps into the alignment.

Figure 3 shows examples of gap shifting and gap insertion.

RAGA is derived from the simple genetic algorithm proposed by Goldberg [5]. Suppose we are given two RNA sequences with the structure information of one of them. Each of the two-sequence alignments is called an *individual (chromosome)*. Initially, the *population*, consisting of many individuals, is established with ClustalW [15] or with

the method designed by Gerstein [4].

Next, RAGA selects some individuals with higher fitness value to produce the children by randomly choosing some operations, such as crossovers and mutations. At each generation, RAGA will produce the children whose number is the same as that of the individuals in the population.

Finally, a half of the population is replaced by the children of better scores. This procedure is repeated iteratively to produce the successive generations until there is no more improvement. Then, a better alignment (locally optimal) will be obtained.

3 OUR METHODS

Our aim is to find possible mismatched stems (Definition 2) in the initial alignment of the two given sequences and realign the parts of these possible mismatched stems to get better alignment. We shall propose a simple method that helps us to align two sequences with the structure information.

Definition 1 *Given two RNA sequences (the master sequence and slave sequence) and the secondary structure information of the master sequence, the problem of RNA secondary structure prediction is to find the secondary structure of the slave sequence.*

In addition, it is also assumed that an initial alignment of the two RNA sequences is given. The initial alignment may be obtained by a random alignment or some other alignment method.

Definition 2 *In an RNA sequence, a stem consists of some stacked pairs. For example, an RNA sequence* = $A_1A_2G_3C_4U_5U_6C_7C_8U_9U_{10}C_{11}$, (A_1, U_{10}) *and* (A_2, U_9) *are base pairs. The set of stacked pairs* (A_1, U_{10}) *and* (A_2, U_9) *form a stem.*

First, we randomly select an individual or select an individual with a lower fitness score from the population. Note that an individual should contain two sequences with an initial alignment. From the structure information of the master sequence, we can find all stems in the master sequence, as shown in Figure 4. Next, we will evaluate the scores of the stems in two aligned sequences.

Definition 3 *Each edge in a stem is called a border of this stem. For example, an RNA sequence* = $A_1A_2G_3C_4U_5U_6C_7C_8U_9U_{10}C_{11}$, (A_1, U_{10}) *and* (A_2, U_9) *are base pairs.* A_1A_2 *and* U_9U_{10} *are two borders in a stem.*

Definition 4 *A block is the extension of the borders until touching other stems and it contains two subsequences in an individual.*

Because each stem is composed of some stacked pairs, it has two borders (Definition 3) in a sequence, as shown in Figure 5 (a). In order to increase the accuracy, we can extend the range of one border until touching other stems. The range of the two aligned sequences is one block (Definition 4). Sometimes, there is only one block in one stem, as shown in Figure 5. Sometimes, there are two blocks in one stem, as shown in Figure 6. There are some blocks in an individual. These blocks will help us to align some regions of two RNA sequences with the structure information.

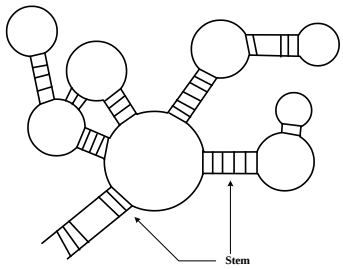

Figure 4: Some stems in an RNA sequence.

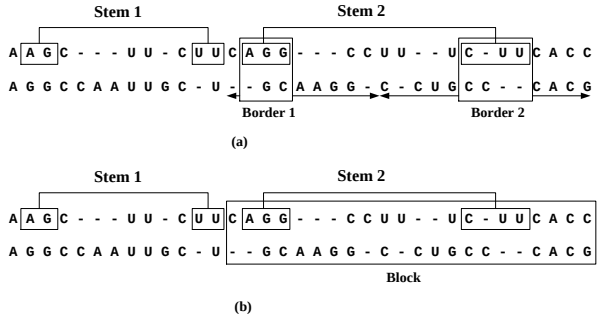

Figure 5: No other stem inside stem 2. (a) Two borders in stem 2 and the extension of each border. (b) The block containing the borders of stem 2.

In order to find which blocks need to be realigned, we define a score *b_score* which consists of primary score (*priS*) and secondary score (*secS*) as follows:

$$b\_score = (priS + secS)/length, \tag{4}$$

where *priS* and *secS* are defined as follows:

priS: In an individual (consisting of sequences A and B with an initial alignment), if $a_i = b_i \neq$ '-' , the value is 1, otherwise it is 0. *priS* is defined as the sum of the values in all positions of the two subsequences in this block.

secS: Suppose (a_i, a_j) is a base pair in A. If (b_i, b_j) is also a base pair of B, the value is set to 2, otherwise 0. *secS* is defined as the sum of the pairing scores in B induced by the structure information of A in the block.

length: It is the number of nucleotides of the master sequence in the block. Here we ignore gaps. For example, if an RNA sequence is `A-A-GCUGGC`, then $length = 8$.

