Spring 2016

A data science approach to pattern discovery in complex structures with applications in bioinformatics

Lei Hua
New Jersey Institute of Technology

ABSTRACT

A DATA SCIENCE APPROACH TO PATTERN DISCOVERY IN COMPLEX STRUCTURES WITH APPLICATIONS IN BIOINFORMATICS

by
Lei Hua

Pattern discovery aims to find interesting, non-trivial, implicit, previously unknown and potentially useful patterns in data. This dissertation presents a data science approach for discovering patterns or motifs from complex structures, particularly complex RNA structures. RNA secondary and tertiary structure motifs are very important in biological molecules, which play multiple vital roles in cells. A lot of work has been done on RNA motif annotation. However, pattern discovery in RNA structure is less studied. In the first part of this dissertation, an *ab initio* algorithm, named DiscoverR, is introduced for pattern discovery in RNA secondary structures. This algorithm works by representing RNA secondary structures as ordered labeled trees and performs tree pattern discovery using a quadratic time dynamic programming algorithm. The algorithm is able to identify and extract the largest common substructures from two RNA molecules of different sizes, without prior knowledge of locations and topologies of these substructures.

One application of DiscoverR is to locate the RNA structural elements in genomes. Experimental results show that this tool complements the currently used approaches for mining conserved structural RNAs in the human genome. DiscoverR can also be extended to find repeated regions in an RNA secondary structure. Specifically, this extended method is used to detect structural repeats in the 3'-untranslated region of a protein kinase gene.

The biological significance of a repeated hairpin found by DiscoverR is discussed, demonstrating the usefulness of the tool.

RNA junctions are important structural elements of RNA molecules. They are formed by three or more helices coming together in three-dimensional space. Recent studies have focused on the annotation of coaxial helical stacking (CHS) motifs within junctions. In the second part of this dissertation, a new method, called CHSalign, is designed, which is capable of finding patterns in RNA secondary structures with CHS motifs through aligning the structures. CHSalign works by (1) employing a random forests algorithm to predict coaxial stacking in junctions, (2) modelling junction topologies as tree graphs, and (3) using a novel dynamic programming algorithm to perform constrained tree pattern matching. CHSalign is intended to be an efficient alignment tool for RNAs containing similar junctions. Experimental results based on thousands of alignments demonstrate that CHSalign can align two RNA secondary structures containing CHS motifs more accurately than other RNA secondary structure alignment tools. CHSalign yields a high score when aligning two RNA secondary structures with similar CHS motifs or helical arrangement patterns, and a low score otherwise. This new method is implemented in a web server accessible on the Internet.

A DATA SCIENCE APPROACH TO PATTERN DISCOVERY IN COMPLEX STRUCTURES WITH APPLICATIONS IN BIOINFORMATICS

by
Lei Hua

A Dissertation
Submitted to the Faculty of
New Jersey Institute of Technology
in Partial Fulfillment of the Requirements for the Degree of
Doctor of Philosophy in Computer Science

Department of Computer Science

May 2016

APPROVAL PAGE

A DATA SCIENCE APPROACH TO PATTERN DISCOVERY IN COMPLEX STRUCTURES WITH APPLICATIONS IN BIOINFORMATICS

Lei Hua

Dr. Jason T.L. Wang, Dissertation Advisor Date
Professor of Computer Science, NJIT

Dr. James McHugh, Committee Member Date
Professor of Computer Science, NJIT

Dr. Dimitrios Theodoratos, Committee Member Date
Associate Professor of Computer Science, NJIT

Dr. Zhi Wei, Committee Member Date
Associate Professor of Computer Science, NJIT

Dr. Yi Chen, Committee Member Date
Associate Professor of School of Management, NJIT

BIOGRAPHICAL SKETCH

Author: Lei Hua

Degree: Doctor of Philosophy

Date: May 2016

Undergraduate and Graduate Education:

- Doctor of Philosophy in Computer Science,
  New Jersey Institute of Technology, Newark, NJ, 2016

- Master of Science in Computer Science,
  Nanjing University, P.R. China, 2001

- Bachelor of Science in Computer Science,
  Nanjing University, P.R. China, 1998

Major: Computer Science

Publications:

Hua, L., Song, Y., Kim, N., Laing, C., Wang, J.T.L. and Schlick, T. (2016) CHSalign: a web server that builds upon Junction Explorer and RNAJAG for pairwise alignment of RNA secondary structures with coaxial helical stacking. PLoS One 11(1): e0147097.

Song, Y., Hua, L., Shapiro, B. A. and Wang, J.T.L. (2015) Effective alignment of RNA pseudoknot structures using partition function posterior log-odds scores. BMC Bioinformatics 16:39.

Hua, L., Wang, J. T.L., Ji, X., Malhotra, A., Khaladkar, M., Shapiro, B. A. and Zhang, K. (2012) A method for discovering common patterns from two RNA secondary structures and its application to structural repeat detection. Journal of Bioinformatics and Computational Biology 10(4): 1250001.

Hua, L., Cervantes-Cervantes, M. and Wang, J. T.L. (2011) A new approach to the discovery of RNA structural elements in the human genome. Advances in Genomic Sequence Analysis and Pattern Discovery. Laura Elnitski, Helen Piontkivska and Lonnie R. Welch, editors. Singapore, World Scientific Publishing Company, pp. 117-132.

谨以此文献给我亲爱的家人：	*This dissertation is dedicated to my beloved family:*
儿子，祝家欣，	*my son, Felix Jiaxin Zhu,*
先生，祝捷博士，	*my husband, Dr. Jie Zhu,*
母亲，汪兆惠，	*my mother, Zhaohui Wang,*
父亲，花培元，	*my father, Peiyuan Hua,*
婆婆，王铭，	*my mother-in-law, Ming Wang,*
公公，祝世宁博士。	*my father-in-law, Dr. Shining Zhu.*
感谢你们的爱，支持和鼓励。	*Thanks for your love, support, and encouragement.*

ACKNOWLEDGMENT

This dissertation owes its existence to the help, support and inspiration of several people.

Firstly, I would like to express my sincere appreciation to my adviser, Dr. Jason T.L. Wang. His guidance and passion at research has led me through my Ph.D. study.

Secondly, I would like to thank all of my committee members, Dr. James McHugh, Dr. Dimitrios Theodoratos, Dr. Zhi Wei, and Dr. Yi Chen. Their supervision and advice have improved this work.

I am thankful to Dr. Kaizhong Zhang, Dr. Christian Laing, and Dr. Namhee Kim. They gave me many useful ideas about algorithm optimization and data analysis.

Here, I also want to thank all the previous members at the Data and Knowledge Engineering Laboratory of NJIT. Dr. Yang Song is a great partner in research. Without his contribution, the project of CHSalign would be more complex. And I am so lucky to have great friends here, in particular, Dr. Mugdha Khaladkar, Dr. Dongrong Wen and Dr. Tao Wu. I would never forget the time we worked and studied together.

TABLE OF CONTENTS

Chapter **Page**

1 INTRODUCTION 1

1.1 Background Information 1

1.2 Motivation and Organization 3

2 AN ALOGRITHM FOR DISCOVERING COMMON PATTERNS 5

2.1 Introduction 5

2.2 Algorithm 5

2.2.1 Representing RNA Secondary Structures by Trees 5

2.2.2 Common Patterns of Two Trees 12

2.2.3 Common Patterns of Two Forests 14

2.2.4 Filling in the Maximum Size Table 15

2.2.5 Algorithm Complexity 17

2.3 Program of DiscoverR 19

2.4 Comparison with Related Works 23

3 APPLICATIONS OF DISCOVERR 27

3.1 Repeats 27

3.2 Finding Genomic Regions within Conserved Substructures 28

3.3 Conclusions 35

TABLE OF CONTENTS
(Continued)

CHAPTER **PAGE**

4 PAIRWISE ALIGNMENT OF RNA SECONDARY STRUCTURES WITH COAXIAL HELICAL STACKING........ 36

4.1 Introduction........ 36

4.2 Materials and Methods........ 47

4.2.1 Tree Model Formalization........ 47

4.2.2 Alignment Scheme........ 51

4.2.3 Time and Space Complexity........ 62

4.2.4 Data Sets........ 63

4.3 Results and Discussion........ 65

4.3.1 Two CHSalign Web Server Versions........ 65

4.3.2 Performance Evaluation Using RMSD........ 70

4.3.3 Performance Evaluation Using Precision........ 73

4.3.4 Potential Application of CHSalign........ 76

4.4 Conclusions........ 81

5 CONCLUSIONS........ 85

5.1 Summary for DiscoverR........ 85

5.2 Summary for CHSalign........ 86

5.3 Future Work........ 88

BIBLIOGRAPHY........ 89

LIST OF TABLES

Table **Page**

3.1 Results of the Experiments Performed in this Study 32

4.1 The 24 RNA Full Structures in Dataset1 Selected from the Protein Data Bank (PDB) to Evaluate the Performance of the Alignment Methods Studied in his Dissertation.. 64

4.2 The Six Riboswitches Selected from the Protein Data Bank (PDB) to Demonstrate the Utility of Our Web Server. 77

4.3 Results Obtained by Aligning Seven Pairs of Riboswitches from Table 4.2. 79

LIST OF FIGURES

Figure **Page**

1.1 Example of an RNA secondary structure.. 2

2.1 The example of the RNA secondary structure with the hairpin, the bulge, the and the multi-branch loop. 7

2.2 Transform an RNA secondary structure to an ordered labeled tree................... 9

2.3 Cutting at the node labeled I24 (rt[19]) means removing the subtree rooted at the node labeled I24. 11

2.4 The substructure obtained by cutting at the nodes labeled P9 and P23 in the secondary structure in Figure 2.3. 12

2.5 (A) The shaded subtree $RT_1[i_q]$ is removed. (B) The shaded subtree $RT_2[j_t]$ is removed. (C) Neither $RT_1[i_q]$ nor $RT_2[j_t]$ is removed... 16

2.6 The node $rt_1[i]$ matches the node $rt_2[j]$. Thus, the size of the common patterns of tree $RT_1[i]$ and tree $RT_2[j]$ equals the size of the common patterns of forest $RF_1[i_1, i_{m_i}]$ and forest $RF_2[j_1, j_{n_j}]$ plus 1. 17

2.7 Procedure for computing $\Phi(i_1..i_{m_i}, j_1..j_{n_j})$ 18

2.8 Procedure for computing $\Psi(i, j)$ for all $1 \le i \le |RT_1|, 1 \le j \le |RT_2|$ 19

2. 9 (A) A query RNA. (B) A subject RNA... 20

2.10 (A) The pattern found in the query RNA. (B) The pattern found in the subject RNA. In (A), (B), beginning and ending positions of the contiguous bases on the common patterns found in the query RNA and subject RNA are displayed. 21

LIST OF FIGURES
(Continued)

Figure **Page**

2.11 Common pattern found by DiscoverR in the RNA molecule gnl|11825421 is highlighted in blue. 22

2.12 Common pattern found by DiscoverR in the RNA molecules gi|118130856 is highlighted in blue. 23

2.13 Examples illustrating the differences between DiscoverR and related algorithms. 25

3.1 Illustration of a structural repeat, circled with solid lines and highlighted in blue, detected in an RNA secondary structure in the 3'-UTR of the DMPK gene. 28

3.2 Illustration of the flowchart of our approach for mining conserved structural RNAs in the human genome. 31

4.1 The RNA molecule (PDB code: 1Y26) with three-way junction. (A) 3D crystal strcutrue view. Helix 1 is showed in blue. Helix 2 is showed in gree. And Helix 3 is showed in red. (B) The secondary structure view. 38

4.2 The four possibilities of three-way junctions, (A) for H1H2, (B) for H2H3, (C) for H1H3 and (D) for none. 49

4.3 The seven possibilities of four-way junctions, (A) for H1H2, (B) for H2H3, (C) for H3H4, (D) for H1H4, (E) for H1H2-H3H4, (F) for H1H4-H2H3 and (G) for none.. 40

4.4 The flowchart of Junction Explorer. 41

4.5 The number of junctions for each junction order.. 41

4.6 The screenshot of the web server of Junction Explorer. 42

4.7 The screenshot of the result of 1E8O. 43

4.8 The screenshot of the Web Sever of CHSalign. 46

4.9 Transformation of an RNA 3D molecule into an ordered labeled tree. 49

LIST OF FIGURES
(Continued)

Figure **Page**

4.10 Example of an alignment between two RNA molecules. (A) The 3D crystal structure of the adenine riboswitch (PDB code: 1Y26) and its tree representation T_1. (B) The 3D crystal structure of the Alu domain of the mammalian signal recognition particle (SRP) (PDB code: 1E8O) and its tree representation T_2. 52

4.11 Illustration for both $t_1[i]$ and $t_2[j]$ junctions, and $\Psi(t_1[i]) = \Psi(t_2[j])$.......... 54

4.12 Illustration of four possibilities when both $t_1[i]$ and $t_2[j]$ are helices.......... 56

4.13 Illustration of the alignment when both $t_1[i]$ and $t_2[j]$ are hairpin loops.......... 57

4.14 Illustration of case 4.......... 59

4.15 Illustration for case 5. 60

4.16 Illustration for case 7. 61

4.17 The screenshot of input for CHSalign_u. 67

4.18 The screenshot of the result for CHSalign_u in Figure 4.6. 68

4.19 The screenshot of CHSalign_p. 69

4.20 The result of CHSalign_p in Figure 4.8.......... 70

4.21 Comparison of the RMSD values obtained by CHSalign_u, CHSalign_p, RSmatch, RNAforester and FOLDALIGN.......... 72

4.22 Comparison of the PR values obtained by CHSalign_u, CHSalign_p, RNAforester, SETTER, RSmatch and FOLDALIGN.......... 75

4.23 Illustration of the coaxial stacking patterns in the six riboswitches used to demonstrate the utility of our web server.......... 78

4.24 The download page of CHSalign.......... 84

CHAPTER 1

INTRODUCTION

1.1 Background Information

Ribonucleic acid (RNA) is formed from DNA by transcription. Unlike double-stranded DNA, RNA is a single-stranded molecule, which consists of a chain of nucleotides linked together by covalent chemical bonds. Each nucleotide contains one of the four bases: Adenine (A), Cytosine (C), Guanine (G) and Uracil (U).

Tools that align biosequences (DNA, RNA, protein), such as FASTA and BLAST, are valuable in identifying homologous regions, which can lead to the discovery of functional units, such as protein domains, DNA *cis* elements, and so on [1,2]. However, their success is more evident in the study of DNA and protein than of RNA. This is mainly because the sequence similarity among DNAs and proteins can usually faithfully reflect their functional relationship, whereas additional structure information is needed to study the functional conservation among RNAs. Therefore, it is necessary to take into account both structural and sequence information in analyzing RNA data.

RNA structure determination via biochemical experiments is laborious and costly. Predictive approaches are valuable in providing guide information for wet lab experiments. RNA structure prediction is usually based on phylogenetic conservation of base-paired regions or thermodynamics of RNA folding. The former infers RNA structures based on covariation of base-paired nucleotides [3-6]. The latter uses thermodynamic properties of various RNA local structures, such as base pair stacking, hairpin loop, and bulge, to derive thermodynamically favorable secondary structures. A dynamic programming algorithm is used to find optimal or suboptimal structures. The most well-known tools belonging to this category are MFOLD [7,8] and RNAfold in the Vienna RNA package [9,10]. Similar tools have been developed in recent years to predict higher order structures, such as pseudoknots [11]. Figure 1.1 shows an example of RNA secondary structure folded by RNAfold of Vienna RNA package [9] and the view is generated by RnaViz 2 [12].

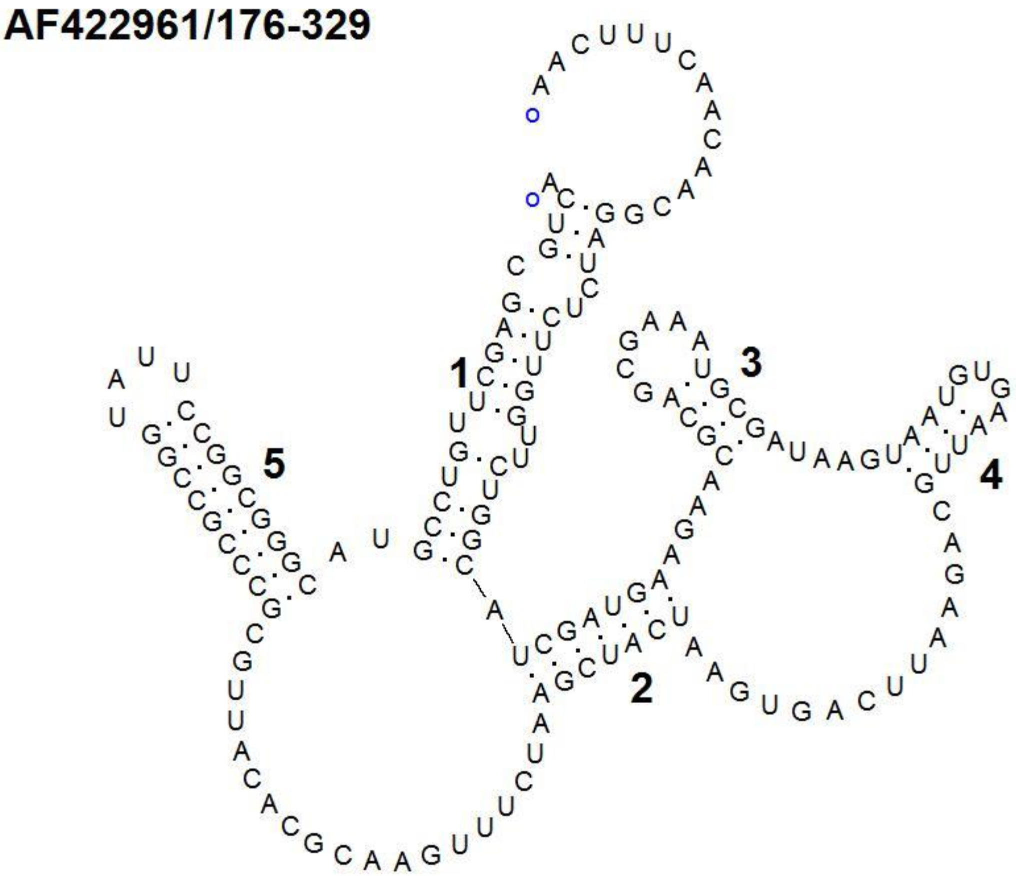


Figure 1.1 Example of an RNA secondary structure.

RNA motifs or patterns refer to structural particularities or conserved substructures of RNA. RNA motifs have been extensively studied for noncoding RNAs (ncRNAs), such as transfer RNA (tRNA), ribosomal RNA (rRNA), small nuclear RNA (snRNA) and small nucleolar RNA (snoRNA), as well as small interfering RNA (siRNA) and microRNA (miRNA) [13,14]. More recently, the structures in the untranslated regions (UTRs) of messenger RNAs (mRNAs) draw much attention from researchers [15,16]. Biochemical and genetic studies have demonstrated a myriad of functions associated with the UTRs in mRNA metabolism, including RNA translocation, translation, and RNA stability [17-19]. For example, the iron response elements (IREs) found in both 5' and 3' UTRs of genes are involved in iron homeostasis in higher eukaryotic species [16]. These motifs interact with iron regulatory proteins (IRPs) and play an important role in RNA stability and translation.

1.2 Motivation and Organization

The objective of this dissertation is to present algorithms for pattern discovery in RNA secondary structures. The first one is an *ab initio* algorithm, named DiscoverR, for finding common patterns from two RNA secondary structures. The algorithm works by representing RNA secondary structures as ordered labeled trees and performs tree pattern discovery using an efficient dynamic programming algorithm. The details of the algorithm are presented in Chapter 2. In Chapter 3, two applications of DiscoverR are demonstrated. One is to identify and extract the largest common substructures from two RNA molecules of different sizes, without prior knowledge of the locations and topologies of these substructures. The other is to find repeated regions in an RNA secondary structure, where DiscoverR can detect structural repeats in the 3'-untranslated region of a protein kinase gene.

In Chapter 4, a new method, called CHSalign, is designed, which is capable of finding patterns in RNA secondary structures with coaxial helical stacking (CHS) motifs through aligning the structures. CHSalign works by (1) employing a random forests algorithm to predict coaxial stacking in junctions, (2) modelling junction topologies as tree graphs, and (3) using a novel dynamic programming algorithm to perform constrained tree pattern matching. CHSalign is intended to be an efficient alignment tool for RNAs containing similar junctions. Experimental results based on thousands of alignments demonstrate that CHSalign can align two RNA secondary structures containing CHS motifs more accurately than other RNA secondary structure alignment tools. CHSalign yields a high score when aligning two RNA secondary structures with similar CHS motifs or helical arrangement patterns, and a low score otherwise. This new method has been implemented in a web server, and the program is also made freely available, at http://bioinformatics.njit.edu/CHSalign/.

Finally, Chapter 5 concludes the dissertation and points out some directions for future research.

CHAPTER 2

AN ALOGRITHM FOR DISCOVERING COMMON PATTERNS

2.1 Introduction

Many functional RNAs exhibit a highly conserved secondary structure although their nucleotide sequences share little similarity. Thus, in developing effective tools for comparing and detecting the functional RNAs as well as important evolutionary divergences, researchers often consider the secondary structures of the RNA molecules [20-22].

We present here a novel algorithm, named DiscoverR, for detecting common patterns from two RNA secondary structures. Built upon the previous accomplishment in tree pattern finding [23], DiscoverR works by representing RNA secondary structures as ordered labeled trees, then performs tree pattern discovery by allowing certain subtrees to be removed at no cost. This algorithm is capable of identifying and extracting the largest common substructures from two RNA molecules of different sizes, without prior knowledge of the locations and topologies of these substructures. It is faster comparing to the existing algorithm for general approximate tree pattern discovery [23].

2.2 Algorithm

2.2.1 Representing RNA Secondary Structures by Trees

Let *RS* be an RNA sequence containing nucleotides or bases A, U, C, G. *RS*[*i*] denotes the base at position *i* of *RS* and *RS*[*i*, *j*] is the subsequence starting at position *i* and ending at position *j* in *RS*. Let *R* be the secondary structure of *RS*. A base pair connecting position *i* and position *j* in *R* is denoted by (*i*, *j*) and its enclosed sequence is *RS*[*i*, *j*]. A loop in *R* refers to a hairpin, a bulge, an internal or a multi-branch loop [9,24]. Given a loop *L* in the secondary structure *R*, the base pair (*i**, *j**) in *L* is called the *exterior pair* of *L* if position *i** (*j**, respectively) is closest to the 5' (3', respectively) end of *R* among all positions in *L*. All other non-exterior base pairs in *L* are called *interior pairs* of *L*[25]. Figure 2.1 gives the example of the RNA secondary structure, which shows the hairpin, the bulge, the internal and the multi-branch loop.