We can assume that one block with higher score *b_score* indicates that under the alignment, the two RNA sequences have the similar structure information (stems) in the block. Thus, we take out some blocks with lower score from the individual, and realign these blocks. Next, we randomly replace the nucleotides of the stems in the master sequence in the blocks. Then we align the two sequences in the blocks by using the LCS (*longest common subsequence*) algorithm. Here, we use a measure score to realign the two sequences. Suppose we are given sequences $A = a_1a_2...a_m$ and $B = b_1b_2...b_n$, and the secondary structure information of A. The measure score $f(a_i, b_j)$ for LCS is given as follows:

$$f(a_i, b_j) = \begin{cases} \beta & \text{if } a_i = b_j \neq \text{'-'} \text{ and } a_i \text{ is in the stems of } A, \\ 1 & \text{if } a_i = b_j \neq \text{'-'} \text{ and } a_i \text{ is not in the stems of } A, \\ 0 & \text{if } a_i \neq b_j,\ a_i \neq \text{'-'} \text{ and } b_j \neq \text{'-'}. \end{cases} \tag{5}$$

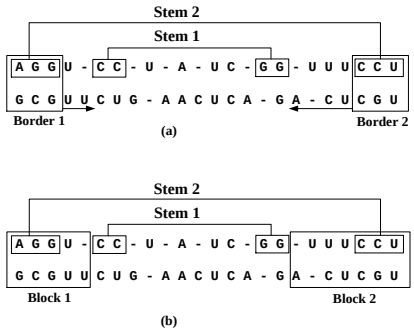

Figure 6: Stem 1 inside stem 2. (a) Two borders in stem 2 and the extension of each border until touching other stems. (b) Each block containing one of the borders in stem 2.

If $a_i \neq b_j$ and one of them is equal to '-', the gap penalty is set to 1. Let $L_{i,j}$ denote the length of the longest common subsequence of $a_1a_2...a_i$ and $b_1b_2...b_j$. $L_{i,j}$ can be calculated by the following formula [9]:

$$L_{i,j} = \max\{L_{i-1,j-1} + f(a_i, b_j), L_{i-1,j} - 1, L_{i,j-1} - 1\}.$$
$$L_{i,0} = L_{0,j} = 0, \quad \text{for } 0 \leq i \leq m, \text{ and } 0 \leq j \leq n. \tag{6}$$

After realigning the two sequences, we restore the stems in the master sequence before we replace. Finally, we replace the original blocks in the individual with these modified blocks.

An example of this realignment procedure is shown in Figure 7. In the example, suppose we get an individual C from the population, and C consists of sequences A and B as follows:

A = `AAGC---UU-CUUCAGG---CCUU--UC-UUCACC`
B = `AGGCCAAUUGC-U--GCAAGG-C-CUGCC--CACG`

There are two stems in A, as shown in Figure 7 (a). Suppose *stem 2* in A is dissimilar to B according to the alignment score. We extend the borders of *stem 2* and get the block, called *block*1, as shown in Figure 7 (b). And this block contains two sequences $S1$ and $S2$ as follows.

$S1$ = `CAGG---CCUU--UC-UUCACC`
$S2$ = `--GCAAGG-C-CUGCC--CACG`

Next, we randomly replace the nucleotides of *stem 2* in $S1$. Suppose a_i, $a_j \in S1$ and (a_i, a_j) is a base pair. If we change a_i to a new a'_i, then the a_j must also be changed to a'_j, such that (a'_i, a'_j) is also a base pair, keeping the original structure of A, as shown in Figure 7 (c). Suppose we get a new sequence $S1'$ as follows:

$S1'$ = `CGCA---CCUU--UU-GCCACC`

Here, in order to enhance the characteristics of the structure. We suppose that aligning *stem 2* has a better score than aligning other parts of the sequences in the block. We get the new block, called *block*2, with two aligned sequences $Sn1$ and $Sn2$ as follows:

$Sn1$ = `CGCA--CCUUUUGCC-ACC`
$Sn2$ = `-GCAAGGC--CUGCCCACG`

And it is shown in Figure 7 (d). Then, we restore the nucleotides of stem 2 in $Sn1$ and replace *block*1 with *block*2 in C. Finally, we get a new individual C' and evaluate its fitness value to determine whether it is suitable to be put into the population.