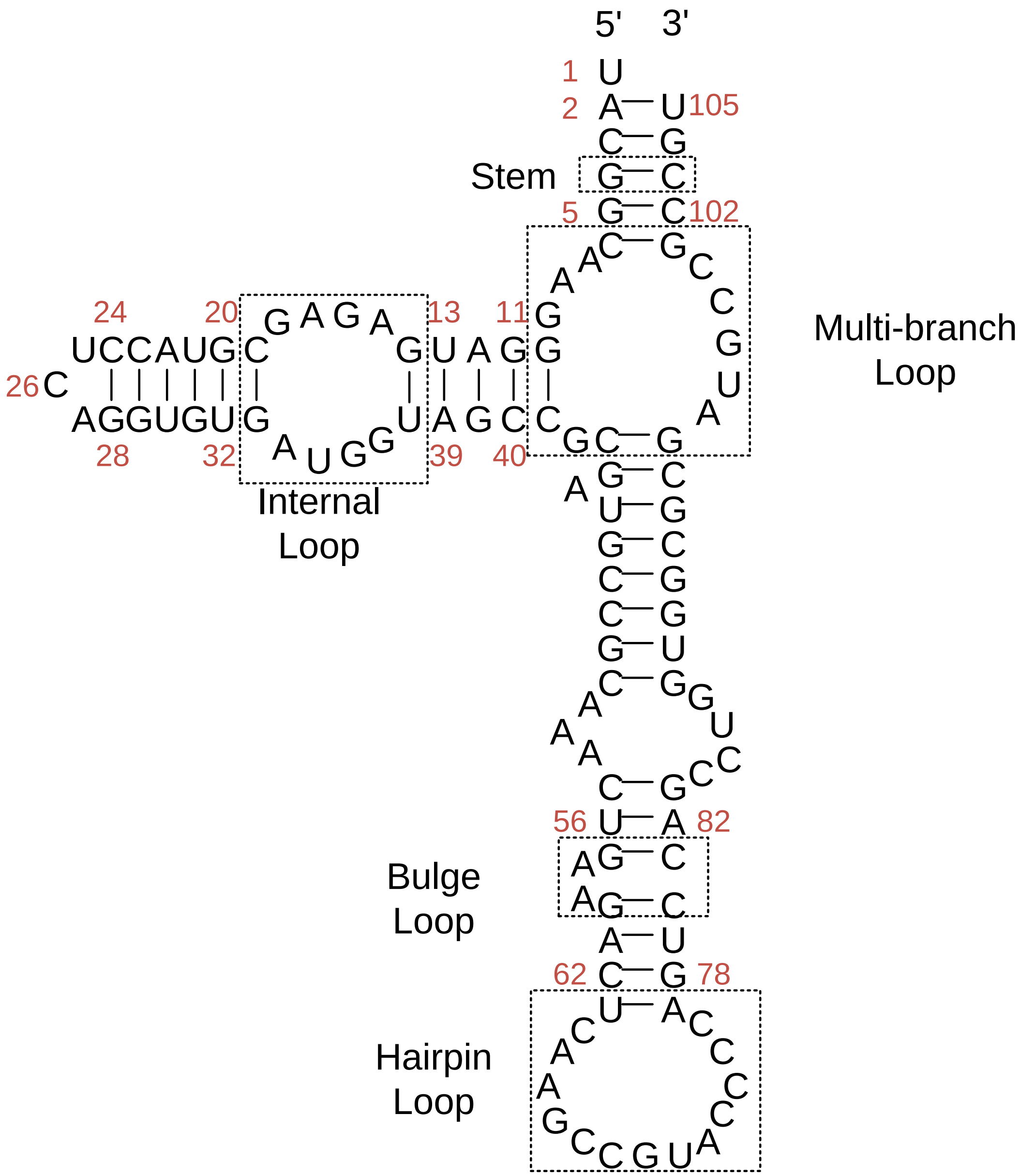


Figure 2.1 The example of the RNA secondary structure with the hairpin, the bulge, the internal and the multi-branch loop.

Here the RNA secondary structure *R* of the sequence *RS* is modeled by an ordered labeled tree *RT* in which each node has a label and the left to right order of siblings is significant (Figure 2.2). With this model, pseudoknots are not allowed. Each node in *RT* corresponds to a base pair in *R* and vice versa. Base pairs are numbered according to the order from the 5' end to the 3' end of *R*. Except for the exterior pairs of loops, the *k*th base pair of *R* corresponds to the node labeled "P*k*" in *RT* and vice versa. For example, the node labeled "P*3*" in the tree *RT* shown in Figure 2.2(B) corresponds to the 3rd base pair in the RNA secondary structure *R* shown in Figure 2.2(A).

The exterior pair of a multi-branch loop containing *n* interior pairs in *R* corresponds to a node *v* with *n* children in *RT* with each child corresponding to one of the *n* interior pairs. Assuming the exterior pair is the *k*th base pair in *R*, the node label of *v* in *RT* is "M*k*". The exterior pair of a bulge loop (internal loop, hairpin loop, respectively) in *R* corresponds to the node labeled "B*k*" ("I*k*", "H*k*", respectively) in *RT* if the exterior pair is the *k*th base pair in *R*. For example, the node labeled "M5" ("B18", "I24", "H31", respectively) in the tree *RT* shown in Figure 2.2(B) corresponds to the exterior pair of the multi-branch loop (bulge loop, internal loop, hairpin loop, respectively) where the exterior pair is the 5th (18th, 24th, 31st, respectively) base pair in the RNA secondary structure *R* shown in Figure 2.2(A). For each node *v* in the tree *RT*, we use *NB*(*v*) to represent the number of bases *v* has. If the node label of *v* is "P*i*" for some *i*, i.e., *v* corresponds to a base pair, *NB*(*v*) = 2. If *v* corresponds to the exterior pair of a loop, *NB*(*v*) equals the number of bases in that loop.

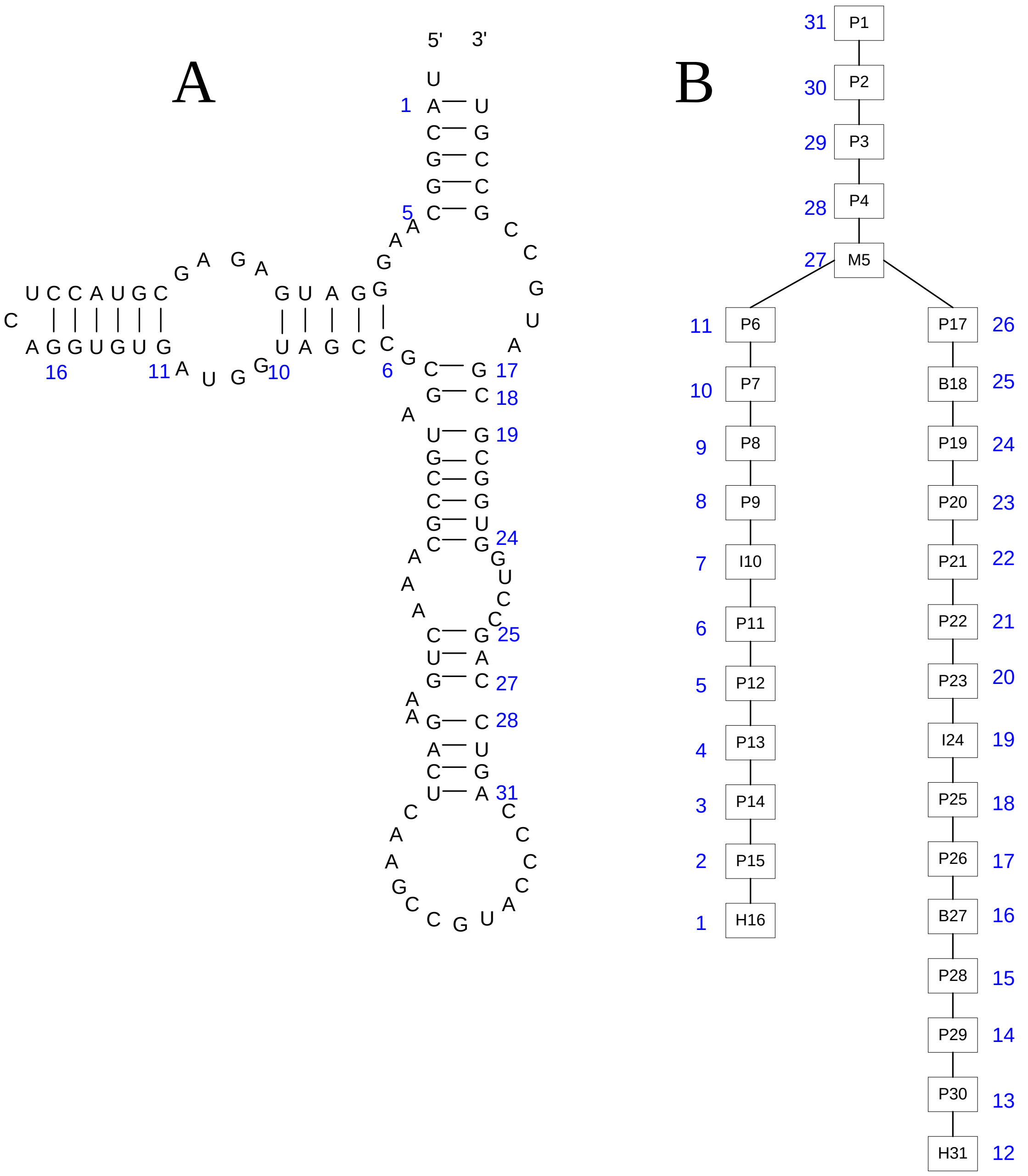


Figure 2.2 Transform an RNA secondary structure to an ordered labeled tree. (A) An RNA secondary structure is comprised of base pairs, which are numbered according to the order from the 5' end to the 3' end of the secondary structure. (B) The base pairs are organized into an ordered labeled tree. Each node in the tree corresponds to a base pair in the secondary structure and vice versa. The numeric value next to each node in the tree is the position of that node in the left-to-right post-order traversal of the tree.

The algorithm, DiscoverR, uses a post-order numbering of nodes in the tree *RT* representing the RNA secondary structure *R*. Let *rt*[*i*] be the node of *RT* whose position in the left-to-right post-order traversal of *RT* is *i*. Referring to the tree *RT* shown in Figure 2.2(B), the numeric value next to each node is the position of that node in the left-to-right post-order traversal of *RT*. Let *RT*[*i*] represent the subtree rooted at *rt*[*i*]. Here, a cut operation on nodes in a tree [26] is introduced. Cutting at node *rt*[*i*] means removing *RT*[*i*] from the tree *RT*, cf. Figure 2.3. A set *S* of nodes of *RT*[*k*] is said to be a set of consistent subtree cuts in *RT*[*k*] if (i) $rt[i] \in S$ implies that *rt*[*i*] is a node in *RT*[*k*], and (ii) $rt[i], rt[j] \in S$ implies that neither is an ancestor of the other in *RT*[*k*]. Intuitively, *S* is the set of all roots of the removed subtrees in *RT*[*k*]. For example, let's consider the nodes labeled P9 and P23 in the tree shown in Figure 2.2(B). Neither node is an ancestor of the other. Thus, the set containing these two nodes is a set of consistent subtree cuts. *Cut*(*RT*, *S*) is used to represent the substructure of *RT* resulted from cutting at all nodes in *S*. Figure 2.3 shows the example of cutting at the node labeled I24. Notice that the substructure *Cut*(*RT*, *S*) is connected at the structure level; that is, if two nodes in *RT* are contained in the substructure such that one node is an ancestor of the other node, then all nodes in between the two nodes are also contained in the substructure. For example, cutting at the nodes labeled P9 and P23 in the secondary structure in Figure 2.2 (B) yields the substructure shown in Figure 2.4, which is connected at the structure level. *Subtrees*(*RT*) is used to represent the set of all possible sets of consistent subtree cuts in *RT* [27].

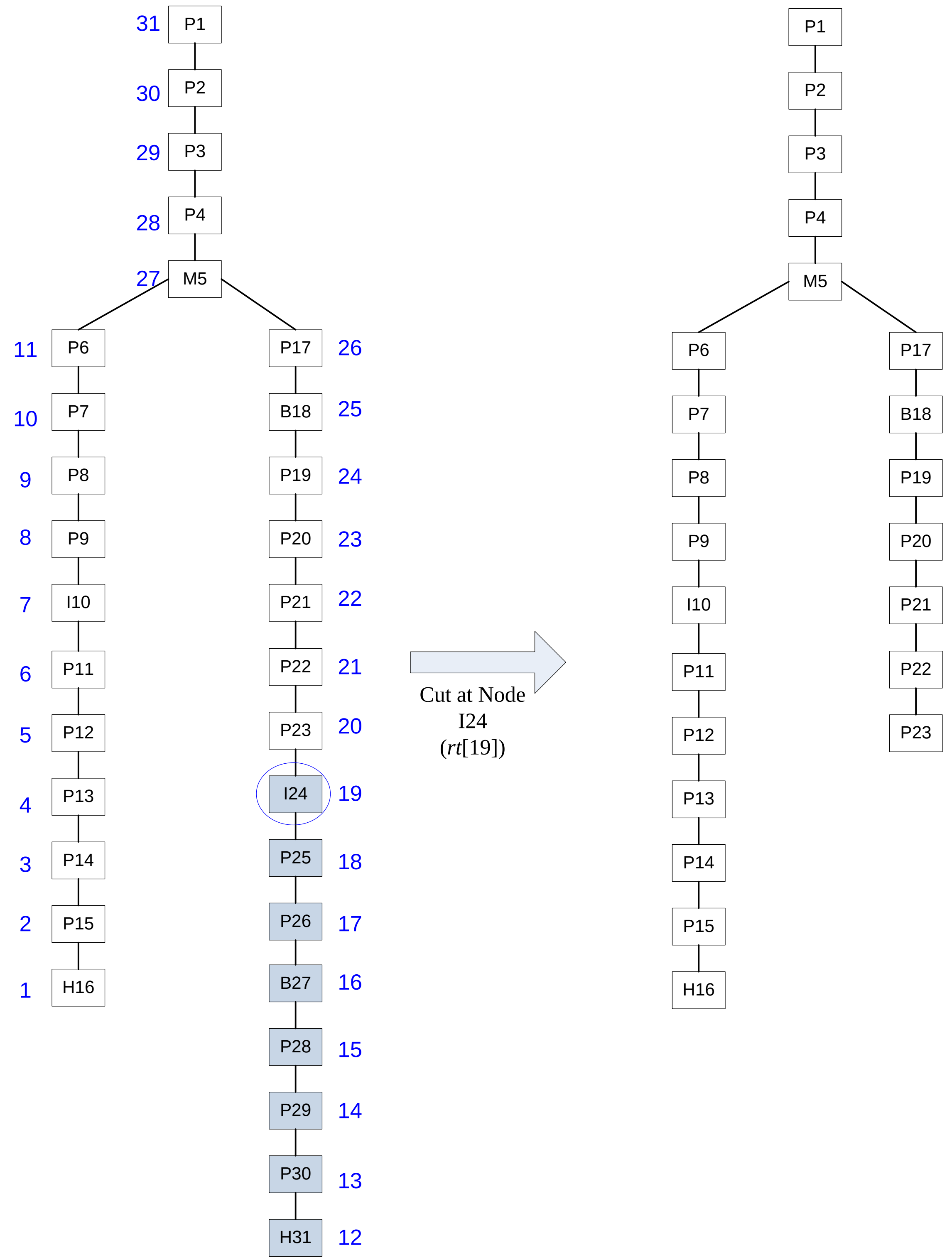


Figure 2.3 Cutting at the node labeled I24 (rt[19]) means removing the subtree rooted at the node labeled I24.

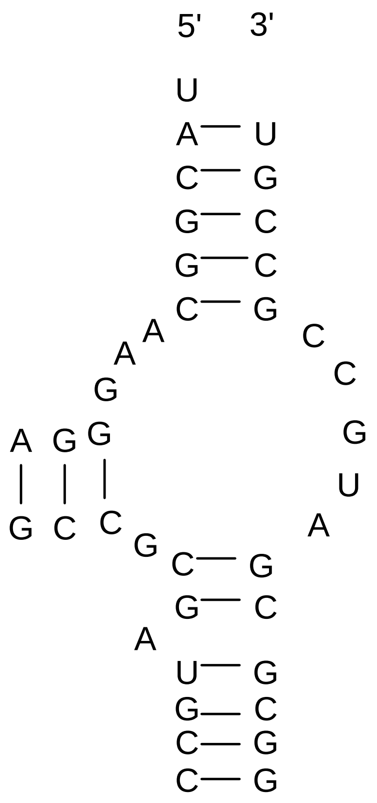


Figure 2.4 The substructure obtained by cutting at the nodes labeled P9 and P23 in the secondary structure in Figure 2.3.

In general, there are stem-loops, with bulges, internal loops, multi-branch loops or pseudoknots. Since a pseudoknot is a secondary structure containing at least two stem-loop structures in which half of one stem is intercalated between the two halves of another stem [28], pseudoknots are not allowed in DiscoverR.

2.2.2 Common Patterns of Two Trees

Let's consider a scenario where R_1 and R_2 are two RNA secondary structures, RT_1 (RT_2, respectively) is the tree representing R_1 (R_2, respectively), rt_1 is a node in RT_1, rt_2 is a node

]]>

in RT_2. The dissimilarity between the two nodes rt_1 and rt_2, denoted $\delta(rt_1, rt_2)$, is calculated by Formula (2.1):

$$\delta(rt_1, rt_2) = \frac{|NB(rt_1) - NB(rt_2)|}{NB(rt_1) + NB(rt_2)} \tag{2.1}$$

$\delta(rt_1, rt_2)$ equals 0 if rt_1 and rt_2 have the same number of bases. Node rt_1 matches node rt_2, denoted $rt_1 \approx \mathrm{rt}_2$, if $\delta(rt_1, rt_2) \leq \varepsilon$ where ε is an adjustable non-negative threshold value. (In the study presented here, the default threshold value is used, which is set to 0.1.) When rt_1 (rt_2, respectively) corresponds to a base pair, rt_1 always matches rt_2, since $\delta(rt_1, rt_2)$ equals 0. We say tree RT_1 matches tree RT_2, denoted $RT_1 \approx RT_2$, if the two trees are isomorphic and each node in RT_1 matches its corresponding node in RT_2.

The size of the largest common substructures or *common patterns* of $RT_1[i]$ and $RT_2[j]$, denoted $\Psi(RT_1[i], RT_2[j])$ (or simply $\Psi(i, j)$ when the context is clear), is $\max\{|Cut(RT_1[i], S_i)|\}$ or $\max\{|Cut(RT_2[j], S_j)|\}$ subject to $Cut(RT_1[i], S_i) \approx Cut(RT_2[j], S_\mathrm{j})$, $S_i \in Subtrees(RT_1[i])$, $S_j \in Subtrees(RT_2[j])$, where $|.|$ is the number of nodes in the indicated substructure. It should be pointed out that $Cut(RT_1[i], S_i)$ is isomorphic to $Cut(RT_2[j], S_j)$, therefore $|Cut(RT_1[i], S_i)| = |Cut(RT_2[j], S_j)|$. The goal is to calculate $\max_{1 \leq i \leq |RT_1|,\ 1 \leq j \leq |RT_2|}\{\Psi(RT_1[i], RT_2[j])\}$ and locate the $Cut(RT_1[i], S_i)$ and $Cut(RT_2[j], S_j)$, where $S_i \in Subtrees(RT_1[i])$ and $S_j \in Subtrees(RT_2[j])$, achieve the maximum size. By memorizing the size information during the computation and applying backtracking technique, one can find the maximum size and a substructure pair yielding the size with the same time complexity.

2.2.3 Common Patterns of Two Forests

The degree of a node v is defined as the number of children of v. Suppose the degree of the node $rt_1[i]$ ($rt_2[j]$, respectively) in the tree RT_1 (RT_2, respectively) is m_i (n_j, respectively). Denote the children of $rt_1[i]$ as $rt_1[i_1]$, $rt_1[i_2]$, … , $rt_1[i_{m_i}]$, and the children of $rt_2[j]$ as $rt_2[j_1]$, $rt_2[j_2]$, … , $rt_2[j_{n_j}]$. For any p, q, $1 \leq p \leq q \leq m_i$, let $RF_1[i_p, i_q]$ represent the forest containing the subtrees $RT_1[i_p]$, $RT_1[i_{p+1}]$, … , $RT_1[i_q]$. $RF_1[i_p, i_q] = \phi$ if $p > q$, and $RF_1[i_p, i_q] = RT_1[i_p]$ if $p = q$. $RF_1[i] = RF_1[i_1, i_{m_i}]$. $RF_2[j_s, j_t]$, $1 \leq s \leq t \leq n_j$, and $RF_2[j]$ are defined similarly. We can say forest RF_1 matches forest RF_2, denoted $RF_1 \approx RF_2$, if the two forests are isomorphic and each node in RF_1 matches its corresponding node in RF_2.

A set S of nodes of forest RF is considered to be a set of consistent subtree cuts in RF if (i) $rt[i] \in S$ implies that $rt[i]$ is a node in RF, and (ii) $rt[i], rt[j] \in S$ implies that neither is an ancestor of the other in RF. $Cut(RF, S)$ is used to represent the subforest of RF resulted from cutting at all nodes in S. Let $Subtrees(RF)$ be the set of all possible sets of consistent subtree cuts in RF. Define the size of the largest common substructures or common patterns of forest RF_1 and forest RF_2, denoted $\Phi(RF_1, RF_2)$, to be $\max\{|Cut(RF_1, S_1)|\}$ or $\max\{|Cut(RF_2, S_2)|\}$ subject to $Cut(RF_1, S_1) \approx Cut(RF_2, S_2)$, $S_1 \in Subtrees(RF_1)$, $S_2 \in Subtrees(RF_2)$. When $RF_1 = RF_1[i_p, i_q]$ and $RF_2 = RF_2[j_s, j_t]$, $\Phi(RF_1, RF_2)$ is also represented by $\Phi(i_p..i_q, j_s..j_t)$ if there is no confusion.

2.2.4 Filling in the Maximum Size Table

It is clear that $\Psi(\emptyset,\emptyset) = 0$, $\Phi(\emptyset,\emptyset) = 0$, $\Psi(RT_1[i],\emptyset) = 0$, $\Psi(\emptyset, RT_2[j]) = 0$, $\Phi(RF_1[i],\emptyset) = \Phi(RF_1[i_1, i_{m_i}],\emptyset) = 0$, and $\Phi(\emptyset, RF_2[j]) = \Phi\left(\emptyset, RF_2\left[j_1, j_{n_j}\right]\right) = 0$, i.e. the size of the common patterns of two trees (forests, respectively) is 0 if one of the trees (forests, respectively) is empty [27]. In general, there are two cases to be considered. In case 1, $\Phi(RF_1[i_1, i_q], RF_2[j_1, j_t])$ is computed, where $1 \leq q \leq m_i$ and $1 \leq t \leq n_j$. There are three subcases (Figure 2.5):

(1) The subtree $RT_1[i_q]$ is removed, hence $\Phi(i_1..i_q, j_1..j_t) = \Phi(i_1..i_{q-1}, j_1..j_t)$.

(2) The subtree $RT_2[j_t]$ is removed, hence $\Phi(i_1..i_q, j_1..j_t) = \Phi(i_1..i_q, j_1..j_{t-1})$.

(3) Neither $RT_1[i_q]$ nor $RT_2[j_t]$ is removed. Hence, the size of the common patterns of $RF_1[i_1, i_q]$ and $RF_2[j_1, j_t]$ equals the size of the common patterns of $RF_1[i_1, i_{q-1}]$ and $RF_2[j_1, j_{t-1}]$ plus the size of the common patterns of $RT_1[i_q]$ and $RT_2[j_t]$.

The following recurrence formula is for the three subcases, and the maximum of them will be taken:

$$\Phi(i_1..\, i_q, j_1..\, j_t) = max\begin{cases}\Phi(i_1..\, i_{q-1}, j_1..j_t) \\ \Phi(i_1..\, i_q, j_1..j_{t-1}) \\ \Phi(i_1..\, i_{q-1}, j_1..j_{t-1}) + \Psi(i_q, j_t)\end{cases} \tag{2.2}$$

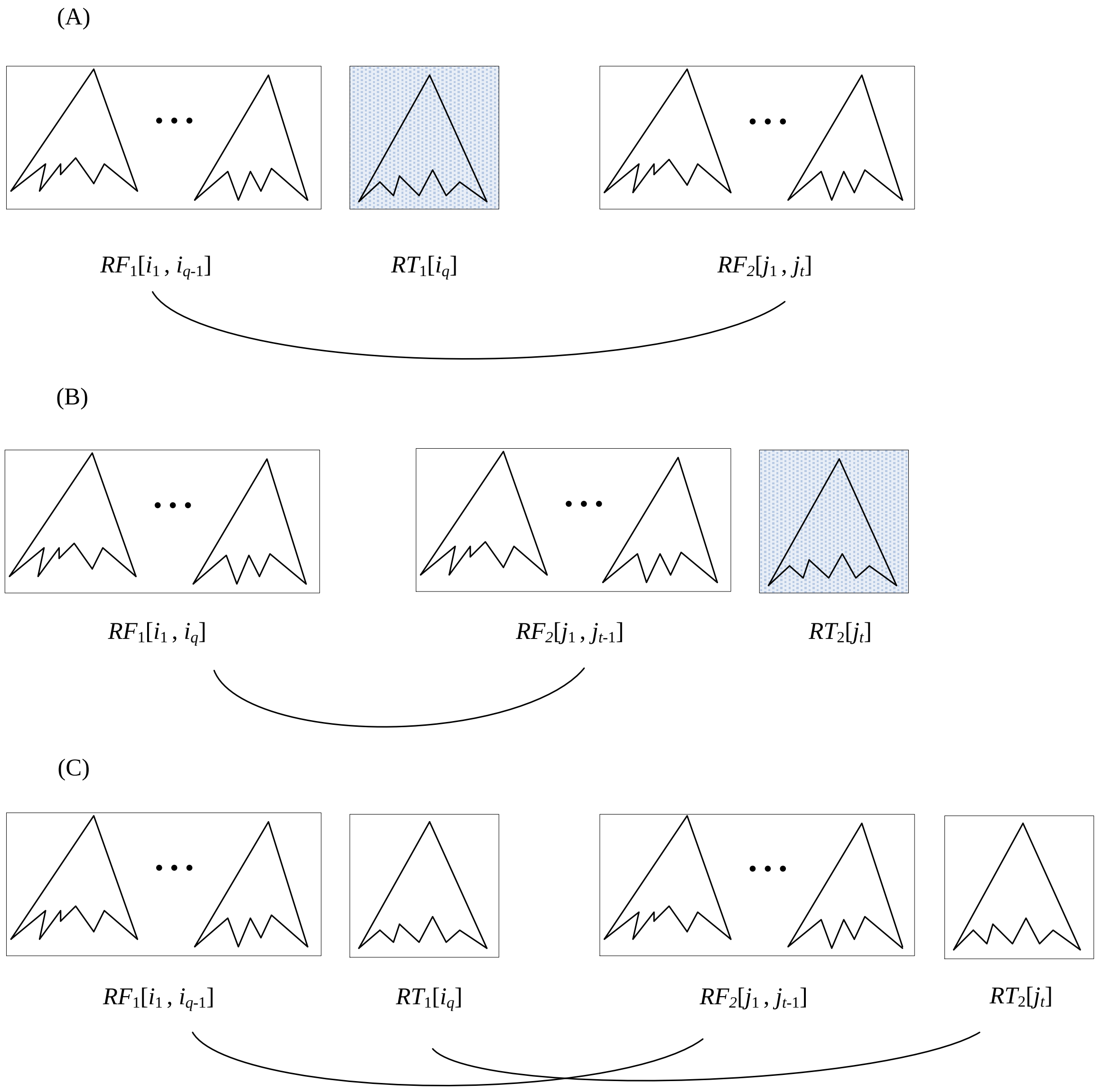


Figure 2.5 (A) The shaded subtree $RT_1[i_q]$ is removed. The size of the common patterns of forest $RF_1[i_1, i_q]$ and forest $RF_2[j_1, j_t]$ is the same as the size of the common patterns of forest $RF_1[i_1, i_{q-1}]$ and forest $RF_2[j_1, j_t]$. (B) The shaded subtree $RT_2[j_t]$ is removed. The size of the common patterns of forest $RF_1[i_1, i_q]$ and forest $RF_2[j_1, j_t]$ is the same as the size of the common patterns of forest $RF_1[i_1, i_q]$ and forest $RF_2[j_1, j_{t-1}]$. (C) Neither $RT_1[i_q]$ nor $RT_2[j_t]$ is removed. The size of the common patterns of forest $RF_1[i_1, i_q]$ and forest $RF_2[j_1, j_t]$ equals the size of the common patterns of $RF_1[i_1, i_{q-1}]$ and forest $RF_2[j_1, j_{t-1}]$ plus the size of the common patterns of tree $RT_1[i_q]$ and tree $RT_2[j_t]$.

In case 2, $\Psi(RT_1[i], RT_2[j])$, $1 \le i \le |RT_1|$, $1 \le j \le |RT_2|$, is computed. There are two subcases need consideration:

(1) The node $rt_1[i]$ matches the node $rt_2[j]$, hence $\Psi(i, j) = \Phi(i_1.. i_{m_i}, j_1..j_{n_j}) + 1$ (Figure 2.6).

(2) The node $rt_1[i]$ does not match the node $rt_2[j]$, hence $\Psi(i, j) = 0$. Therefore,

$$\Psi(i, j) = \begin{cases} \Phi(i_1.. i_{m_i}, j_1..j_{n_j}) + 1 & \text{if } rt_1[i] \approx rt_2[j] \\ 0 & \text{otherwise} \end{cases} \qquad (2.3)$$

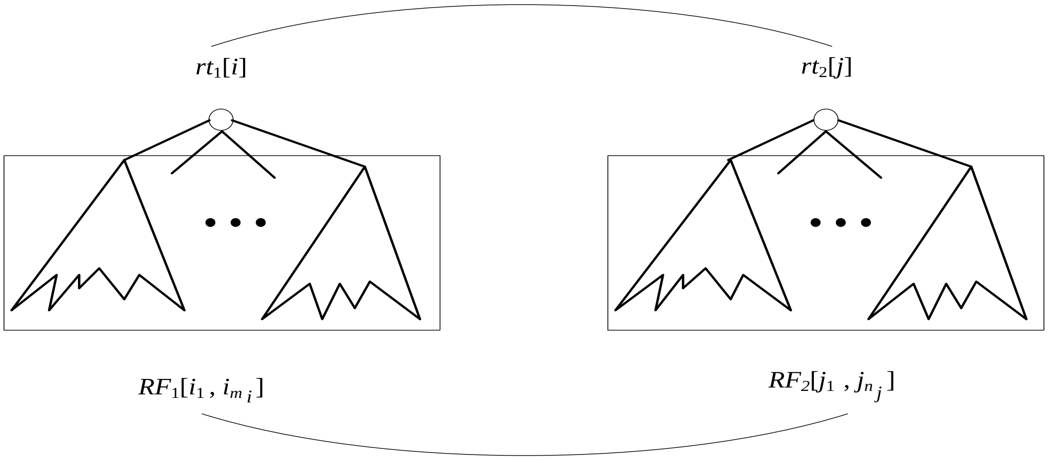


Figure 2.6 The node $rt_1[i]$ matches the node $rt_2[j]$. Thus, the size of the common patterns of tree $RT_1[i]$ and tree $RT_2[j]$ equals the size of the common patterns of forest $RF_1[i_1, i_{m_i}]$ and forest $RF_2[j_1, j_{n_j}]$ plus 1.

2.2.5 Algorithm Complexity

DiscoverR employs a dynamic programming algorithm that maintains a two-dimensional table in which $c(i, j)$ represents the cell located at the intersection of the ith row and the jth column of the table. The value stored in the cell $c(i, j)$, $1 \leq i \leq |RT_1|$, $1 \leq j \leq |RT_2|$, is $\Psi(RT_1[i], RT_2[j])$. This algorithm calculates the values in the table by traversing the trees RT_1 and RT_2 in a bottom-up manner. Figures 2.7 and 2.8 present the main procedures employed. For each input $RF1[i_1, i_{m_i}]$ and $RF2[j_1, j_{n_j}]$, the running time of Procedure 1 is $O(m_i \times n_j)$. Thus, the time complexity of the algorithm is

$$\sum_{i=1}^{|RT_1|}\sum_{j=1}^{|RT_2|} O(m_i \times n_j) = O(|RT_1| \times |RT_2|) \tag{2.4}$$

DiscoverR is much faster than the existing algorithm for general approximate tree pattern discovery [23]. $\max_{1\le i\le|RT_1|,\ 1\le j\le|RT_2|}\{\Psi(RT_1[i], RT_2[j])\}$ can be calculated in the same time. Since the number of nodes in the tree RT_1 (RT_2, respectively) equals the number of base pairs in the RNA secondary structure R_1 (R_2, respectively), and 54% of the nucleotides on average in an RNA sequence are involved in the base pairs of its secondary structure[29], the time complexity of the DiscoverR algorithm is $O(|R_1| \times |R_2|)$ where |.| is the number of nucleotides in the indicated RNA secondary structure. After calculating and locating the largest common substructures or common patterns of RT_1 and RT_2 that yield the maximum size, the common patterns are printed out, which constitute the output of DiscoverR.

```
Procedure 1: Computing $\Phi(i_1..\ i_{m_i}, j_1..j_{n_j})$

Input: $RF_1[i_1, i_{m_i}]$ and $RF_2[j_1, j_{n_j}]$

Output: $\Phi(i_1..\ i_{m_i}, j_1..j_{n_j})$

1. $\Phi(\phi,\phi) \leftarrow 0$

2. for $q$ := 1 to $m_i$

3.       $\Phi(i_1..\ i_q, \phi) \leftarrow 0$

4. for $t$ := 1 to $n_j$
```

Figure 2.7 Procedure for computing $\Phi(i_1..\ i_{m_i}, j_1..j_{n_j})$.

m_i

n_j

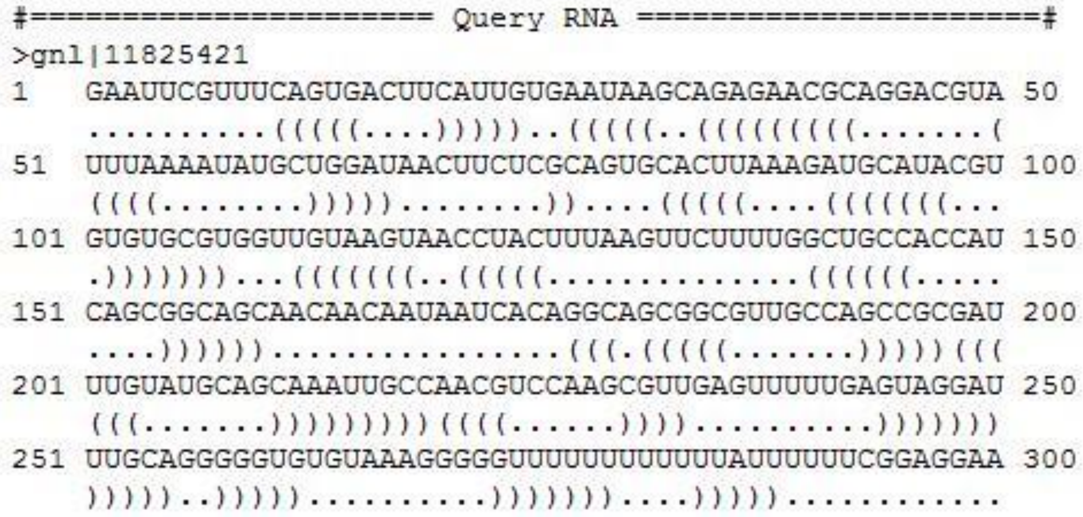
```
Procedure 2: Computing $\Psi(i, j)$ for all $1 \le i \le |RT_1|, 1 \le j \le |RT_2|$

Input: $RT_1$ and $RT_2$

Output: $\Psi(i, j)$ for all $1 \le i \le |RT_1|, 1 \le j \le |RT_2|$

$\Psi(\phi, \phi) \leftarrow 0$
for $i$ := 1 to $|RT_1|$
    $\Psi(RT_1[i], \phi) \leftarrow 0$
for $j$ := 1 to $|RT_2|$
    $\Psi(\phi, RT_2[j]) \leftarrow 0$
for $i$ := 1 to $|RT_1|$
    for $j$ := 1 to $|RT_2|$
        Compute $\Psi(i, j)$ as in Formula (3)
          by calling Procedure 1 on $RF_1[i_1, i_{m_i}]$ and $RF_2[j_1, j_{n_j}]$
```

Figure 2.8 Procedure for computing $\Psi(i, j)$ for all $1 \le i \le |RT_1|, 1 \le j \le |RT_2|$.

2.3 Program of DiscoverR

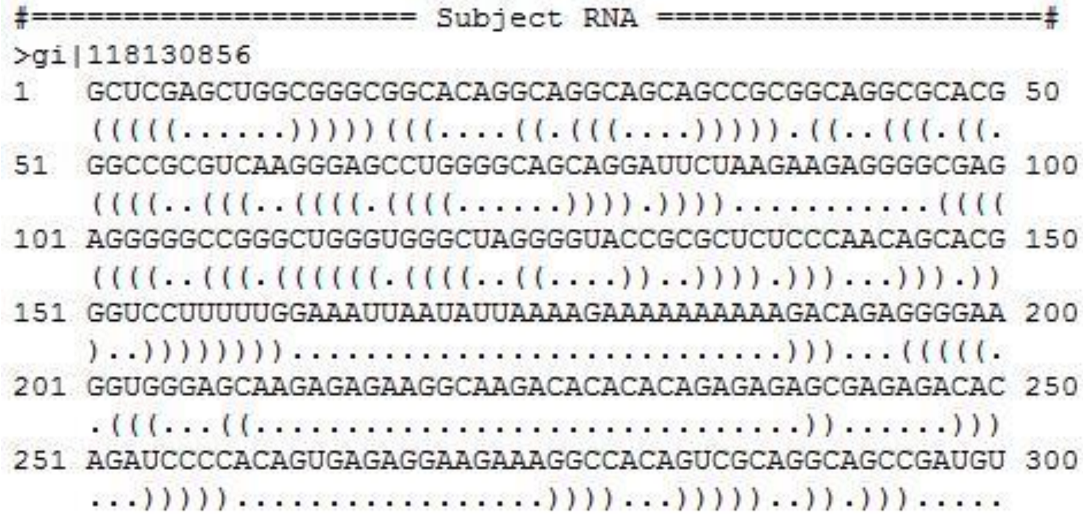
The DiscoverR program is implemented in Java. The jar file including the source code of the program is available for download at http://bioinformatics.njit.edu/DiscoverR. Figure 2.9 shows the two RNA secondary structures as in input in the output of the DiscoverR program and Figure 2.10 shows the two patterns given by the DiscoverR program. The beginning and ending positions of contiguous bases on the common patterns in two input structures are printed out. In Figure 2.11 and Figure 2.12, tow RNA secondary structures are portrayed using RnaViz 2 [12], where the common patterns of the two input structures found by DiscoverR are highlighted in blue.

(A)



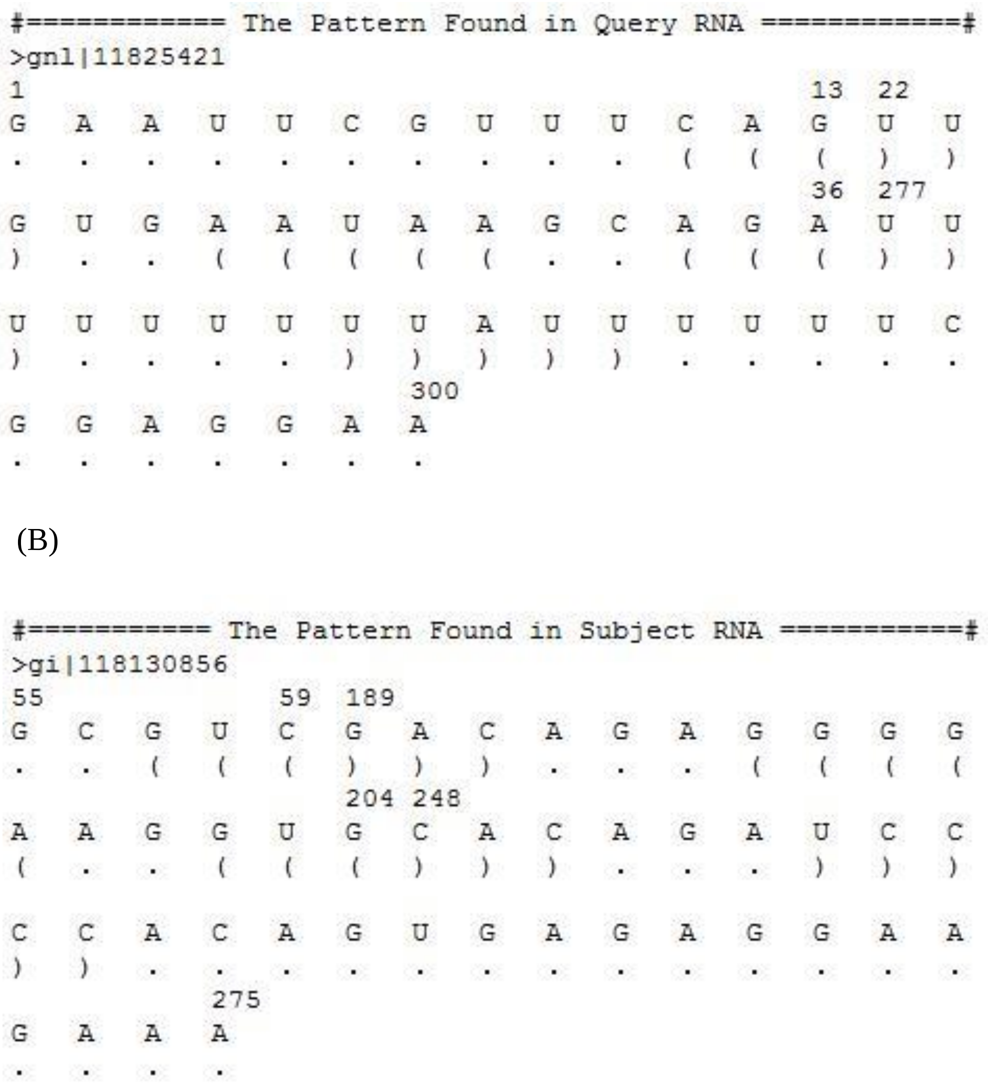
```
#===================== Query RNA =====================#
>gnl|11825421
1   GAAUUCGUUUCAGUGACUUCAUUGUGAAUAAGCAGAGAACGCAGGACGUA 50
    ..........(((((....)))))..(((((..(((((((((.......(
51  UUUAAAAUAUGCUGGAUAACUUCUCGCAGUGCACUUAAAGAUGCAUACGU 100
    ((((........)))))........))....(((((....(((((((...
101 GUGUGCGUGGUUGUAAGUAACCUACUUUAAGUUCUUUUGGCUGCCACCAU 150
    .)))))))...(((((((..(((((..............((((((.....
151 CAGCGGCAGCAACAACAAUAAUCACAGGCAGCGGCGUUGCCAGCCGCGAU 200
    ....))))))................(((.(((((.......)))))(((
201 UUGUAUGCAGCAAAUUGCCAACGUCCAAGCGUUGAGUUUUUGAGUAGGAU 250
    (((.......)))))))))((((......))))..........)))))))
251 UUGCAGGGGGUGUGUAAAGGGGGUUUUUUUUUUUAUUUUUUCGGAGGAA 300
    )))))..)))))..........)))))))....)))))............
```

(B)



```
#===================== Subject RNA =====================#
>gi|118130856
1   GCUCGAGCUGGCGGGCGGCACAGGCAGGCAGCAGCCGCGGCAGGCGCACG 50
    (((((......)))))(((....((.(((....))))).((..(((.((.
51  GGCCGCGUCAAGGGAGCCUGGGGCAGCAGGAUUCUAAGAAGAGGGGCGAG 100
    ((((..(((..((((.((((......)))).))))...........((((
101 AGGGGGCCGGGCUGGGUGGGCUAGGGGUACCGCGCUCUCCCAACAGCACG 150
    ((((..(((.((((((.((((..((....))..)))).)))...))).))
151 GGUCCUUUUUGGAAAUUAAUAUUAAAAGAAAAAAAAAAAGACAGAGGGGAA 200
    )..))))))))...........................)))...(((((.
201 GGUGGGAGCAAGAGAGAAGGCAAGACACACACAGAGAGAGCGAGAGACAC 250
    .(((...((..............................))......)))
251 AGAUCCCCACAGUGAGAGGAAGAAAGGCCACAGUCGCAGGCAGCCGAUGU 300
    ...)))))........................))))...)))))..)).))).....
```

Figure 2. 9 (A) A query RNA. (B) A subject RNA.