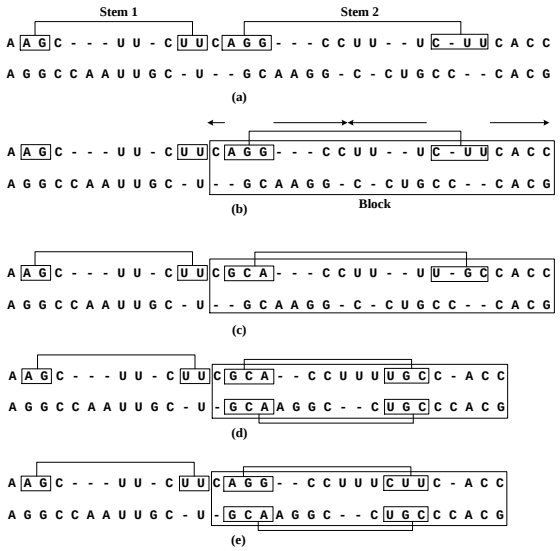

Figure 7: The procedure of the realignment of two sequences. (a)An individual of the two sequences with initial alignment. (b) Extending stem 2 and finding the block. (c) Randomly replacing the nucleotides of stem 2. (d) Realigning the two sequences in the block with the LCS algorithm. (e) The final result of the realignment, obtained by restoring the original nucleotides.

We summarize our above stem mutation procedure as the following algorithm:

Algorithm: Stem Mutation

Input: An individual of sequences A and B with an initial alignment, and the secondary structure information of A.

Output: A new individual of A and B with a better structure alignment.

Step 1: Find the stems of A with low scores.

Step 2: Extend the borders of these stems to get the blocks.

Step 3: Randomly replace the nucleotides of the stems in each block. Apply the LCS algorithm to realign the two sequences in each block.

Step 4: After realigning in each block, restore the original nucleotides of A.

Step 5: Replace these new blocks to the old blocks in the individual, and a new individual is formed.

Step 6: Discard the new individuals with low score.

Here we can find some stems with lower scores in an initial alignment. We replace the nucleotides in these stems and perform realignment to increase the probability of better structure alignment. Our new operation is added into the pool of the operations in RAGA and our modified RAGA is as follows.

Algorithm: Modified RAGA

Input: Two homologous RNA sequences A and B, and the secondary structure information of A.

Output: The sequence alignment of A and B with the structure information of A.

Step 1: Randomly produce some pairwise alignments of A and B, and put them into the population.

Step 2: According to the fitness scores of these alignments, select the alignments with good score, called parents, from the population, and begin to evolve.

Step 3: Produce the children from the parents by mutations, crossovers, and our method (Stem Mutation). Evaluate the scores of the children.

Step 4: Put the children with better scores into the population and replace some parents with lower scores.

Step 5: Repeat Step 2 to Step 4 until no improved alignment is produced in each generation.

4 EXPERIMENTAL RESULTS

In our experiments, there are several test cases referring to the experimental data from Notredame et al. [12]. All structure information and RNA sequences are obtained from the small subunit (SSU) [3]. SSU is an rRNA database and it contains a large number of alignments of rRNA which are made by hand and called *reference alignments*. These alignments include the structure information inside. The reference alignment in each test case is obtained from the SSU in RAGA. We compare the experimental results in each test case by dynamic programming (DP), RAGA, our modified RAGA (MRAGA) in Table 1.

There are two sequences in each test case. One is the master sequence, denoted as A, with a known structure; another is the slave sequence, denoted as B, without knowing its structure. Some parameters are used to measure the similarity of the two sequences. *Distance* is the estimated mean number of substitutions per position in the reference alignment of the two sequences. *Similarity* represents the structure information in the reference alignment. For example, suppose sequence $A = a_1g_2a_3a_4a_5c_6u_7$ and sequence $B = a_1g_2c_3a_4a_5a_6g_7u_8u_9$. The base pairs of B are (a_1, u_9), (g_2, u_8) and (c_3, g_7). The base pairs of A are (a_1, u_7) and (g_2, c_6). And according to the structure alignment, in B, we can find two base pairs, (a_1, u_9) and (g_2, u_8), corresponding to the base pairs of A. The reference alignment made by hands is as follows:

A = `AG-aaa-CU`
B = `AGcaaagUU`

There are only two base pairs found in B by the structure information of A. Then $similarity = \frac{2}{3} \times 100\% = 66.6\%$. *Length* is the length of the reference alignment. α is the parameter of the objective function in RAGA.