(A)

```
#============ The Pattern Found in Query RNA ============#
>gnl|11825421
1                                                           13   22
G    A    A    U    U    C    G    U    U    U    C    A    G    U    U
.    .    .    .    .    .    .    .    .    .    (    (    (    )    )
                                                            36   277
G    U    G    A    A    U    A    A    G    C    A    G    A    U    U
)    .    .    (    (    (    (    (    .    .    (    (    (    )    )

U    U    U    U    U    U    U    A    U    U    U    U    U    U    C
)    .    .    .    .    )    )    )    )    )    .    .    .    .    .
                              300
G    G    A    G    G    A    A
.    .    .    .    .    .    .
```

(B)

```
#=========== The Pattern Found in Subject RNA ===========#
>gi|118130856
55                  59   189
G    C    G    U    C    G    A    C    A    G    A    G    G    G    G
.    .    (    (    (    )    )    )    .    .    .    (    (    (    (
                         204  248
A    A    G    G    U    G    C    A    C    A    G    A    U    C    C
(    .    .    (    (    (    )    )    )    .    .    .    )    )    )

C    C    A    C    A    G    U    G    A    G    A    G    G    A    A
)    )    .    .    .    .    .    .    .    .    .    .    .    .    .
               275
G    A    A    A
.    .    .    .
```

Figure 2.10 (A) The pattern found in the query RNA. (B) The pattern found in the subject RNA. In (A), (B), beginning and ending positions of the contiguous bases on the common patterns found in the query RNA and subject RNA are displayed.

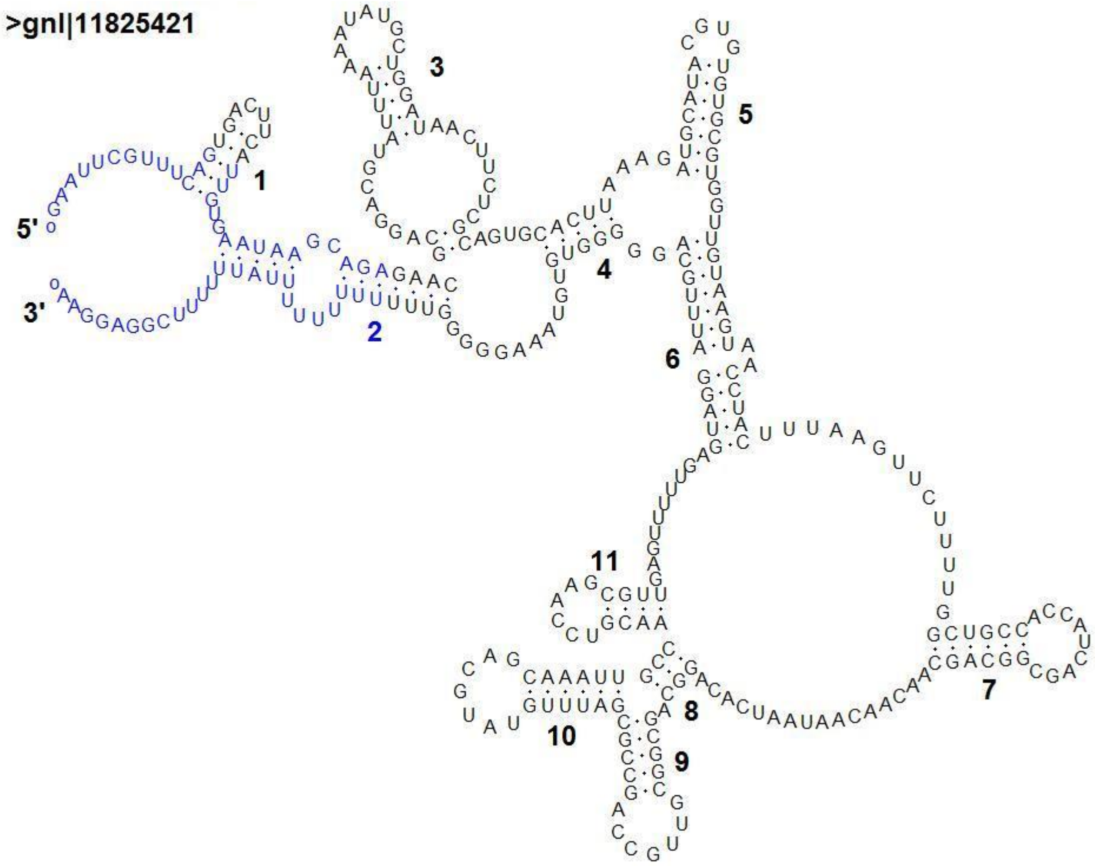


Figure 2.11 Common pattern found by DiscoverR in the RNA molecule gnl|11825421 is highlighted in blue.

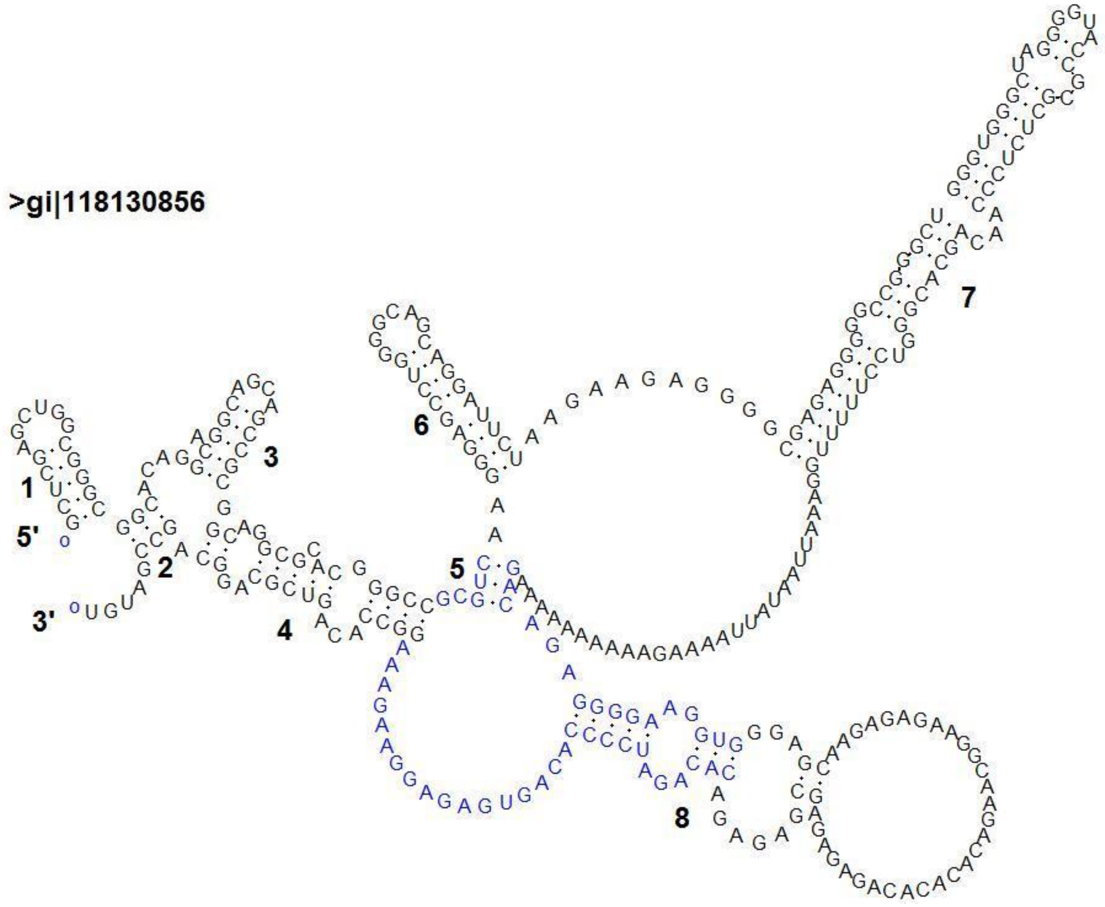


Figure 2.12 Common pattern found by DiscoverR in the RNA molecules gi|118130856 is highlighted in blue.

2.4 Comparison with Related Works

Backofen and Siebert[30] presented an algorithm for finding common sequence structure patterns between two RNAs. These common patterns share the same local sequential and structural properties. Like DiscoverR, the patterns found by Backofen and Siebert are connected at the structure level (whose definition is given in Section 2.2.1). In addition, the patterns found by Backofen and Siebert are also connected at the sequence level, meaning that for any two nodes in a common substructure, there is a matched path via backbone or structure bonds that connects the two nodes. Their algorithm is useful in detecting local regions of large RNAs that do not share global similarities. The time complexity of their algorithm is $O(m \times n)$, where m and n are the lengths of the two input RNAs, respectively.

Höchsmann *et al.*[31] developed another approach for detecting local similarities in RNA secondary structures. They treated RNA secondary structures as forests and gave a dynamic programming algorithm to calculate local forest alignments. These alignments gave rise to local similar regions in RNA secondary structures. The time complexity of their algorithm is $O(|F_1|\times|F_2|\times deg(F_1)\times deg(F_2)\times(deg(F_1)+deg(F_2)))$, where $|F_i|$ is the number of nodes in forest F_i and $deg(F_i)$ is the degree of F_i. Höchsmann *et al.* showed that their algorithm can discover potential regulatory motifs solely by their structural preservation, independent of their sequence conservation and position.

Mauri and Pavesi[32] employed affix trees to locate patterns in an RNA sequence (secondary structure). The time complexity of their approach is asymptotically $O(n)$ where n is the length of the sequence. Mauri and Pavesi described in detail how to locate hairpins in the input sequence. For more complex RNA motifs, these motifs are firstly decomposed into single hairpins. Their approach then locates all the single hairpins in the sequence. Through post-processing, the complex motifs comprising the hairpins are determined and identified. Due to the use of affix trees, the patterns found by their approach contain contiguous bases in the RNA sequence.

DiscoverR has two major differences and improvements over the above algorithms: (i) the discovered patterns and (ii) the algorithms used to find the patterns. Unlike the patterns found by Backofen and Siebert, which are connected both at the structure level and sequence level, DiscoverR can find the patterns connected at the structure level only. For example, consider the hypothetical RNA secondary structure in Figure 2.13(A). The substructure in Figure 2.13(B), obtained by cutting at the two C-G base pairs as shown in Figure 2.13(A), is a potential pattern that can be found by DiscoverR. However, since this pattern is not connected at the sequence level (e.g. there is no path via backbone or structure bonds connecting the two A-U base pairs circled by dashed lines), the pattern in Figure 2.13(B) cannot be found by Backofen and Siebert's algorithm. Furthermore, since this pattern contains non-contiguous bases (bases between position 24 and position 35 are removed), the pattern in Figure 2.13(B) cannot be located by Mauri and Pavesi's algorithm neither.

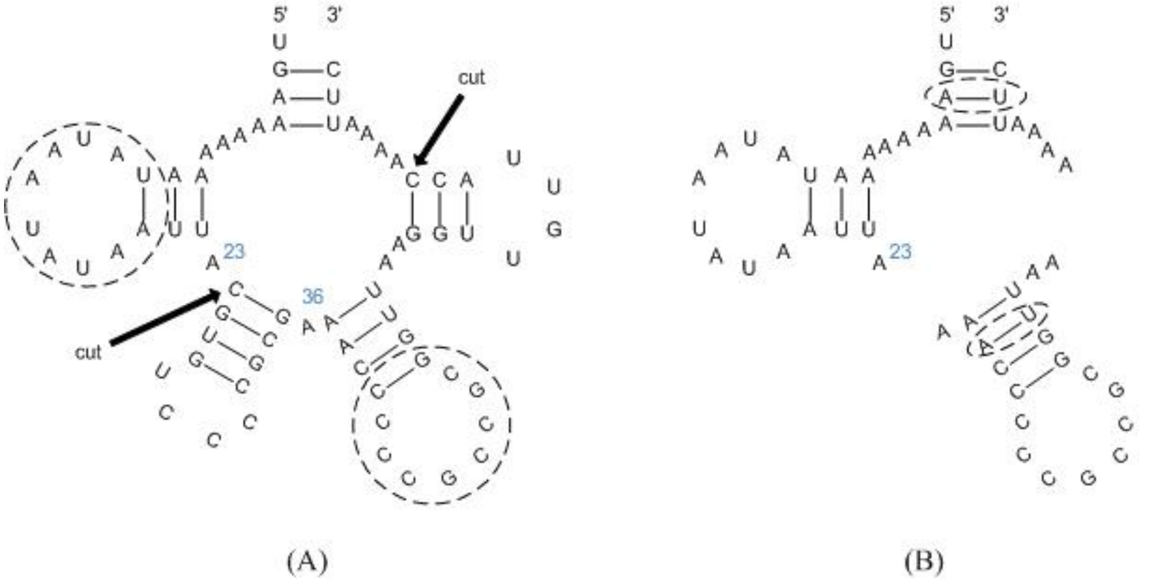


Figure 2.13 Examples illustrating the differences between DiscoverR and related algorithms.

In contrast to the local alignment algorithm developed by Höchsmann *et al.*[31], which seeks small, local regions with high similarity where bases are close to each other, DiscoverR looks at the entire RNA molecules to extract their largest common substructures possibly with distant bases on the respective molecules. For example, consider again the structure in Figure 2.11 (A) and the structure in Figure 2.11(B). When comparing these two structures, DiscoverR can find their common patterns containing the two hairpin loops circled by dashed lines by freely cutting at the two C-G base pairs as shown in Figure 2.11 (A). However, the local alignment algorithm would not identify these patterns due to the penalty incurred in aligning the bases on the stem-loop between position 24 and position 35 in Figure 2.11 (A) with gaps.

To locate the patterns with distant bases, DiscoverR employs cost-free cut operations, which do not exist in the above mentioned algorithms. The only algorithm that also uses cut operations for tree pattern discovery is the algorithm developed in the previous work [23]. That algorithm finds the largest approximately common substructures U_1 and U_2 of two given ordered labeled trees T_1 and T_2, where the substructure U_1 of T_1 is within edit distance d of the substructure U_2 of T_2. The time complexity of that algorithm is $O(d^2 \times |T_1| \times |T_2| \times \min(H_1, L_1) \times \min(H_2, L_2))$, where H_i, $i = 1, 2$, is the height of T_i and L_i is the number of leaves in T_i. In contrast, DiscoverR is a faster algorithm with a time complexity of $O(|T_1| \times |T_2|)$.

CHAPTER 3

APPLICATIONS OF DISCOVERR

In this chapter, two major applications of DiscoverR are presented. The first application is to find repeated regions in an RNA secondary structure. The second application is to find conserved RNA secondary structures in the human.

3.1 Repeats

Repeat finding has been an important subject in bioinformatics and computational biology. Past work has mainly focused on detecting repeats in sequences[33,34]. In contrast, DiscoverR is capable of locating structural repeats or repeated regions in an RNA secondary structure. In this section, we demonstrate show how DiscoverR can be used to find structural repeats in the 3'-untranslated region of a protein kinase gene.

Structural repeats involving trinucleotides such as CUG are present in many genomes and their expansion in specific genes causes neurological disorder or disease[35]. Repeated hairpin structures containing CAG play regulatory roles mediated by their interactions with RNA-binding proteins[36]. These structural repeats are also involved in RNA splicing.

DiscoverR can be easily extended to detect repeated regions in a given RNA secondary structure R by using DiscoverR to compare R with R itself. Thus, both RT_1 and RT_2 as shown in Figure 2.8 correspond to the same structure R. As described before, the algorithm maintains a two-dimensional table in which $c(i, j)$ represents the cell located at the intersection of the ith row and the jth column of the table. The value stored in the cell

$c(i, j)$, $1 \leq i \leq |RT_1|$, $1 \leq j \leq |RT_2|$, is $\Psi(RT_1[i], RT_2[j])$. DiscoverR calculates the values in the table by traversing the trees RT_1 and RT_2 in a bottom-up manner. If the value in the cell $c(i, j)$, $i \neq j$, is greater than or equal to a user-determined size threshold, the two substructures rooted at $rt_1[i]$ and $rt_2[j]$, respectively, which are common patterns of tree $RT_1[i]$ and tree $RT_2[j]$, giving rise to a repeated region or structural repeat in R. (In the study presented here, the size threshold is set to 2.) Figure 3.1 shows a structural repeat, highlighted in blue, that DiscoverR detects in an RNA secondary structure in the 3'-untranslated region (UTR) of the DM protein kinase (DMPK) gene[37]. The repeated hairpin structure in Figure 3.1 forms the genetic basis of myotonic dystrophy[38]. This example efficiently proves DiscoverR's capability in detecting biologically significant structural repeats in RNA molecules.

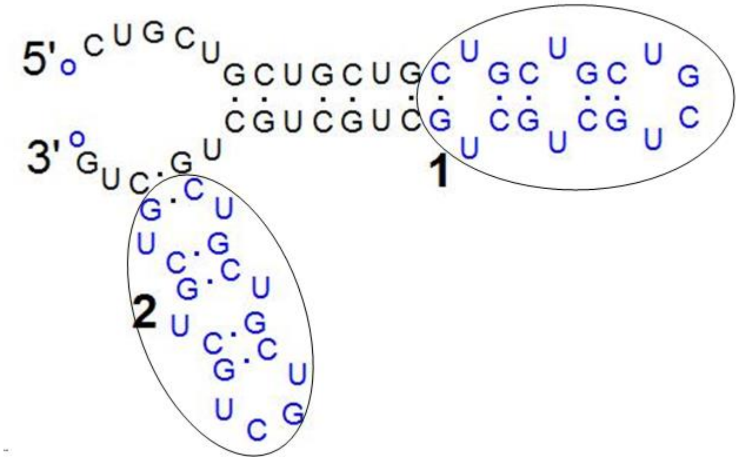


Figure 3.1 Illustration of a structural repeat, circled with solid lines and highlighted in blue, detected in an RNA secondary structure in the 3'-UTR of the DMPK gene.

3.2 Finding Genomic Regions within Conserved Substructures

Using 8-way human-referenced vertebrate genome alignments, Washietl *et al*. [21] detected 91,676 conserved RNA structures (at P > 0.5) using the RNAz program, which identified RNA structures with similar thermodynamic stabilities across multiple species. Pedersen *et al*. [39] developed a phylogenetics-based stochastic context-free grammar (phylo-SCFG), and identified 48,479 candidate RNA structures using the same genome alignments. Torarinsson *et al*.[40] focused on human and mouse genomic sequences that could not be aligned on the sequence level, and identified conserved structures by OLDALIGN surveyed in the Related Work section. Khaladkar *et al.*[22] developed a clustering-based approach, named GLEAN-UTR, to identify stem-loop RNA structure elements in untranslated regions (UTRs) that were conserved between human and mouse orthologs, and existed in multiple genes with common Gene Ontology terms. For the 10,448 human genes that were analyzed, Khaladkar *et al*. obtained 90 RNA structure groups, containing 748 distinct RNA structures in 5' or 3' UTRs from 698 genes.

We began with 130 conserved human RNA structures each having at least 14 bases identified by GLEAN-UTR that were found to be overlapping with the conserved structures detected by Washietl *et al*. and Pedersen *et al*. (Figure 4 and Additional file 4 in [22]). The structures predicted by Torarinsson *et al*. [40] did not overlap with these 130 RNA structures. The genomic regions of these 130 RNA structures [41] are located, and mapped to the 8-way human-referenced (hg17) vertebrate genome alignments available at the UCSC Genome Browser (http://genome.ucsc.edu/). The 8-way genome alignments are selected, that fully contained the genomic regions of the RNA structures (if a structure straddled two different genome alignments, that structure was excluded) [42]. Some of the selected genome alignments were long, with several thousand nucleotides. A sub-alignment or alignment block are extracted from each selected genome alignment where the length of an alignment block was *Ln* and each alignment block fully contained the genomic region of at least one structure listed in Additional file 4 in [22]. (In the study presented here, *Ln* was set to 300.) If the length of a selected genome alignment was less than *Ln*, that whole genome alignment was treated as an alignment block. This step resulted in 102 alignment blocks where each alignment block had 4 to 8 sequences (species).

We subsequently designed a systematic approach to detect conserved human structures using DiscoverR. For each alignment block *B*, after removing gaps in it, a set S_B of 8 or fewer sequences for that alignment block are obtained. Using the Vienna RNA Package [9], each sequence in S_B is folded to get its minimum-energy secondary structure, also placed in S_B. The human structure, *H*, is compared with each of the other structures, *R*, in S_B using DiscoverR (with $\varepsilon = 0.1$). Specifically, for the tree *HT* representing *H* and the tree *RT* representing *R*, the largest common substructures of $HT[i]$ and $RT[j]$, for all $1 \leq i \leq |HT|$ and $1 \leq j \leq |RT|$, are found. The discovered patterns or substructures of the human structure *H* were stored in a list, denoted *List*. Each substructure in *List* has at least 14 bases as in [22]; substructures with less than 14 bases were excluded from *List*. What we identified is those human substructures in *List* occurred in at least *Occur* secondary structures in S_B. (In the study presented here, *Occur* was set to 6.) If the number of secondary structures in S_B was less than *Occur*, no substructure in *List* qualified to be a solution. Here a solution was a conserved human substructure that occurred in at least *Occur* species and had at least 14 bases. This step results in 577 qualified substructures. Among the 577 found substructures, some were substructures of others; these subpatterns were eliminated from further consideration. Within the remaining qualified substructures, there were 56 genomic regions each having at least 14 contiguous bases (short regions with less than 14 bases were not considered as in [22]. This structure mining algorithm is illustrated in Figure 3.2. The genomic regions within the conserved human substructures found by our approach are listed in Table 3.1. It can be seen from Table 3.1 that some of the conserved human substructures found by our approach overlap with the known structures detected by the existing algorithms (K for GLEAN-UTR [22], P for Pedersen *et al*. [39] and W for Washietl *et al*. [21], while others are novel ones that are not identified previously.

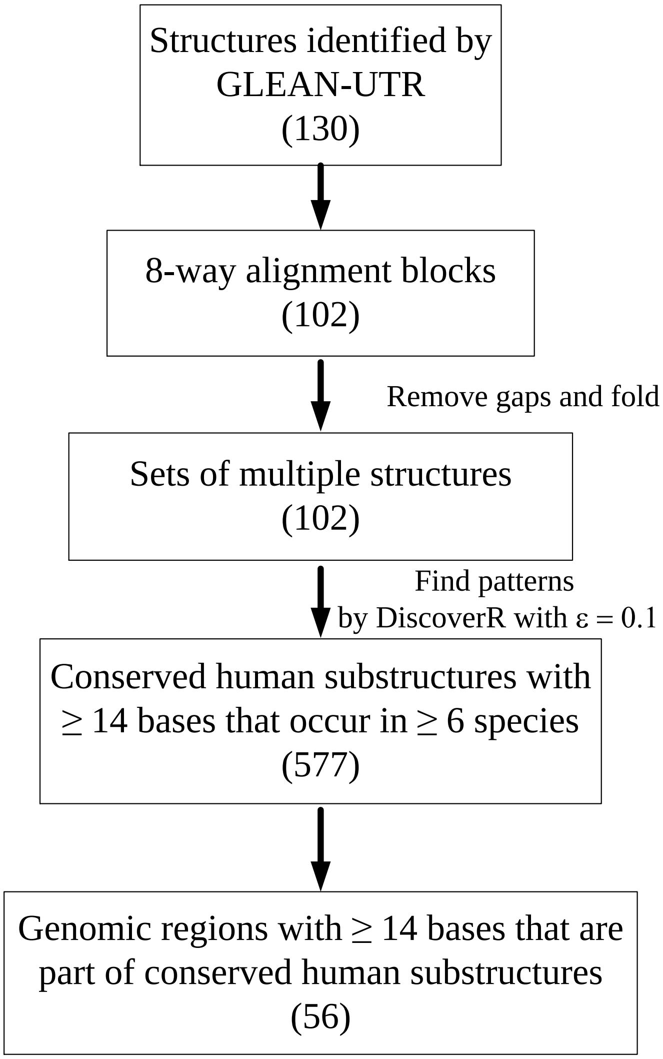

Structures identified by GLEAN-UTR
(130)
8-way alignment blocks
(102)
Remove gaps and fold
Sets of multiple structures
(102)
Find patterns
by DiscoverR with ε = 0.1
Conserved human substructures with ≥ 14 bases that occur in ≥ 6 species
(577)
Genomic regions with ≥ 14 bases that are part of conserved human substructures
(56)

Figure 3.2 Illustration of the flowchart of our approach for mining conserved structural RNAs in the human genome.

Table 3.1 Results of the Experiments Performed in this Study

Genomic regions within conserved human substructures that occur in 8 species

Chromosome	Start Position	Length	Strand	Overlap with
Chr6	26265302	14	+	P
ChrX	106763864	27	-	K, P

Genomic regions within conserved human substructures that occur in 7 species

Chromosome	Start Position	Length	Strand	Overlap with
Chr1	96992213	19	+	W
Chr2	144979616	17	-	-
Chr2	144979712	17	-	K, P
Chr3	197265646	23	-	P
Chr3	197265693	23	-	-
Chr3	197265758	23	-	K, P
Chr3	197266161	25	-	K, P
Chr3	197266211	25	-	K, P
Chr6	19947838	20	+	K, W, P
Chr6	19947871	21	+	K, W
Chr6	168884945	15	+	K, P
Chr14	28308288	16	+	-
Chr17	59926225	18	-	-
ChrX	106763863	29	-	K, P

Table 3.1 (Continued) Results of the Experiments Performed in this Study

Genomic regions within conserved human substructures that occur in 6 species

Chromosome	Start Position	Length	Strand	Overlap with
Chr1	8006261	15	-	-
Chr1	96991635	15	+	W
Chr1	96991697	18	+	W
Chr1	96992209	27	+	K, W
Chr2	14726767	16	+	W
Chr2	144979653	18	-	P
Chr2	144979710	21	-	K, P
Chr2	144979850	16	-	-
Chr2	190270896	21	-	K, P
Chr3	37835407	14	+	-
Chr3	37835524	14	+	-
Chr3	161701103	17	-	K, P
Chr3	161701244	17	-	P
Chr3	197265645	25	-	P
Chr3	197265757	25	-	K, P
Chr3	197266160	27	-	K, P
Chr3	197266210	27	-	K, P
Chr5	179136573	17	+	P, W
Chr5	179136607	17	+	W

Table 3.1 (Continued) Results of the Experiments Performed in this Study

Chr5	179136709	16	+	-
Chr6	134532540	16	-	K, P
Chr7	38196970	16	-	K, P
Chr7	77229459	15	+	W
Chr7	77229519	33	+	K, P, W
Chr7	101486144	16	+	K, P
Chr7	101486165	14	+	W
Chr8	117927555	16	-	-
Chr8	117927581	22	-	-
Chr8	136728826	16	+	W
Chr9	89160953	15	+	K, P, W
Chr10	30790413	19	+	K, W, P
Chr10	119298963	14	+	-
Chr10	119298984	17	+	P
Chr14	28308435	17	+	W, P
Chr14	53964017	14	-	-
Chr14	53964115	22	-	-
Chr17	59926370	23	-	-
Chr19	1386284	25	+	P
Chr19	39410717	38	+	K, P, W
Chr19	39410811	20	+	W

3.3 Conclusions

In practice, DiscoverR is computationally efficient. It is mainly based on a quadratic-time dynamic programming algorithm, which makes it a suitable tool for pattern mining in RNAs. Among many potential applications suitable for DiscoverR, we presented two in this chapter:

1) Repeated regions finding in an RNA secondary structure. Past work has mainly focused on detecting repeats in sequences (Sokol 2007), (Wexler 2005). In contrast, DiscoverR is capable of locating structural repeats or repeated regions in an RNA secondary structure.

2) Conserved RNA secondary structures discovery in the human genome. By examining how the discovered structures differ from the results obtained from other studies that were recently carried out to search conserved RNA secondary structures in the human genome (Washietl 2005), (Pedersen 2006), (Khaladkar 2008), one can conclude that DiscoverR is not only a powerful tool for RNA motif discovery, but also presents unique searching capability that other current algorithms cannot provide. What's more exciting here is that this finding indicates there may exist much more conserved RNA secondary structures in the human genome that remain to be explored. And DiscoverR can play a critical role.

CHAPTER 4

PAIRWISE ALIGNMENT OF RNA SECONDARY STRUCTURES WITH COAXIAL HELICAL STACKING

4.1 Introduction

RNA secondary structures are composed of double-stranded segments such as helices connected to single-stranded regions such as junctions and hairpin loops. These structural elements serve as building blocks in the design of diverse RNA molecules with various functions in the cell [43-45]. In particular, RNA junctions are important structural elements due to their ability to orient many parts of the RNA molecule [46].

An RNA junction, also known as a multi-branch loop, forms when more than two helical segments are brought together [47-52]. RNA junctions exist in numerous RNA molecules; they play important roles in a wide variety of biochemical activities such as self-cleavage of the hammerhead ribozyme [53], the recognition of the binding pocket domain by purine riboswitches [54] and the translation initiation of the hepatitis C virus at the internal ribosome entry site [55]. Recent studies have classified RNA junctions with three and four branches into three and nine families, respectively [56,57]. Experiments have verified that a three-way junction in *Arabidopsis* has an important functional role [58]. A junction database, called RNAJunction, has been established, which contains junctions of all known degrees of branching [47].

A common tertiary motif within junctions of an RNA molecule is the coaxial stacking of helices [59-61], which occurs when two separate helical segments are aligned on a common axis to form a pseudocontiguous helix [62]. Coaxial stacking configurations have been observed in all large RNAs for which crystal structures are available, including tRNA, group I and II introns, RNase P, riboswitches and large ribosomal subunits. Coaxial helical stacking (CHS) provides thermodynamic stability to the RNA molecule as a whole [63] and reduces the separation between loop regions within junctions [64]. Moreover, coaxial stacking configurations form cooperatively with long-range interactions in many RNAs [56,59,65], and are therefore crucial as for correct tertiary structure formation as well as the formation of different junction topologies [57,59,66]. Since junctions are major architectural components in RNA, it is important to understand their structural properties. For example, the function of RNA molecules may be inferred if their junction components are similar in structure to other well-studied junction domains.

Figure 4.1 shows the example of an RNA molecule (PDB code: 1Y26) obtained from Protein Data Bank (PDB) [67], with a three-way junction [68,69]. Figure 4.1 (A) shows the 3D crystal structure of this adenine riboswitch molecule and drawn by PyMOL (http://www.pymol.org/). Each helix of this three-way junction is highlighted in different colors. The first helix according to the 5' to 3' orientation is Helix1, which is highlighted in blue. The second helix is Helix2, which is highlighted in green. The third helix is Helix 3, which is highlighted in red. The junction and two hairpin loops are highlighted in light grey. The junction is a multi-branch loop where three helices Helix1, Helix2 and Helix3 connect. Hairpin1 and Haprin2 are hairpin loops connected to Helix2 and Helix3, respectively. Figure 4.1 (B) shows the corresponding secondary structure of 1Y26, obtained from [70]. Notice that, there is a yellow bar across Helix1, Junction and Helix3, symbolizing a coaxial helical stacking H1H3 in the molecule 1Y26, as described in [59,68].

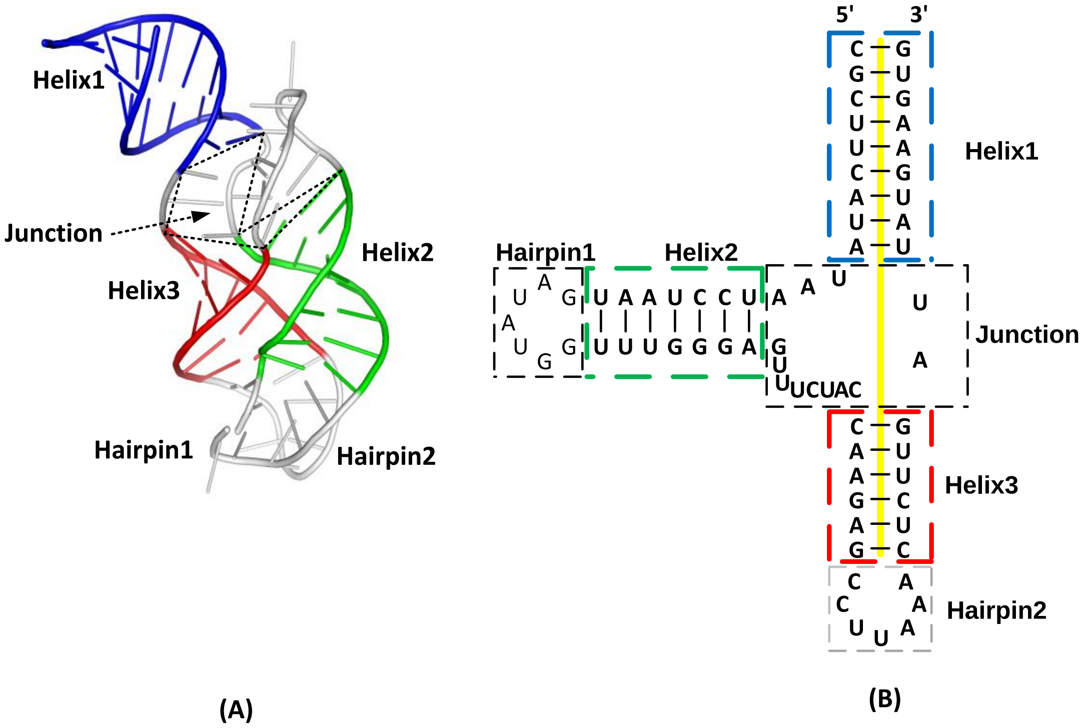
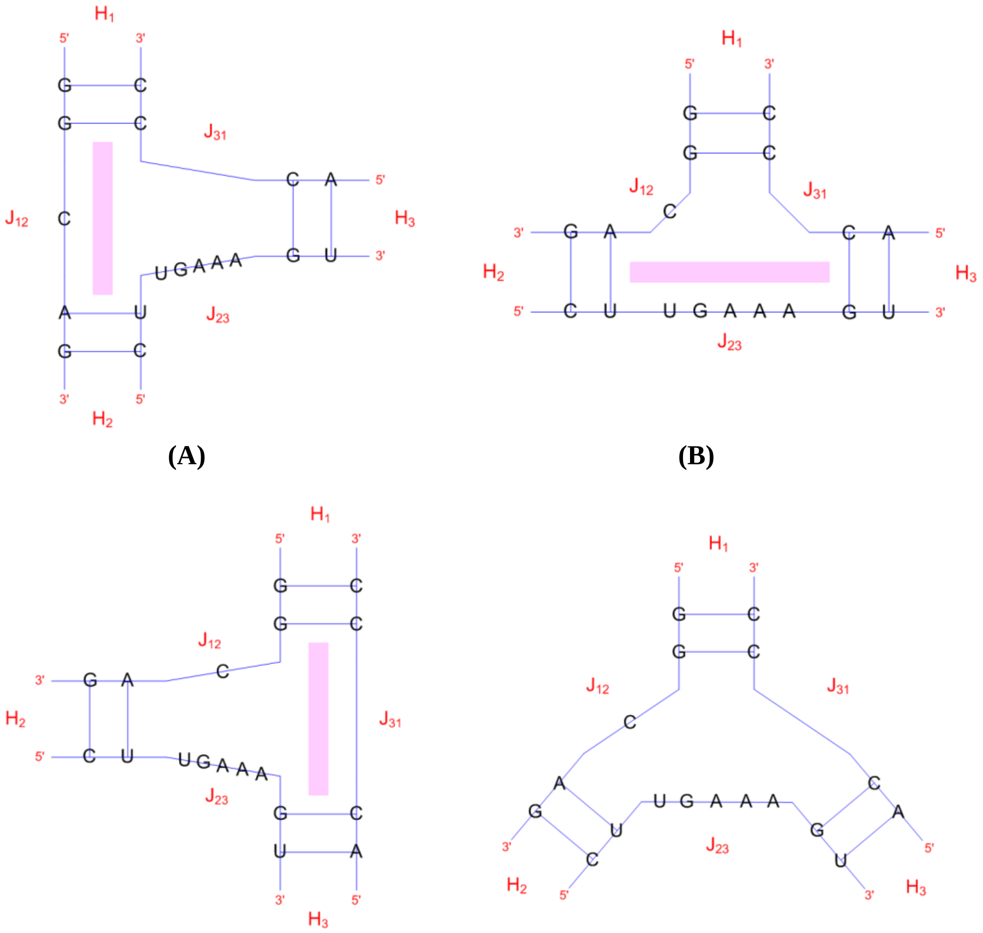


Figure 4.1 The RNA molecule (PDB code: 1Y26) with three-way junction. (A) 3D crystal strcutrue view. Helix 1 is showed in blue. Helix 2 is showed in gree. And Helix 3 is showed in red. (B) The secondary structure view.

In general, the coaxial helical stacking status of a three-way junction such as the junction in Figure 4.1 is described as one of four possibilities: $H_1H_{2,}$ $H_2H_{3,}$ $H_1H_{3,}$ or none, where H_xH_y indicates that H_x and H_y are coaxially stacked, i.e., helix H_x shares a common axis with helix H_y. The locations of the junctions and the coaxial helical stacking status of each junction in a given 2D structure can be determined using the methods described in [68]. Figure 4.2 gives the examples of these four possibilities of three-way junctions. Each pink bar across helices, symbolized coaxial helical stacking in that junction. There are seven possibilities for each four-way junction, H1H2, H2H3, H3H4, H1H4, H1H2-H3H4, H1H4-H2H3, or none, which are showed in Figure 4.3.



(A) (B)

(C) (D)

Figure 4.2 The four possibilities of three-way junctions, (A) for H1H2, (B) for H2H3, (C) for H1H3 and (D) for none.

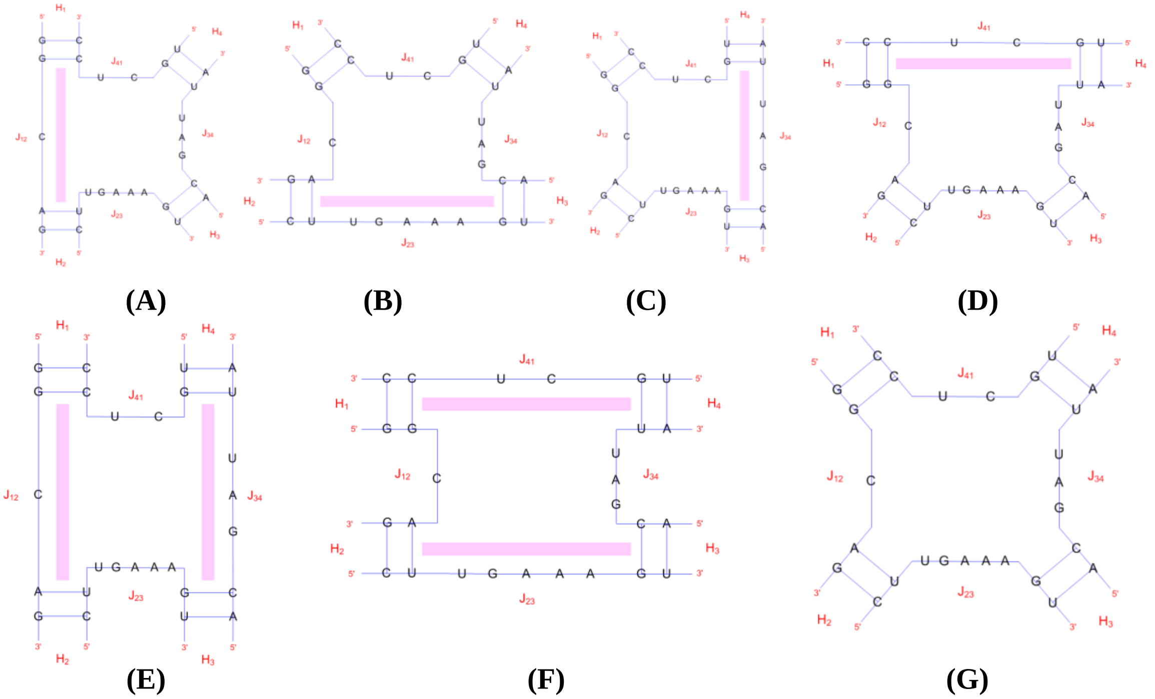


Figure 4.3 The seven possibilities of four-way junctions, (A) for H1H2, (B) for H2H3, (C) for H3H4, (D) for H1H4, (E) for H1H2-H3H4, (F) for H1H4-H2H3 and (G) for none.

Dr. Laing's research group previously developed Junction Explorer tool [68,69] for predicting coaxial stacking and RNAJAG [46] for modelling junction topologies as tree graphs. By a data mining approach known as random forests [71], which relies on a set of decision trees trained using length, sequence and other variables specified for any given junction, Junction Explorer predicts coaxial stacking within junctions with high accuracy [68]. The flowchart in Figure 4.4 is the procedure of the Junction Explorer [68,69]. The dataset we used is the updated dataset from Dr. Laing's previous works [57,66]. There are 216 RNA junctions collected in the dataset and only the Watson-Crick (AU, GC) and Wobble (GU) base pairs are considered. In Junction Explorer, a helix is defined as at least two consecutive base pairs. The number of junctions for each junction order is showed in Figure 4.5 [68].

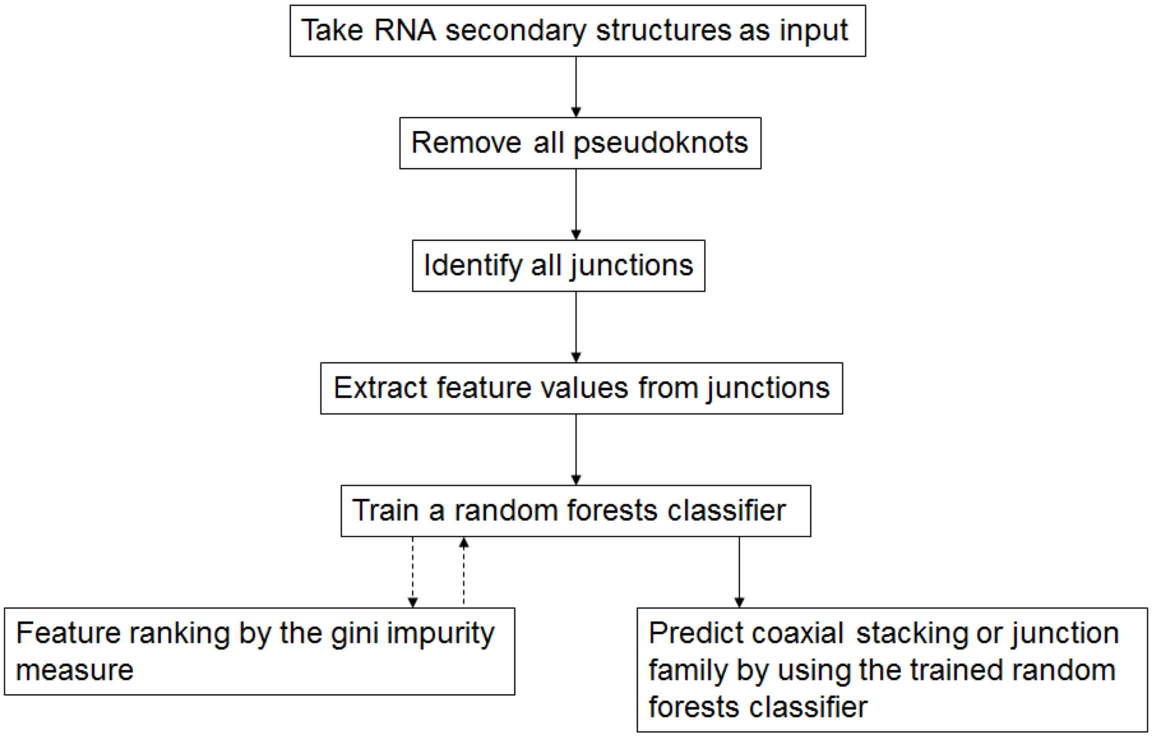

Take RNA secondary structures as input
Remove all pseudoknots
Identify all junctions
Extract feature values from junctions
Train a random forests classifier
Feature ranking by the gini impurity measure
Predict coaxial stacking or junction family by using the trained random forests classifier

Figure 4.4 The flowchart of Junction Explorer.