Two dynamic programming methods are compared with RAGA and MRAGA. One is RNAlign, used in cases 3 and 5 for shorter sequences [2]. Another is Gotoh's algorithm, used in other test cases.

Here we introduce a measure ASS (aligned structure score) [12] to compare the structure alignments obtained by various methods with the reference alignment. NR is defined as the number of base pairs of the slave sequence aligned with the master sequence in the reference alignment. In other words, if a pair (b_i, b_j) of B is aligned with (a_p, a_q) of A in the reference alignment, and both (b_i, b_j) and (a_p, a_q) are base pairs, then the value is 1, otherwise it is 0. NR denotes the sum of these values of all positions in the slave sequence B.

NT is defined as the number of base pairs of the slave sequence counted in NR and also found by aligning with the master sequence in the test alignment. ASS is defined as $\frac{NT}{NR}$.

TC	The ratio of our operation in MRAGA				
	1/10	2/11	1/4	4/13	5/14
4	97.1	97.1	97.1	97.1	97.1
5	74.5	77.0	82.0	79.0	78.0
6	63.4	62.9	64.0	68.0	62.3
8	52.1	49.7	49.7	43.0	49.7

Table 3: The results with different ratio of our operation in the pool of operations.

In most cases, MRAGA is better than RAGA. In cases 7, 8 and 9, the master sequence is shorter than the slave sequence. Some stems of the slave sequence are possibly not found or mismatched in the stems of the master sequence. Our method is to enhance the stem information of the master sequence to realign the possible stems in the slave sequence. If the master sequence does not contain all stem structures of the slave sequence, some stems in the slave sequence still may be missed in our method. In case 9, we get no improvement. In cases 7 and 8, MRAGA has some improvement compared with RAGA.

Besides, if two sequences are very similar with the matched nucleotides, when we select the blocks with the worse scores to realign, sometimes the stems we found are the same as ones in the reference alignment. Thus, replacing the nucleotides of the stems in the master sequence may cause the original aligned stems to be missed . Fortunately, we can also get better results. In cases 1, 2 and 3, MRAGA has some improvement compared with RAGA.

In all cases, RAGA and MRAGA are better than DP when the structure alignment is required.

We also change the parameter β of LCS in our method, and compare MRAGA with RAGA. Here we test four cases, 4, 5, 6 and 8. In cases 4, 5 and 6, two sequences are of similar lengths and are aligned with different α in RAGA. In case 8, the master sequence is shorter than the slave sequence.

The ASS and the required computing time of MRAGA and RAGA are shown in Table 2. We also find that when β is set to 1, it usually has better score than others.

In Table 3, we change the ratio of our operation in the pool of operations in RAGA and compare the results in cases, 4, 5, 6 and 8. We find that when the ratio is 1/4, it usually has better score than others.

5 CONCLUSION

RAGA is a useful method to align two homologous RNA sequences. According to the structure information of one of the two sequences and fitness score, RAGA can produce a nearly optimal solution based on sequence alignment and structure alignment simultaneously. Besides, RAGA can predict the secondary structure of a longer RNA sequence than the dynamic programming approach and treat the pseudoknots as same as other simple structure in RNA sequences during the generation.

In RAGA, crossovers and mutations do not include the structure information. They used a meaningful score function to evaluate the structure alignment of two RNA sequences. But they did not handle crossovers or mutations with the structure information. In this paper, we propose a new method to realign two aligned RNA sequences by aligning their stems. This method can help RAGA doing sequence alignment more efficiently. It can also be used in the sequence alignment with other general alignment algorithms.

To align with the slave sequence, we randomly replace the nucleotides of the stems in the master sequence. It may

TC	Master	Slave	Distance	Similarity	Length	α	ASS		
							DP	RAGA	MRAGA
1	Homo sapiens	Oxtrichia nova	0.41	82.5	1914	1	85.3	89.1	89.9
2	Homo sapiens	Giarda ardeae	0.57	82.1	1895	3	65.2	80.6	86.8
3	Homo sapiens mitochondrion	Latimeria chalumnae mitochondrion	0.31	81.2	998	1	82.6	95.4	96.9
4	Homo sapiens mitochondrion	Xenopus laevis mitochondrion	0.43	84.9	985	1	77.4	97.1	97.1
5	Homo sapiens mitochondrion	Drosophila virilis mitochondrion	0.76	82.6	973	3	48.6	74.5	82.0
6	Homo sapiens mitochondrion	Apis mellifera mitochondrion	1.23	72.1	977	4	24.1	54.3	68.0
7	Homo sapiens mitochondrion	Penicillium chrysogenum mitochondrion	1.26	81.3	1478	4	15.7	56.3	56.9
8	Homo sapiens mitochondrion	Chlamydomonas reinhardtii mitochondrion	1.30	66.6	1271	4	8.90	46.1	52.1
9	Homo sapiens mitochondrion	Saccharomyces cerevisiae mitochondrion	1.33	80.3	1699	6	21.6	62.6	62.6