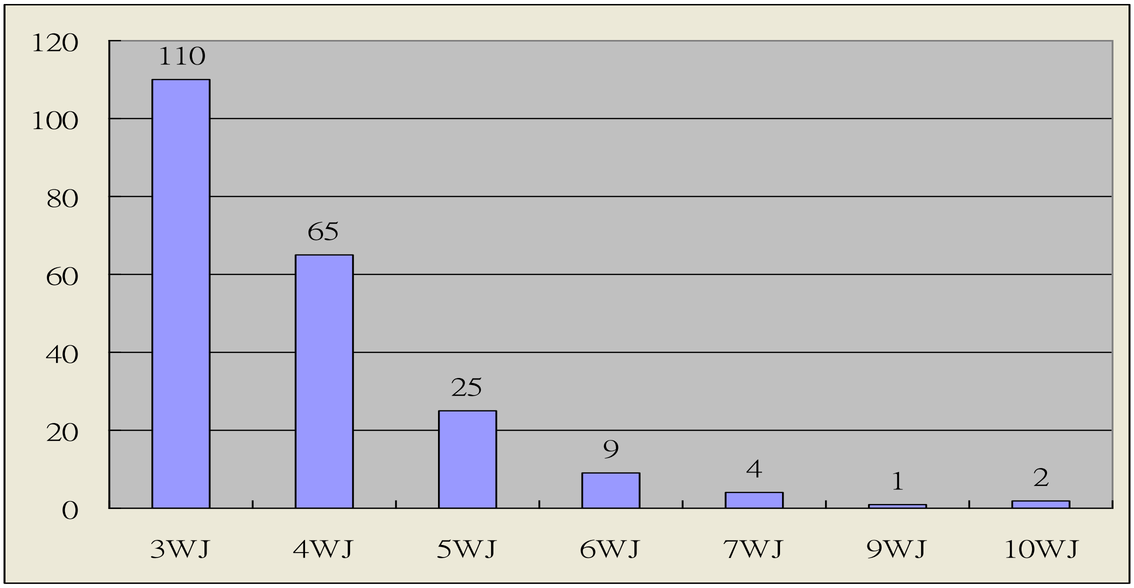
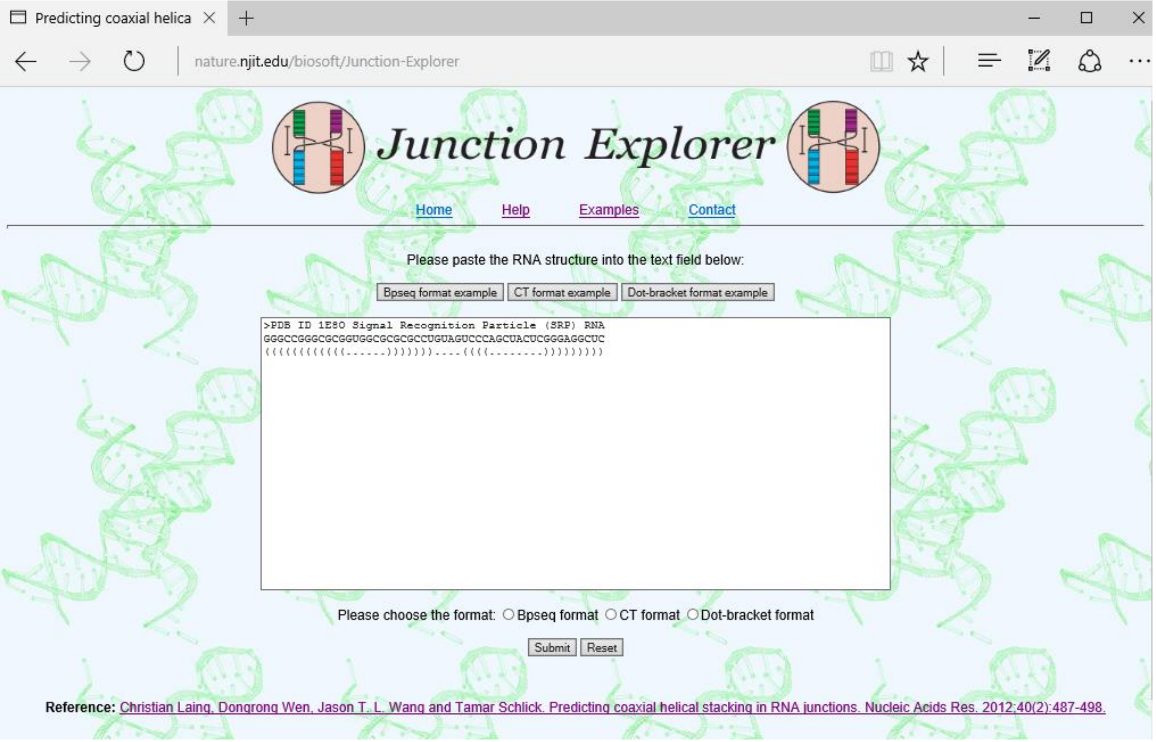


Figure 4.5 The number of junctions for each junction order.

We re-implement the web server of Junction Explorer. Figure 4.6 shows the web server of Junction Explorer and Figure 4.7 shows the result of 1E8O from Junction Explorer, which is including Junction Location, Junction Loops, Coaxial Stacking Prediction, Topology Prediction and Prediction Visualization of the junction in 1E8O.


Predicting coaxial helica
nature.njit.edu/biosoft/Junction-Explorer
Junction Explorer
Home
Help
Examples
Contact
Bpseq format example
CT format example
Dot-bracket format example
Submit
Reset

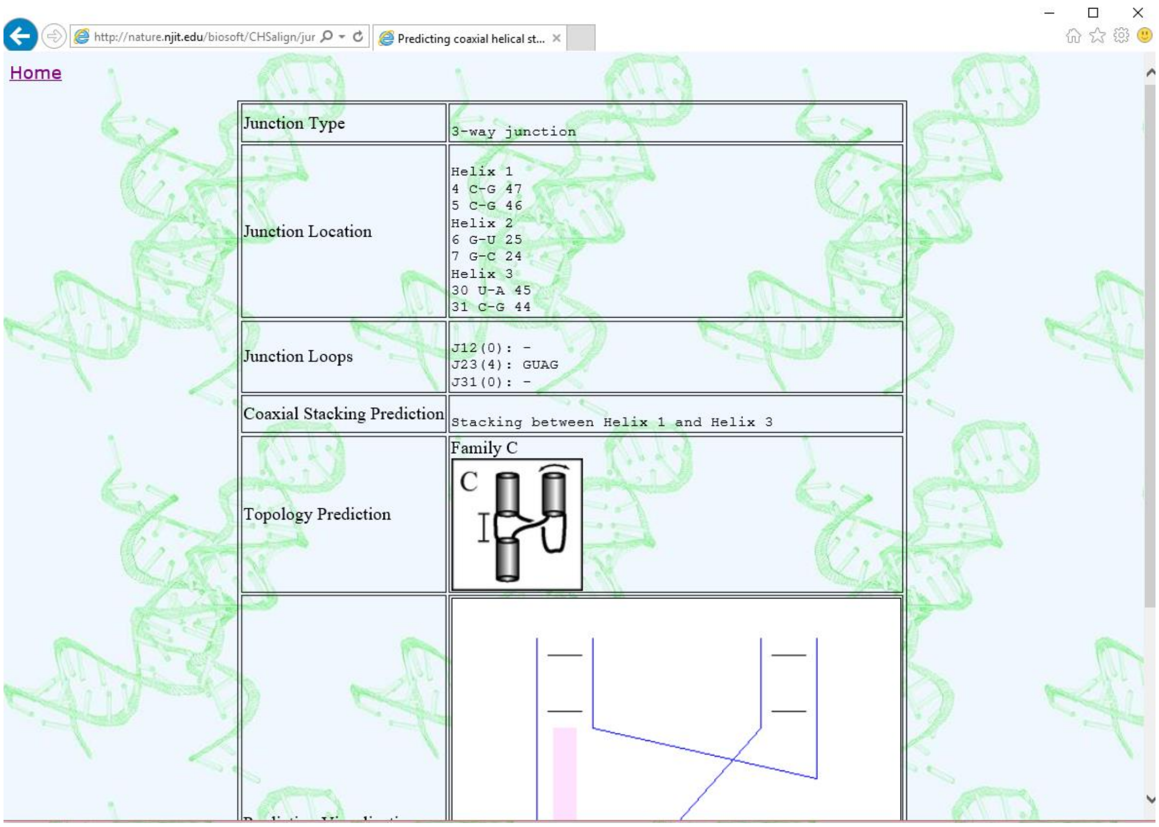
Please paste the RNA structure into the text field below:

```
>PDB ID 1E8O Signal Recognition Particle (SRP) RNA
GGGCCGGGCGCGGUGGCGCGCCUGUAGUCCCAGCUACUCGGGAGGCUC
((((((((((((......)))))))....((((........)))))))))
```

Please choose the format: ○ Bpseq format ○ CT format ○ Dot-bracket format

Reference: Christian Laing, Dongrong Wen, Jason T. L. Wang and Tamar Schlick. Predicting coaxial helical stacking in RNA junctions. Nucleic Acids Res. 2012;40(2):487-498.

Figure 4.6 The screenshot of the web server of Junction Explorer.


http://nature.njit.edu/biosoft/CHSalign/jur
Predicting coaxial helical st...
Home
C
I

Junction Type	3-way junction
Junction Location	Helix 1 4 C-G 47 5 C-G 46 Helix 2 6 G-U 25 7 G-C 24 Helix 3 30 U-A 45 31 C-G 44
Junction Loops	J12(0): - J23(4): GUAG J31(0): -
Coaxial Stacking Prediction	Stacking between Helix 1 and Helix 3
Topology Prediction	Family C
Prediction Visualization	

Figure 4.7 The screenshot of the result of 1E8O.

In this dissertation, we present a method, CHSalign, for aligning two RNA secondary (2D) structures that possess CHS motifs within the junctions of the two RNA structures. Coaxial stacking interactions in junctions are part of tertiary (3D) motifs [66]. Thus, CHSalign differs from both RNA 2D and 3D structure alignment tools. Existing secondary (2D) structure alignment tools focus on sequences and base pairs without considering tertiary motifs. Existing tertiary (3D) structure alignment tools accept as input two RNA 3D structures including all types of tertiary motifs in the Protein Data Bank

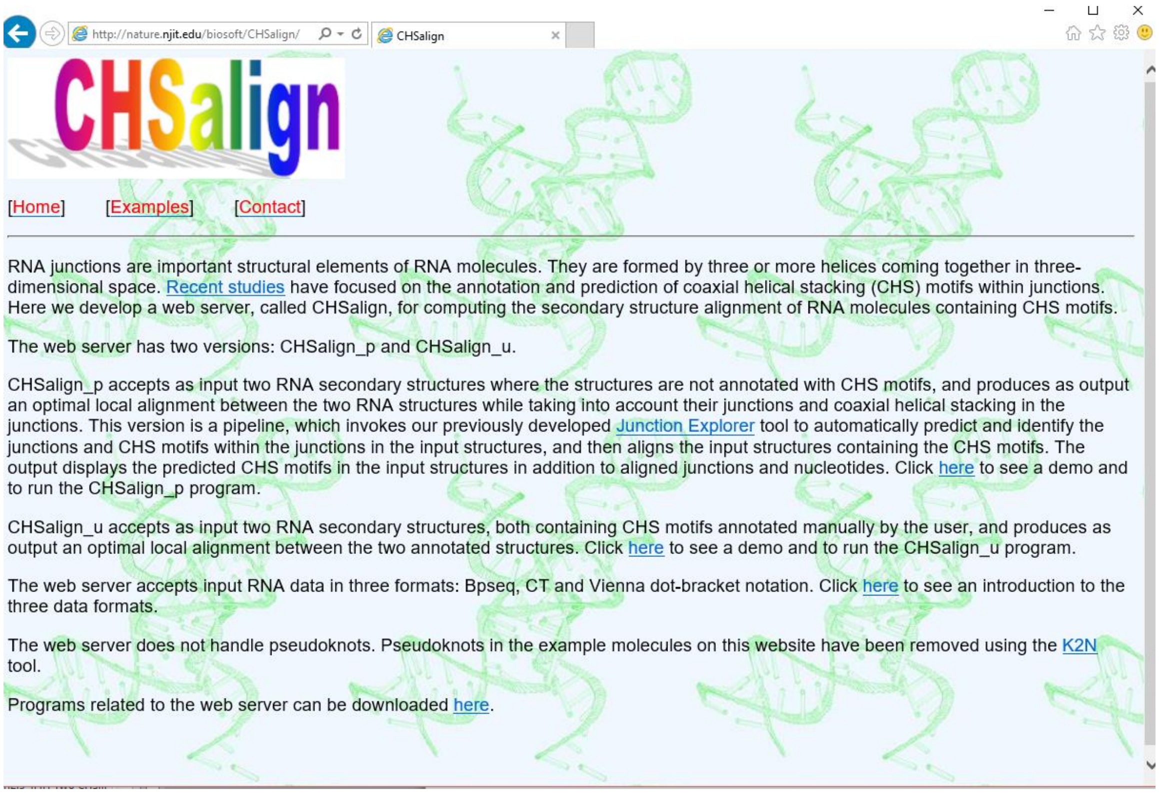
(PDB) [67] and align the 3D structures by considering their geometric properties, torsion angles, and base pairs.

For 3D structure alignment, Ferre *et al.* [72] developed a dynamic programming algorithm based on nucleotide, dihedral angle, and base pairing similarities. Capriotti and Marti-Renom [73] developed a program to align two RNA 3D structures based on a unit-vector root-mean-square approach. Chang, Huang, Lu [74] and Wang, Chen, Lu [75] employed a structural alphabet of different nucleotide conformations to align RNA 3D structures. Hoksza and Svozil [76] developed a pairwise comparison method based on 3D similarity of generalized secondary structure units. Sarver *et al.* [77] designed the FR3D tool for finding local and composite recurrent structural motifs in RNA 3D structures. Dror, Nussinov and Wolfson [78] described the RNA 3D structure alignment program, ARTS, and its use in the analysis and classification of RNA 3D structures [79]. Rahrig *et al.* [80] presented the R3D Align tool for performing global pairwise alignment of RNA 3D structures using local superpositions. He *et al.* [81] developed the RASS web server for comparing RNA 3D structures using both sequence and 3D structure information.

On the other hand, a well-adopted strategy for RNA 2D structure alignment is to use a tree transformation technique and perform RNA alignment through tree matching [15,82,83]. For instance, RNAforester [83] aligns two RNA 2D structures by calculating the edit-distance between tree structures symbolizing RNAs. By utilizing tree models to capture the structural particularities in RNA, RSmatch [15] aligns two RNA 2D structures effectively. Additional methods are described in [82,84].

In contrast to these methods for aligning two RNAs when their 2D structures are available, another group of closely related methods achieved RNA folding and alignment simultaneously. For instance, FOLDALIGN [85] uses a lightweight energy model and sequence similarity to simultaneously fold and align RNA sequences. Dynalign [86] finds a secondary structure common to two sequences without requiring any sequence identity. DAFS [87] simultaneously aligns and folds RNA sequences based on maximizing the expected accuracy of a predicted common secondary structure of the sequences. Similar techniques are implemented in CentroidAlign [88] and SimulFold [89]. SCARNA [90] employs a method of comparing RNA sequences based on the structural alignment of the fixed-length fragments of the stem candidates in the RNAs.

While many methods have been developed for RNA structure alignment, as surveyed above, few are tailored to junctions, especially junctions with coaxial stacking interactions. Junctions and coaxial stacking patterns are common in many RNA molecules and, as mentioned above, are involved in a wide range of functions. Furthermore, experimental probing techniques, such as RNA SHAPE chemistry, SAXS, NMR, and fluorescence resonance energy transfer (FRET), often provide sufficient information to determine coaxial stacking configurations [44,91-93]. Thus, a junction-tailored tool capable of comparing RNA structures on the basis of coaxial stacking patterns in their junctions could be particularly valuable. To this end, we present CHSalign, which performs RNA alignment by applying a constrained tree matching algorithm and dynamic programming techniques to ordered labelled trees symbolizing RNA structures with coaxial stacking patterns. Experimental results on different data sets demonstrate the effectiveness of this newly developed tool. The CHSalign web server is freely available at http://bioinformatics.njit.edu/CHSalign/, showed in Figure 4.8.



RNA junctions are important structural elements of RNA molecules. They are formed by three or more helices coming together in three-dimensional space. Recent studies have focused on the annotation and prediction of coaxial helical stacking (CHS) motifs within junctions. Here we develop a web server, called CHSalign, for computing the secondary structure alignment of RNA molecules containing CHS motifs.

The web server has two versions: CHSalign_p and CHSalign_u.

CHSalign_p accepts as input two RNA secondary structures where the structures are not annotated with CHS motifs, and produces as output an optimal local alignment between the two RNA structures while taking into account their junctions and coaxial helical stacking in the junctions. This version is a pipeline, which invokes our previously developed Junction Explorer tool to automatically predict and identify the junctions and CHS motifs within the junctions in the input structures, and then aligns the input structures containing the CHS motifs. The output displays the predicted CHS motifs in the input structures in addition to aligned junctions and nucleotides. Click here to see a demo and to run the CHSalign_p program.

CHSalign_u accepts as input two RNA secondary structures, both containing CHS motifs annotated manually by the user, and produces as output an optimal local alignment between the two annotated structures. Click here to see a demo and to run the CHSalign_u program.

The web server accepts input RNA data in three formats: Bpseq, CT and Vienna dot-bracket notation. Click here to see an introduction to the three data formats.

The web server does not handle pseudoknots. Pseudoknots in the example molecules on this website have been removed using the K2N tool.

Programs related to the web server can be downloaded here.

Figure 4.8 The screenshot of the Web Sever of CHSalign.

4.2 Materials and Methods

CHSalign accepts as input two RNA 2D structures which contain manually annotated coaxial stacking of helices, and produces as output an alignment between the two input structures. When manually annotated coaxial stacking patterns are not available, CHSalign invokes our previously developed Junction Explorer tool [68] to predict the coaxial stacking configurations of the input structures.

Our approach is to transform each input RNA 2D structure with coaxial stacking patterns into an ordered labeled tree. Tree graphs are popular models for representing RNA structures [46,65,83,94-96]. We extend modeling as tree graphs in RNAJAG [46] to obtain an ordered tree model, in which each tree node represents a secondary structure element such as a helix (stem), junction or hairpin loop. When comparing two tree nodes, we use a dynamic programming algorithm [15,82] to align the 2D structural elements in the tree nodes, obtaining a score between the two nodes. We then use a constrained tree matching algorithm to find an optimal alignment between the two input RNA 2D structures, taking into account their coaxial stacking configurations. Below, this dissertation details the tree model and the constrained tree matching algorithm.

4.2.1 Tree Model Formalization

Let R_{seq} be an RNA sequence containing nucleotides or bases A, C, G, U. $R_{seq}[i]$ denotes the base at position i of R_{seq} ordered from the 5′ to 3′ ends. $R_{seq}[i, j]$, $i < j$, is the subsequence starting at position i and ending at position j. Let R be the 2D structure of R_{seq} with at least one base pair. A helix in R is a double-stranded segment composed of contiguous base pairs. A base pair connecting position i and position j is denoted by (i, j) and its enclosed subsequence is $R_{seq}[i, j]$. If all nucleotides in $R_{seq}[i, j]$ except $R_{seq}[i]$ and $R_{seq}[j]$ are unpaired single bases, and (i, j) is a base pair in R, we call $R_{seq}[i+1, j-1]$ a hairpin loop.

A junction, or a multi-branch loop, is an enclosed area connecting different helices [49]. An n-way junction in R has n branches. This junction connects n helices where there are n base pairs (i_1, j_1)... (i_n, j_n) (one base pair for each helix), and n subsequences participating in the junction. The n subsequences are denoted by $R_{seq}[i_1+1, i_2-1]$, $R_{seq}[j_2+1, i_3-1]$, $R_{seq}[j_3+1, i_4-1]$, ... , $R_{seq}[j_{n-1}+1, i_n-1]$, and $R_{seq}[j_n+1, j_1-1]$. All the unpaired bases on the n subsequences comprise the n-way junction, and the subsequences are called the loop regions of the junction. Internal loops or bulges can be considered as special cases of “two-way” junctions [47]. However, for the purpose of this work, n must be greater than 2. Thus, internal loops or bulges are not considered as junctions in our work; instead, they are considered as part of the helices in R.

CHSalign transforms the 2D structure R into an ordered labeled tree T in which each node has a label and the left-to-right order among sibling nodes is important. Each node of T represents a 2D structural element of R, belonging to one of three types: helix, junction, and hairpin loop. With this tree model, pseudoknots are excluded.

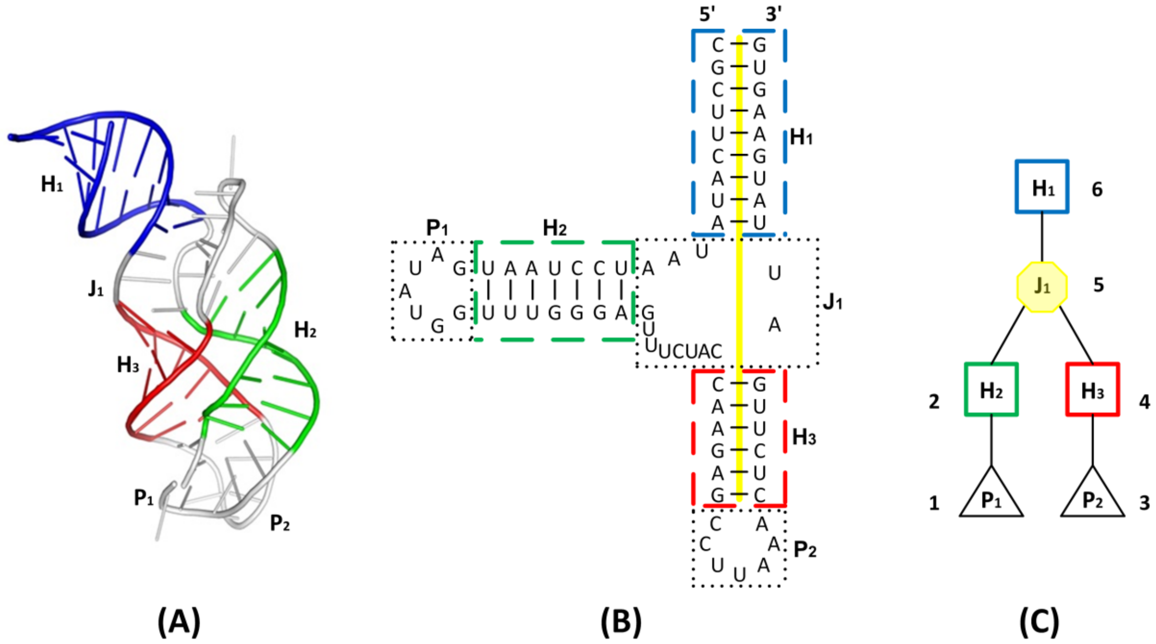


Figure 4.9 Transformation of an RNA 3D molecule into an ordered labeled tree.

Figure 4.9 illustrates the transformation process. Figure 4.9 (A) shows the 3D crystal structure of the adenine riboswitch molecule (PDB code: 1Y26) obtained from the Protein Data Bank (PDB) [67] and drawn by PyMOL. Figure 4.9 (B) shows the corresponding 2D structure, obtained from RNAView [97]. Each 2D structural element in Figure 4.9 (B) is highlighted as in Figure 4.9 (A). The yellow bar across H_1, J_1 and H_3, symbolizing a coaxial helical stacking H_1H_3 in the molecule 1Y26. Figure 4.9 (C) shows the tree, *T*, used to represent the 2D structure *R* in Figure 4.9 (B). Each node of *T* corresponds to a 2D structural element of *R* where the octagon (squares, triangles, respectively) in *T* represents the junction (helices, hairpin loops, respectively) in *R*. Thus, like the 2D structural elements, each tree node belongs to one of three types, namely helix, junction, and hairpin loop. Tree nodes of different types are prohibited to be aligned with each other, and hence the term “constrained tree matching” is used in our work (reminiscent of structural constraints in RNA described in [98]).

Use $t[i]$ to represent the node of tree T whose position in the left-to-right post-order traversal of T is i. The post-order procedure works by first traversing the left subtree, then traversing the right subtree, and finally visiting the root. In Figure 4.9 (C), the post-order position number of each node is shown next to the node. By construction, the tree node corresponding to an n-way junction consists of $n – 1$ children. The first helix according to the 5′ to 3′ orientation is the parent node of the junction node. The other $n – 1$ helices are the children of that junction node. The number of children of node $t[i]$ is the degree of $t[i]$. In Figure 4.9 (C), H_1 is the parent node of J_1, which has two children, H_2 and H_3. The degree of the junction node J_1 is 2. In general, the degree of an n-way junction node is $n – 1$.

Consider two RNA 2D structures R_1 and R_2 and their tree representations T_1 and T_2, respectively. Let $t_1[i]$ ($t_2[j]$, respectively) be the node of T_1 (T_2, respectively) whose position in the post-order traversal of T_1 (T_2, respectively) is i (j, respectively). Let $T_1[i]$ be the subtree rooted at $t_1[i]$, and $T_2[j]$ be the subtree rooted at $t_2[j]$. $F_1[i]$ represents the forest obtained by removing the root $t_1[i]$ from subtree $T_1[i]$. $F_2[j]$ represents the forest obtained by removing the root $t_2[j]$ from subtree $T_2[j]$. Suppose the degree of $t_1[i]$ is m_i (i.e., $t_1[i]$ has m_i children $t_1\left[i_1\right],\ldots,t_1\left[i_{m_i}\right]$) and the degree of $t_2[j]$ is n_j (i.e., $t_2[j]$ has n_j children

$t_2\left[j_1\right],\ldots,t_2\left[j_{n_j}\right]$). Use $S(T_1[i], T_2[j])$ to represent the alignment score of subtree $T_1[i]$ and subtree $T_2[j]$, and use $\gamma(t_1[i], t_2[j])$ to represent the alignment score of node $t_1[i]$ and node $t_2[j]$. We use ∅ to represent an empty node; matching a tree node with ∅ amounts to aligning all nucleotides in the tree node to gaps.

4.2.2 Alignment Scheme

CHSalign employs a dynamic programming algorithm to align two RNA 2D structures with coaxial stacking patterns. The approach is to transform each RNA 2D structure into an ordered labeled tree as explained in the previous subsection. CHSalign then apply the dynamic programming algorithm to the ordered labeled trees representing the two RNA 2D structures. Based on the alignment of the trees, CHSalign obtain the alignment of the corresponding RNA 2D structures. As noted above, each tree node belongs to one of three types: helix, junction, and hairpin loop. Different types of tree nodes are prohibited to be aligned with each other. Figure 4.10 gives two PDB molecules, A-riboswitch (PDB code: 1Y26) and the Alu domain of the mammalian signal recognition particle (SRP) (PDB code: 1E8O), are considered. Figure 4.10 (A) shows the 3D crystal structure of the adenine riboswitch molecule and its tree representation T_1. Figure 4.10 (B) shows the 3D crystal structure of the Alu domain of the mammalian SRP molecule and its tree representation T_2. When aligning two subtrees $T_1[i]$ and $T_2[j]$ and calculating the score $S(T_1[i], T_2[j])$, there are nine cases to be considered.

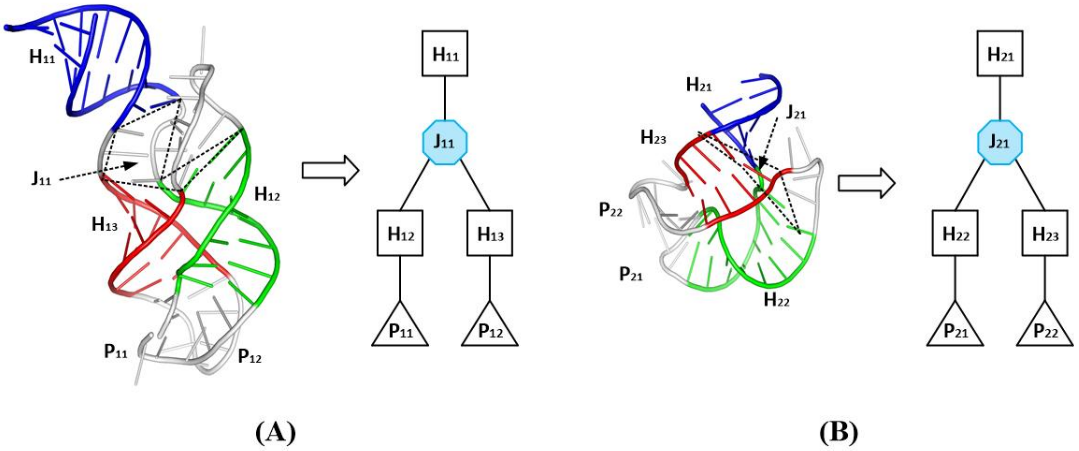


Figure 4.10 Example of an alignment between two RNA molecules. (A) The 3D crystal structure of the adenine riboswitch (PDB code: 1Y26) and its tree representation T_1. (B) The 3D crystal structure of the Alu domain of the mammalian signal recognition particle (SRP) (PDB code: 1E8O) and its tree representation T_2.

Case 1. Both $t_1[i]$ and $t_2[j]$ are junctions.

One constraint we impose on pairwise alignment is that when aligning a p-way junction node v_1 with a q-way junction node v_2, p must be equal to q. Furthermore, the coaxial helical stacking status of v_1 must be the same as the coaxial stacking status of v_2. Thus, a three-way junction must be aligned with a three-way junction, which is not allowed to align with a four-way junction. Furthermore, a three-way junction whose coaxial helical stacking status is H_1H_2 must be aligned with a three-way junction having the same H_1H_2 status, which is not allowed to align with a three-way junction whose coaxial helical stacking status is H_2H_3. In general, junctions with different branches and different coaxial stacking configurations have different biological properties. This constraint is established to ensure a biologically meaningful alignment is obtained, and to avoid introducing too many gaps in the alignment.

According to our tree model, if a tree node is a junction, it must have at least two children and the children must be helix nodes. A junction contains loop regions with single bases whereas helices are double-stranded regions with base pairs. A junction node is thus prohibited to be aligned with a helix node. Hence, $t_1[i]$ must be aligned with $t_2[j]$ provided they have the same number of branches and the same coaxial helical stacking status, denoted by $\Psi(t_1[i]) = \Psi(t_2[j])$. Their children are trees, which together form forests $F_1[i]$ and $F_2[j]$, respectively. $F_1[i]$ must be aligned with $F_2[j]$ (Figure 4.11). Thus the alignment score of $T_1[i]$ and $T_2[j]$ can be calculated as:

$$S\left(T_1[i], T_2[j]\right) = \max \begin{cases} \gamma\left(t_1[i], t_2[j]\right) + S\left(F_1[i], F_2[j]\right) \\ 0 \end{cases}. \tag{4.1}$$

If $\Psi(t_1[i]) = \Psi(t_2[j])$, $t_1[i]$ and $t_2[j]$ must have the same number of children, and the order among the sibling nodes is important. If $\Psi(t_1[i]) \neq \Psi(t_2[j])$, i.e., $t_1[i]$ and $t_2[j]$ have different numbers of children (branches) or they have different coaxial helical stacking statuses, they are prohibited to be aligned together. Thus, the score of matching $F_1[i]$ with $F_2[j]$ can be calculated as:

$$S\left(F_1[i], F_2[j]\right) = \begin{cases} S\left(T_1[i_1], T_2[j_1]\right) + \ldots + S\left(T_1[i_m], T_2[j_m]\right) & \text{if } \Psi\left(t_1[i]\right) = \Psi\left(t_2[j]\right) \\ -\infty & \text{otherwise} \end{cases} \tag{4.2}$$

where m is the number of children of $t_1[i]$ and $t_2[j]$ respectively.

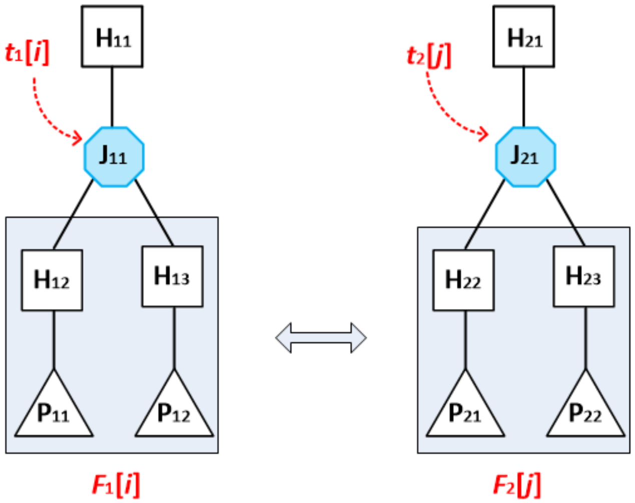


Figure 4.11 Illustration for both $t_1[i]$ and $t_2[j]$ junctions, and $\Psi(t_1[i]) = \Psi(t_2[j])$.

We use $\Pi(t_1[i])$ to represent the coaxial helical stacking status of $t_1[i]$; $\Pi(t_1[i]) =$ 1 (2, 3, 0, respectively) if the coaxial helical stacking status of $t_1[i]$ is H_1H_2 (H_2H_3, H_1H_3, none, respectively). The score of matching $t_1[i]$ with $t_2[j]$ is

$$\gamma(t_1[i],t_2[j]) = \begin{cases} s+w & \text{if } \Psi(t_1[i]) = \Psi(t_2[j]),\ \Pi(t_1[i]) \neq 0,\ \Pi(t_2[j]) \neq 0 \\ s+w/2 & \text{if } \Psi(t_1[i]) = \Psi(t_2[j]),\ \Pi(t_1[i]) = \Pi(t_2[j]) = 0 \\ -\infty & \text{otherwise} \end{cases} . \quad (4.3)$$

Here, s is the score obtained by aligning the junction in $t_1[i]$ with the junction in $t_2[j]$. We use a dynamic programming algorithm [15,82] to calculate the alignment score s, and adopt the RIBOSUM85-60 matrix [99] to calculate the score of aligning two bases or base pairs in RNA 2D structures. (The default gap penalty is –1.) With this scoring matrix, CHSalign can handle non-canonical base pairs. The addition of a parameter w to the alignment score is a computational device to enforce the right alignment of the RNAs when the junction patterns match. Thus, if $t_1[i]$ and $t_2[j]$ have the same number of branches, their CHS patterns are alike, and $\Pi(t_1[i]) \neq 0, \Pi(t_2[j]) \neq 0$, we use s+w as the modified alignment score. When $t_1[i]$ and $t_2[j]$ have the same number of branches and $\Pi(t_1[i]) = \Pi(t_2[j]) = 0$, we use s+(w/2) as the modified score. The value of w required experimentation, as the discussion in the later chapter, but a value of 100 seems to work well in practice.

Case 2. Both $t_1[i]$ and $t_2[j]$ are helices.

Due to the nature of RNA 2D structures and based on our tree model, a helix has only one child, which is either a junction or a hairpin loop. The subtree rooted at the child of $t_1[i]$ is denoted by $T_1[i-1]$ and the subtree rooted at the child of $t_2[j]$ is denoted by $T_2[j-1]$ (Figure 4.12). CHSalign has to match helix nodes $t_1[i]$ and $t_2[j]$ first, and then add the alignment score of their subtrees $T_1[i-1]$ and $T_2[j-1]$ if the alignment score of the subtrees is greater than or equal to zero, or simply match $t_1[i]$ with $t_2[j]$ if the alignment score of their subtrees is negative (i.e., the subtrees are not aligned). Therefore, the alignment score of $T_1[i]$ and $T_2[j]$ can be calculated as:

$$S\left(T_1[i],T_2[j]\right)=\max\begin{cases}\gamma\left(t_1[i],t_2[j]\right)+S\left(T_1[i-1],T_2[j-1]\right)\\ \gamma\left(t_1[i],t_2[j]\right)\\ 0\end{cases}. \tag{4.4}$$

The score $\gamma(t_1[i], t_2[j])$ is obtained by aligning the helix in $t_1[i]$ with the helix in $t_2[j]$ using a dynamic programming algorithm [15,82]. The value 0 is used if the other entries in Equation (4.4) yield negative scores.

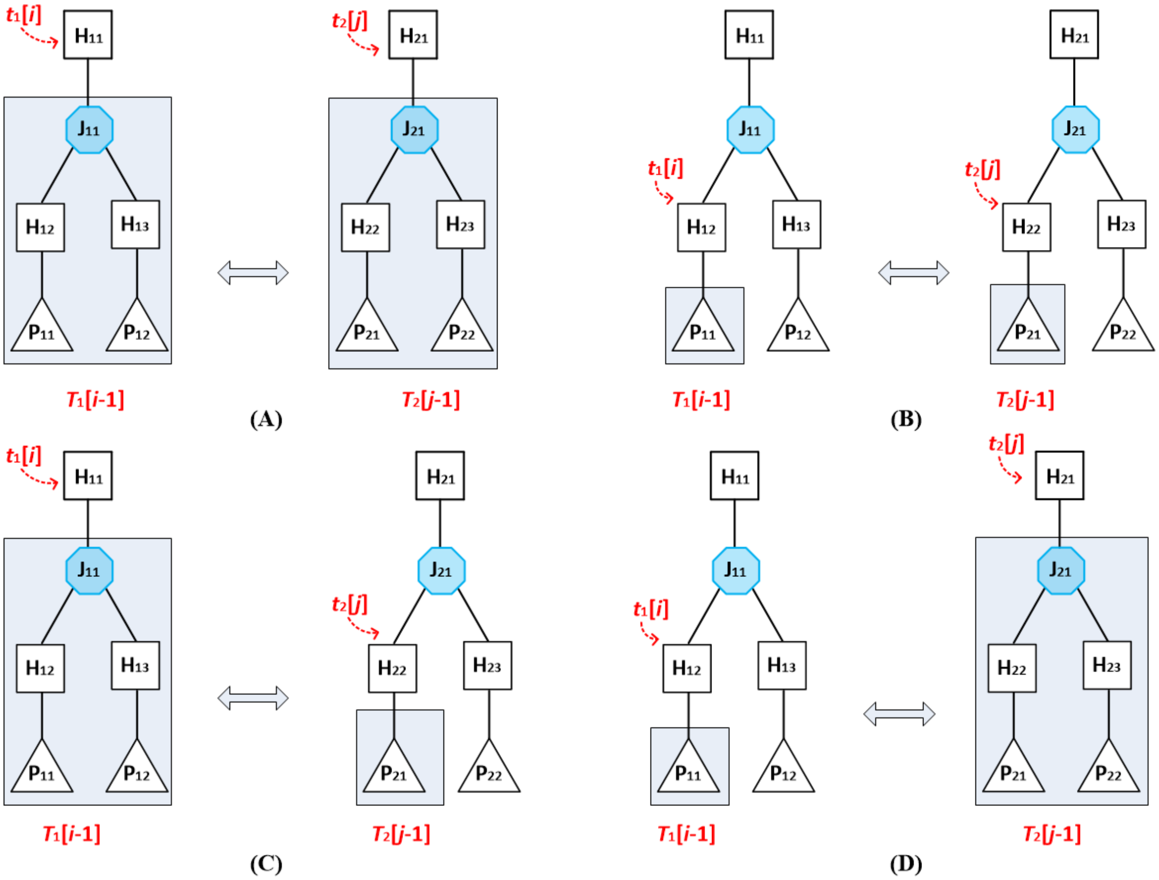


Figure 4.12 Illustration of four possibilities when both $t_1[i]$ and $t_2[j]$ are helices.

Case 3. Both $t_1[i]$ and $t_2[j]$ are hairpin loops.

Due to the nature of RNA 2D structures and based on our tree model, a hairpin does not have any child. Therefore hairpin nodes are always leaves in the tree representation of an RNA 2D structure. When both $t_1[i]$ and $t_2[j]$ are hairpin loops, matching $T_1[i]$ with $T_2[j]$ amounts to matching $t_1[i]$ with $t_2[j]$ (Figure 4.13). Thus, the alignment score becomes:

$$S\left(T_1[i],T_2[j]\right)=\max\begin{cases}\gamma\left(t_1[i],t_2[j]\right)\\0\end{cases}. \tag{4.5}$$

The score $\gamma(t_1[i], t_2[j])$ is obtained by aligning the hairpin loop in $t_1[i]$ with the hairpin loop in $t_2[j]$ using a dynamic programming algorithm [15,82].

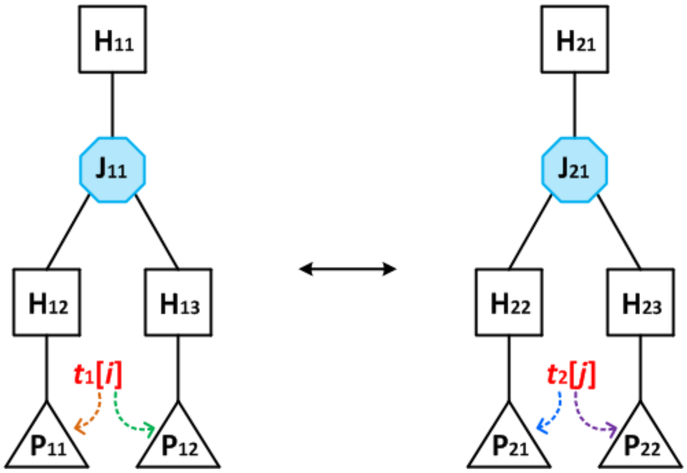


Figure 4.13 Illustration of the alignment when both $t_1[i]$ and $t_2[j]$ are hairpin loops.

Case 4. $t_1[i]$ is a junction and $t_2[j]$ is a helix.

Since $t_1[i]$ and $t_2[j]$ have different types, they cannot be aligned with each other. There are two subcases.

Subcase 1. $t_2[j]$ is aligned to gaps. Then $T_1[i]$ must be aligned with $T_2[j-1]$, which is the subtree rooted at the child of $t_2[j]$.

Subcase 2. $t_1[i]$ is aligned to gaps. Suppose $t_1[i]$ has m_i children $t_1[i_1],\ldots,t_1\left[i_{m_i}\right]$. The subtrees rooted at these children are denoted by $T_1[i_1],\ldots,T_1\left[i_{m_i}\right]$, respectively. Then, one of these subtrees must be aligned with $T_2[j]$; specifically the subtree yielding the maximum alignment score is aligned with $T_2[j]$.

We take the maximum of the above two subcases. Thus, the score of matching $T_1[i]$ with $T_2[j]$ can be calculated as:

$$S\left(T_1\left[i\right],T_2\left[j\right]\right)=\max\begin{cases}\gamma\left(\varnothing,t_2\left[j\right]\right)+S\left(T_1\left[i\right],T_2\left[j-1\right]\right)\\ \gamma\left(t_1\left[i\right],\varnothing\right)+\max_{1\le k\le m_i}\left\{S\left(T_1\left[i_k\right],T_2\left[j\right]\right)\right\}\\ 0\end{cases}. \qquad (4.6)$$

The value 0 is used if both of the two subcases yield negative scores.

When matching $T_1[i]$ with $T_2[j]$, since $t_1[i]$ and $t_2[j]$ have different types where $t_1[i]$ is a junction and $t_2[j]$ is a helix, there are two subcases to be considered, as detailed above. Figure 4.14 (A) illustrates subcase 1, in which $t_2[j]$ is aligned to gaps and $T_1[i]$ is aligned with $T_2[j-1]$. Figure 4.14 (B) illustrates subcase 2, in which $t_1[i]$ is aligned to gaps, and the subtree rooted at one of the children of $t_1[i]$ is aligned with $T_2[j]$. In our example here, $t_1[i]$ has two children, $t_1\left[i_1\right]$ and $t_1\left[i_2\right]$. Thus, either the subtree rooted at $t_1\left[i_1\right]$, denoted by $T_1\left[i_1\right]$, is aligned with $T_2[j]$ as illustrated in Figure 4.14 (B1), or the subtree rooted at $t_1\left[i_2\right]$, denoted by $T_1\left[i_2\right]$, is aligned with $T_2[j]$ as illustrated in Figure 4.14 (B2). The maximum alignment score obtained from Figure 4.14 (B1) and Figure 4.14 (B2) is used. Then $S(T_1[i], T_2[j])$ is calculated by taking the maximum of the two subcases illustrated in Figure 4.14 (A) and Figure 4.14 (B), respectively.

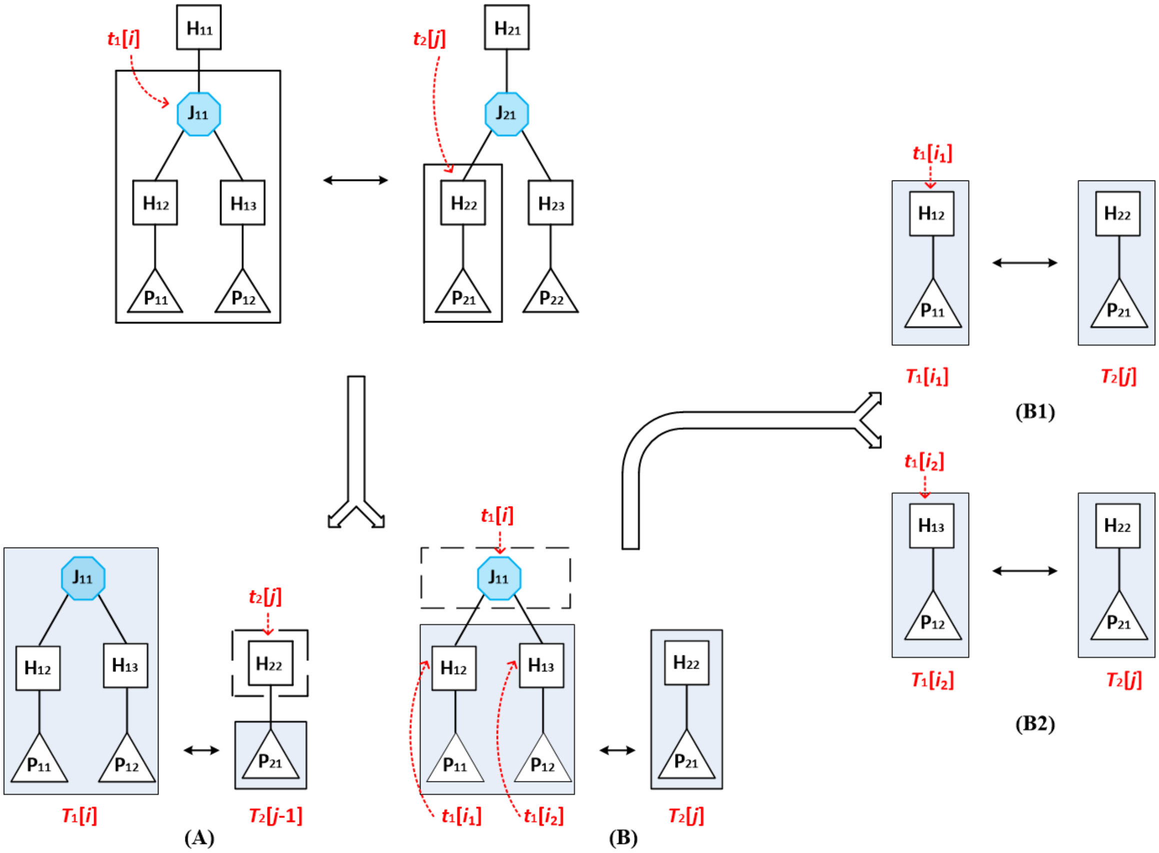


Figure 4.14 Illustration of case 4.