Table 1: TC: test case number; Master: an RNA sequence with a known structure; Slave, an RNA sequence without knowing its structure; Distance: mean number of substitutions per position in the reference alignment; α: value of α; *ASS*: percentage of pairs which are correctly aligned. These sequences can be obtained from SSU rRNA query.

TC	The parameter β of LCS in MRAGA							RAGA	
	1	2	3	4	5	6	time		time
4	95.1	97.1	95.1	97.1	90.7	97.1	401s	97.1	361s
5	79.0	73.0	72.5	74.5	76.0	82.0	833s	74.5	1155s
6	64.0	53.7	61.7	57.1	57.1	54.9	807s	57.1	1008s
8	49.7	49.7	33.3	40.0	36.4	34.0	1152s	40.0	1457s

Table 2: ASS and required time of MRAGA and RAGA. TC: test case number.

be more efficient by changing nucleotides in the stems with a deterministic way. The idea may be useful on the protein sequence alignment in the future.

References

[1] T. Akutsu, "Dynamic programming algorithms for RNA secondary structure prediction with pseudoknots," *Discrete Applied Mathematics*, Vol. 104, pp. 45–62, 2000.

[2] 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.

[3] Y. V. de Peer, J. Jansen, P. D. Rijk, and R. D. Wachter, "Database on the structure of small ribosomal subunit RNA," *Nucleic Acids Research*, Vol. 25, pp. 111–116, 1997.

[4] M. Gerstein and M. Levitt, "Using iterative dynamic programming to obtain accurate pairwise and multiple alignments of protein structures," *In Proceedings of the Fourth International Conference on Intelligent Systems in Molecular Biology*, Menlo Park, CA, AAAI Press, 1996.

[5] D. E. Goldberg, *Genetic Algorithms in Search Optimization and Machine Learning*. AddisonWesley Publishing Company, 1989.

[6] J. Kim, J. Cole, and S. Pramanik, "Alignment of possible secondary structures in multiple RNA sequences using simulated annealing," *Computer Applications in the Biosciences*, Vol. 12, Aug. 1996.

[7] B. Knudsen and J. Hein, "RNA secondary structure prediciton using stochastic context-free grammars and evolutionary history," *Bioinformatics*, Vol. 15, pp. 446–454, 1999.

[8] S. Kobayashi and T. Yokomori, "Modeling RNA secondary structures using tree grammars," *In Proceedings of Genome Informatics Workshop V*, pp. 29–38, 1994.

[9] R. C. T. Lee, S. S. Tseng, R. C. Chang, and Y. T. Tsai, *Introduction to the Design and Analysis of Algorithms*. Unalis Corporation, first ed., 1999.

[10] Z. Michalewicz, *Genetic Algorithms + Data Structures = Evolution Programs*. Springer-Verlag, 1992.

[11] C. Notredame and D. G. Higgins, "SAGA: sequence alignment by genetic algorithm," *Nucleic Acids Research*, Vol. 24, No. 8, pp. 1515–1524, 1996.

[12] 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.

[13] D. Sankoff, "Simultaneous solution of the RNA folding, alignment and protosequence problems," *SIAM Journal on Applied Mathematics*, Vol. 45, No. 5, pp. 810–825, 1985.

[14] J. Tabaska and G. Stormo, "Automated alignment of RNA sequences to pseudoknotted structures," *Fifth International Conference on Intelligent Systems for Molecular Biology*, The AAAI Press, Menlo Park, California (USA), pp. 311–318, 1997.

[15] J. Thompson, D. Higgins, and T. Gibson, "ClustalW: improving the sensitivity of progressive multiple sequence alignment through sequence weighting, psition-special gap penalties and weight matrix choice," *Nucleic Acids Research*, Vol. 22, pp. 4673–4690, 1994.

[16] Y. Uemura, A. Hasegawa, S. Kobayashi, and T. Yokomori, "Tree adjoining grammars for RNA structure prediction," *Theoretical Computer Science*, Vol. 210, pp. 277–303, Jan. 1999.

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

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