Case 5. $t_1[i]$ is a junction and $t_2[j]$ is a hairpin loop.

Since $t_1[i]$ and $t_2[j]$ have different types, the two nodes cannot be aligned together. Furthermore, $t_2[j]$ is a hairpin loop, which does not have any child. Thus $t_1[i]$ must be aligned to gaps, and the subtree rooted at one of the children of $t_1[i]$ is aligned with $T_2[j]$ (Figure 4.15); specifically the subtree yielding the maximum alignment score is aligned with $T_2[j]$. Therefore, the alignment score of $T_1[i]$ and $T_2[j]$ can be calculated as:

$$S\left(T_1[i],T_2[j]\right)=\max\begin{cases}\gamma\left(t_1[i],\varnothing\right)+\max_{1\le k\le m_i}\left\{S\left(T_1[i_k],T_2[j]\right)\right\}\\0\end{cases}. \tag{4.7}$$

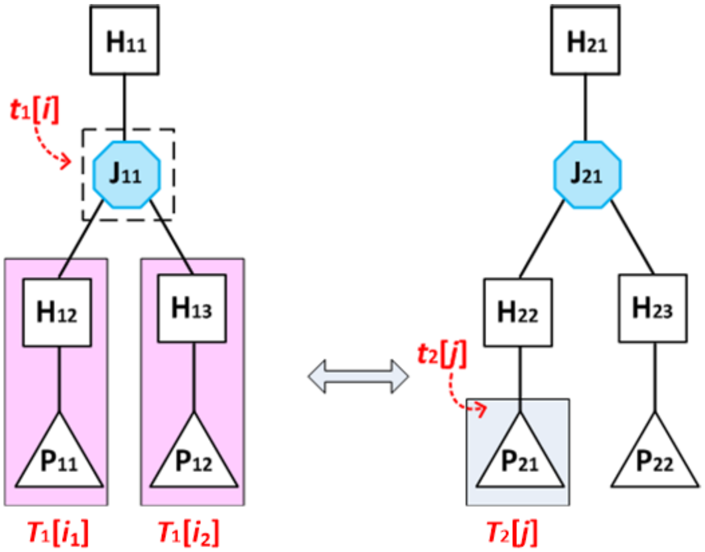


Figure 4.15 Illustration for case 5.

Case 6. $t_1[i]$ is a helix and $t_2[j]$ is a junction.

Similar to Case 4, there are two subcases.

Subcase 1. $t_i[i]$ is aligned to gaps. Thus, the subtree rooted at the child of $t_1[i]$, denoted by $T_1[i-1]$, must be aligned with $T_2[j]$.

Subcase 2. $t_2[j]$ is aligned to gaps. Suppose $t_2[j]$ has n_j children $t_2[j_1],\ldots,t_2\left[j_{n_j}\right]$. The subtrees rooted at these children are $T_2[j_1],\ldots,T_2\left[j_{n_j}\right]$, respectively. Then $T_1[i]$ must be aligned with one of these subtrees.

Taking the maximum of these two subcases, we calculate the score of matching $T_1[i]$ with $T_2[j]$ as:

$$S\left(T_1[i],T_2[j]\right)=\max\begin{cases}\gamma\left(t_1[i],\varnothing\right)+S\left(T_1[i-1],T_2[j]\right)\\ \gamma\left(\varnothing,t_2[j]\right)+\max_{1\le k\le n_j}\left\{S\left(T_1[i],T_2[j_k]\right)\right\}\\ 0\end{cases}. \tag{4.8}$$

Case 7. $t_1[i]$ is a helix and $t_2[j]$ is a hairpin loop.

Because $t_1[i]$ and $t_2[j]$ have different types, the two nodes cannot be aligned together. Furthermore, since $t_1[i]$ is a helix, it has only one child; $t_2[j]$ is a hairpin loop with no children. Therefore, $t_1[i]$ must be aligned to gaps and the subtree rooted at the child of $t_1[i]$, denoted by $T_1[i-1]$, must be aligned with $T_2[j]$ (Figure 16), or if the alignment yields a negative score, we use the value 0. Thus, the alignment score is

$$S\left(T_1[i],T_2[j]\right)=\max\begin{cases}\gamma\left(t_1[i],\varnothing\right)+S\left(T_1[i-1],T_2[j]\right)\\ 0\end{cases}. \tag{4.9}$$

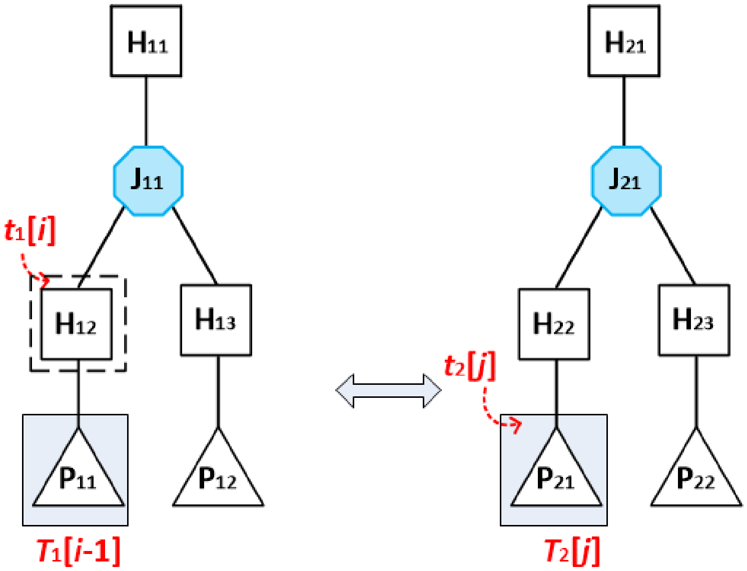


Figure 4.16 Illustration for case 7.

Case 8. $t_1[i]$ is a hairpin loop and $t_2[j]$ is a junction.

This is similar to Case 5. Thus, we can calculate the score of matching $T_1[i]$ with $T_2[j]$ as:

$$S\left(T_1[i],T_2[j]\right)=\max\begin{cases}\gamma\left(\varnothing,t_2[j]\right)+\max_{1\le k\le n_j}\left\{S\left(T_1[i],T_2[j_k]\right)\right\}\\ 0\end{cases}. \tag{4.10}$$

Case 9. $t_1[i]$ is a hairpin loop and $t_2[j]$ is a helix.

This is similar to Case 7, with the alignment score:

$$S\left(T_1[i],T_2[j]\right)=\max\begin{cases}\gamma\left(\varnothing,t_2[j]\right)+S\left(T_1[i],T_2[j-1]\right)\\0\end{cases}. \tag{4.11}$$

4.2.3 Time and Space Complexity

Let $|T_1|$ ($|T_2|$, respectively) denote the number of nodes in tree T_1 (T_2, respectively) that represents RNA structure R_1 (R_2, respectively). CHSalign maintains a two-dimensional table in which $c(i, j)$ represents the cell located at the intersection of the ith row and the jth column of the table. The value stored in the cell $c(i, j)$, $1 \le i \le |T_1|$, $1 \le j \le |T_2|$, is $S(T_1[i], T_2[j])$. The dynamic programming algorithm employed by CHSalign calculates the values in the table by traversing the trees T_1 and T_2 in a bottom-up manner. After all the values in the table are computed, the algorithm locates the cell c with the maximum value. A backtrack procedure starting with the cell c and terminating when encountering a zero identifies the alignment lines of an optimal alignment and calculates the alignment score between T_1 and T_2.

Let $|R_1|$ ($|R_2|$, respectively) denote the number of nucleotides, i.e., the length, of RNA structure R_1 (R_2, respectively). Let $|t_1[i]|$ ($|t_2[j]|$, respectively) be the number of nucleotides in node $t_1[i]$ ($t_2[j]$, respectively). Let d_1 (d_2, respectively) be the maximum degree of any node in tree T_1 (T_2, respectively). The time complexity of computing $\gamma(t_1[i],$

$t_2[j]$) is $O\left(\left|t_1[i]\right|\times\left|t_2[j]\right|\right)$ [15]. Thus, the time complexity of computing $S(T_1[i],T_2[j])$ is $O\left(\max(d_1,d_2)+\left|t_1[i]\right|\times\left|t_2[j]\right|\right)$. Here max($d_1$, d_2) is a constant because a junction has at most twelve branches in solved RNA crystal structures [46,68,96]. Furthermore, $\sum_{i=1}^{|T_1|}\left|t_1(i)\right|=\left|R_1\right|$ and $\sum_{j=1}^{|T_2|}\left|t_2(j)\right|=\left|R_2\right|$. Therefore, the time complexity of calculating all the values in the two-dimensional table is

$$\begin{aligned} & O\left(\sum_{i=1}^{|T_1|}\sum_{j=1}^{|T_2|}\left(\max(d_1,d_2)+\left|t_1[i]\right|\times\left|t_2[j]\right|\right)\right) \\ = & O\left(\sum_{i=1}^{|T_1|}\sum_{j=1}^{|T_2|}\left(\left|t_1[i]\right|\times\left|t_2[j]\right|\right)\right) \\ = & O\left(\left|R_1\right|\times\left|R_2\right|\right). \end{aligned} \tag{4.12}$$

Locating the cell c with the maximum value in the two-dimensional table and executing the backtrack procedure require $O\left(\sum_{i=1}^{|T_1|}\sum_{j=1}^{|T_2|}\left(\left|t_1[i]\right|\times\left|t_2[j]\right|\right)\right)=O\left(\left|R_1\right|\times\left|R_2\right|\right)$ computational time. Therefore the time complexity of CHSalign is $O\left(\left|R_1\right|\times\left|R_2\right|\right)$. Since only a two-dimensional table is used, the space complexity of CHSalign is $O\left(\left|T_1\right|\times\left|T_2\right|\right)=$ $O\left(\left|R_1\right|\times\left|R_2\right|\right)$.

4.2.4 Data Sets

Popular benchmark datasets such as BRAliBase [100] and Rfam [101] are not suitable for testing CHSalign, since they do not contain coaxial helical stacking information. As a consequence, we manually created two datasets for testing CHSalign and comparing it with related methods. The first dataset, Dataset1, contains 24 RNA 3D structures from the Protein Data Bank (PDB) [67] (see Table 4.1). This dataset was studied and published in [46,68,96], in which all annotations for junctions and coaxial helical stacking were taken from crystallographic structures. Each 3D structure in Dataset1 contains at least one three-way junction, and the lengths of the 3D structures range from 40 nt to 2,958 nt. Some 3D structures contain higher-order junctions such as ten-way junctions with coaxial stacking patterns. The 2D structure of each 3D structure in Dataset1 is obtained with RNAView retrieved from RNA STRAND [102]. The pseudoknots in these structures are removed using the K2N tool [103].

Table 4.1 The 24 RNA Full Structures in Dataset1 Selected from the Protein Data Bank (PDB) to Evaluate the Performance of the Alignment Methods Studied in this Dissertation

	PDB Code	Molecule Name	Length
1	1E8O	Alu domain of the Signal recognition particle (7SL RNA)	50
2	1L9A	Signal recognition particle RNA S domain	126
3	1LNG	Signal recognition particle (7S.S RNA)	97
4	1NBS	Ribonuclease P RNA	119
5	1NKW	23S ribosomal RNA	2884
6	1NYI	Hammerhead ribozyme	40
7	1S72	23S ribosomal RNA	2876
8	1U6B	Group I intron	222
9	1U8D	xpt-pbuX guanine riboswitch aptamer domain	67
10	1UN6	5S ribosomal RNA	122
11	1X8W	Tetrahymena ribozyme RNA (group I intron)	968
12	1Y26	Vibrio vulnificus A-riboswitch	71
13	2A64	Ribonuclease P RNA	298
14	2AVY	16S ribosomal RNA	1530
15	2AW4	23S ribosomal RNA	2958
16	2B57	Guanine riboswitch	65
17	2CKY	Thiamine pyrophosphate riboswitch	154
18	2CZJ	Transfer-messenger RNA (tmRNA)	248
19	2EES	Guanine riboswitch	68
20	2GDI	TPP riboswitch	80
21	2HOJ	THI-box riboswitch	75
22	2J00	16S ribosomal RNA	1687
23	2J01	23S ribosomal RNA	2891
24	2QBZ	M-Box RNA, ykoK riboswitch aptamer	153

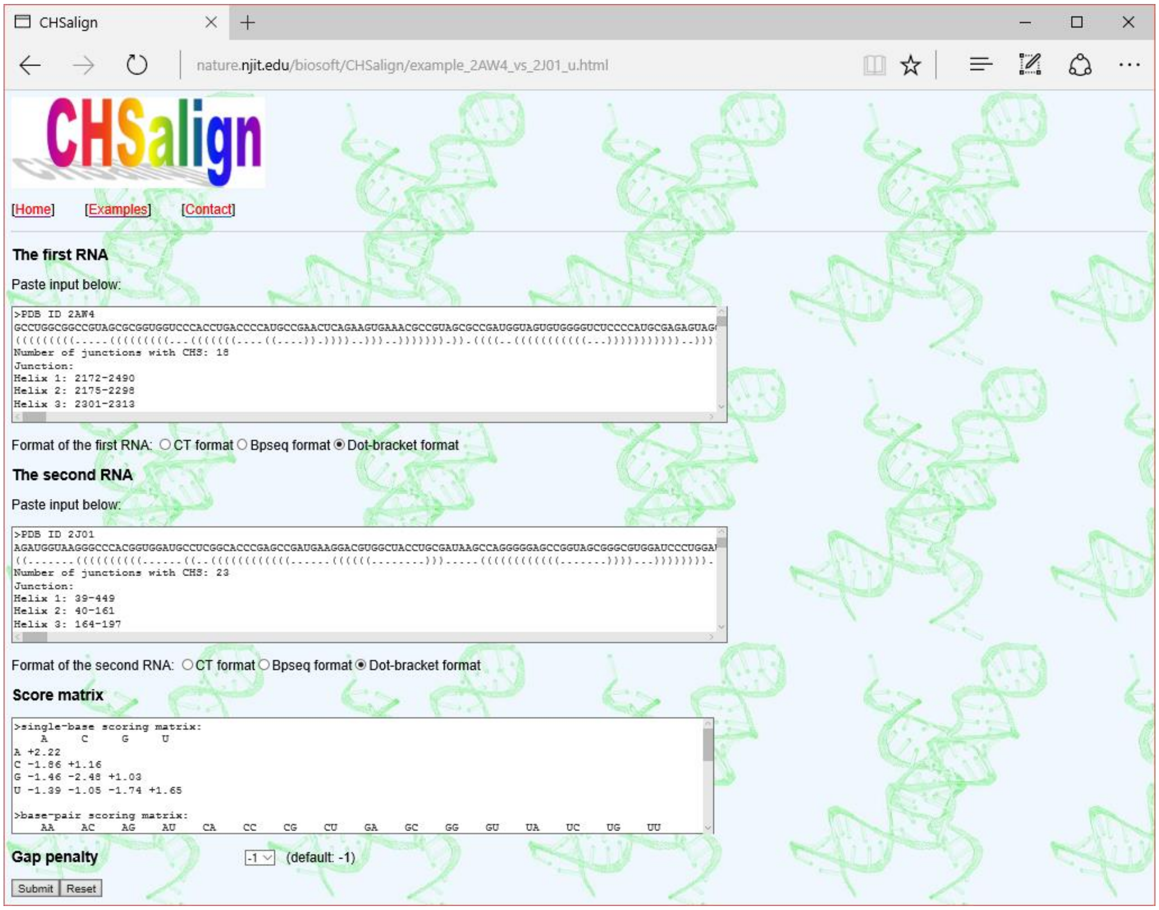
The second dataset, Dataset2, contains 76 three-way junctions extracted from the 24 3D structures in Dataset1. (Some 3D structures in Dataset1 contain more than one three-way junction and all those three-way junctions in a 3D structure are extracted.) The lengths of the three-way junctions range from 28nt to 153nt. The coaxial helical stacking status of each three-way junction in Dataset2 is described as one of three possibilities: $H_1H_{2,}$ $H_2H_{3,}$ $H_1H_{3.}$ Thus, every three-way junction in Dataset2 contains a coaxial stacking pattern. In the RNA literature, most research efforts have been focused on three-way and four-way junctions [48,57,104-106] partly due to the fact that higher-order junctions are rare. In particular, three-way junctions are the most abundant type of junctions, accounting for over 50% of the available crystal data. We also performed experiments on four-way junctions; results obtained from the four-way junctions were similar to those for the three-way junctions reported here, and hence omitted.

4.3 Results and Discussion

4.3.1 Two CHSalign Web Server Versions

We have implemented two programs in Java, a standalone version denoted by CHSalign_u, and the other a pipeline denoted by CHSalign_p. CHSalign_u requires the user to manually annotate the coaxial stacking patterns within junctions of the pair of RNA 2D structures in the input, and produces an optimal alignment between the two input structures.

By contrast, CHSalign_p accepts as input two unannotated RNA secondary structures and produces as output an optimal alignment between the two input structures while taking into account their junctions and coaxial stacking configurations within the junctions. This pipeline invokes our previously developed Junction Explorer tool [68] to automatically predict and identify the junctions and coaxial stacking patterns within the junctions in the input structures, and then aligns the input structures containing the predicted coaxial stacking patterns. Both CHSalign_u and CHSalign_p are available on the web. Figure 4.17 shows an example of CHSalign_u and Figure 4.18 shows the result of the example in Figure 4.19. Figure 4.20 shows an example of CHSalign_p and Figure 4.9 shows the result of the example in Figure 4.8.


CHSalign
nature.njit.edu/biosoft/CHSalign/example_2AW4_vs_2J01_u.html
CHSalign
[Home] [Examples] [Contact]

The first RNA

Paste input below:

```
>PDB ID 2AW4
GCCUGGCGGCCGUAGCGCGGUGGUCCCACCUGACCCCAUGCCGAACUCAGAAGUGAAACGCCGUAGCGCCGAUGGUAGUGUGGGGUCUCCCCAUGCGAGAGUAG
((((((((.....((((((((...((((((....((....)).))))..)))..))))))).)).((((..((((((((((...)))))))))))..)))
Number of junctions with CHS: 18
Junction:
Helix 1: 2172-2490
Helix 2: 2175-2298
Helix 3: 2301-2313
```

Format of the first RNA: ○ CT format ○ Bpseq format ● Dot-bracket format

The second RNA

Paste input below:

```
>PDB ID 2J01
AGAUGGUAAGGGCCCACGGUGGAUGCCUCGGCACCCGAGCCGAUGAAGGACGUGGCUACCUGCGAUAAGCCAGGGGGAGCCGGUAGCGGGCGUGGAUCCCUGGA
((.......((((((((((......((..(((((((((((......((((((.........)))......(((((((((((((.......))))...)))))))))).
Number of junctions with CHS: 23
Junction:
Helix 1: 39-449
Helix 2: 40-161
Helix 3: 164-197
```

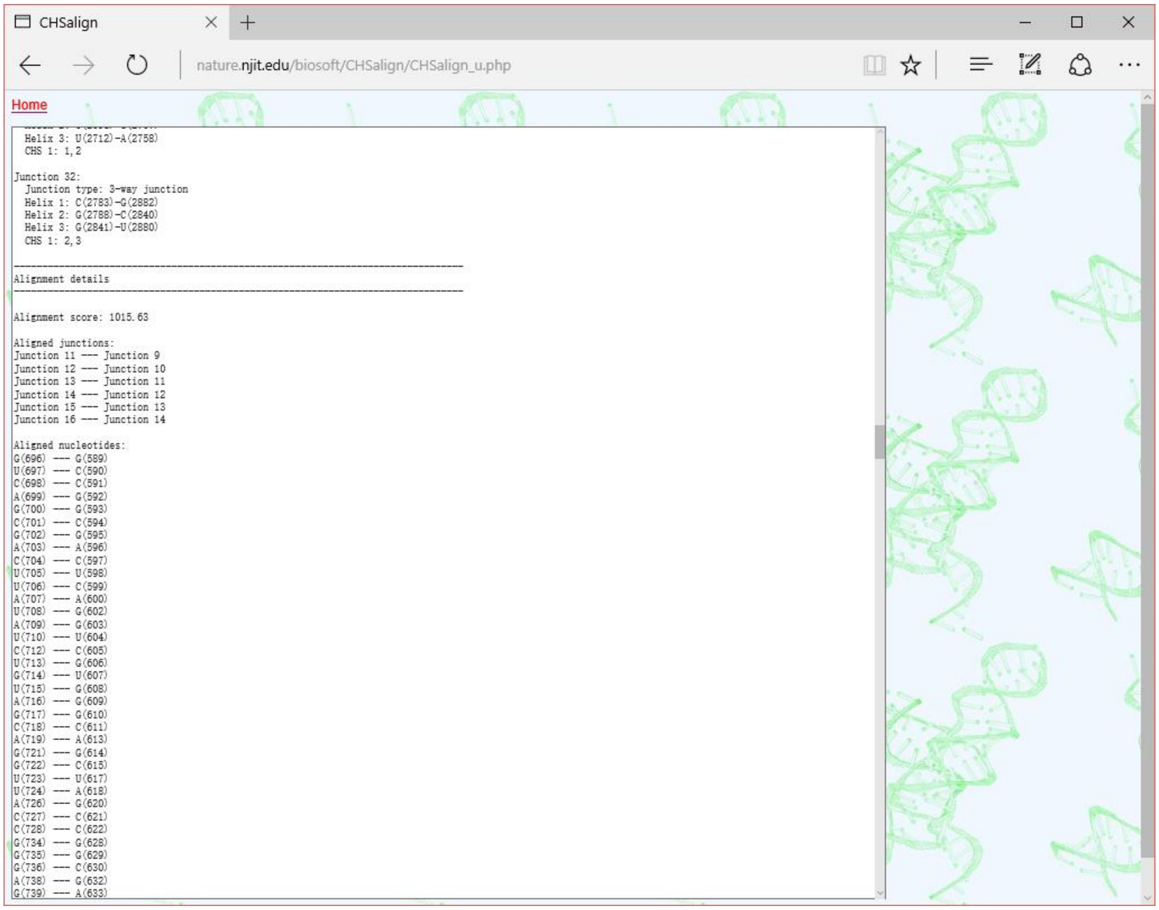
Format of the second RNA: ○ CT format ○ Bpseq format ● Dot-bracket format

Score matrix

```
>single-base scoring matrix:
    A     C     G     U
A +2.22
C -1.86 +1.16
G -1.46 -2.48 +1.03
U -1.39 -1.05 -1.74 +1.65

>base-pair scoring matrix:
    AA    AC    AG    AU    CA    CC    CG    CU    GA    GC    GG    GU    UA    UC    UG    UU
```

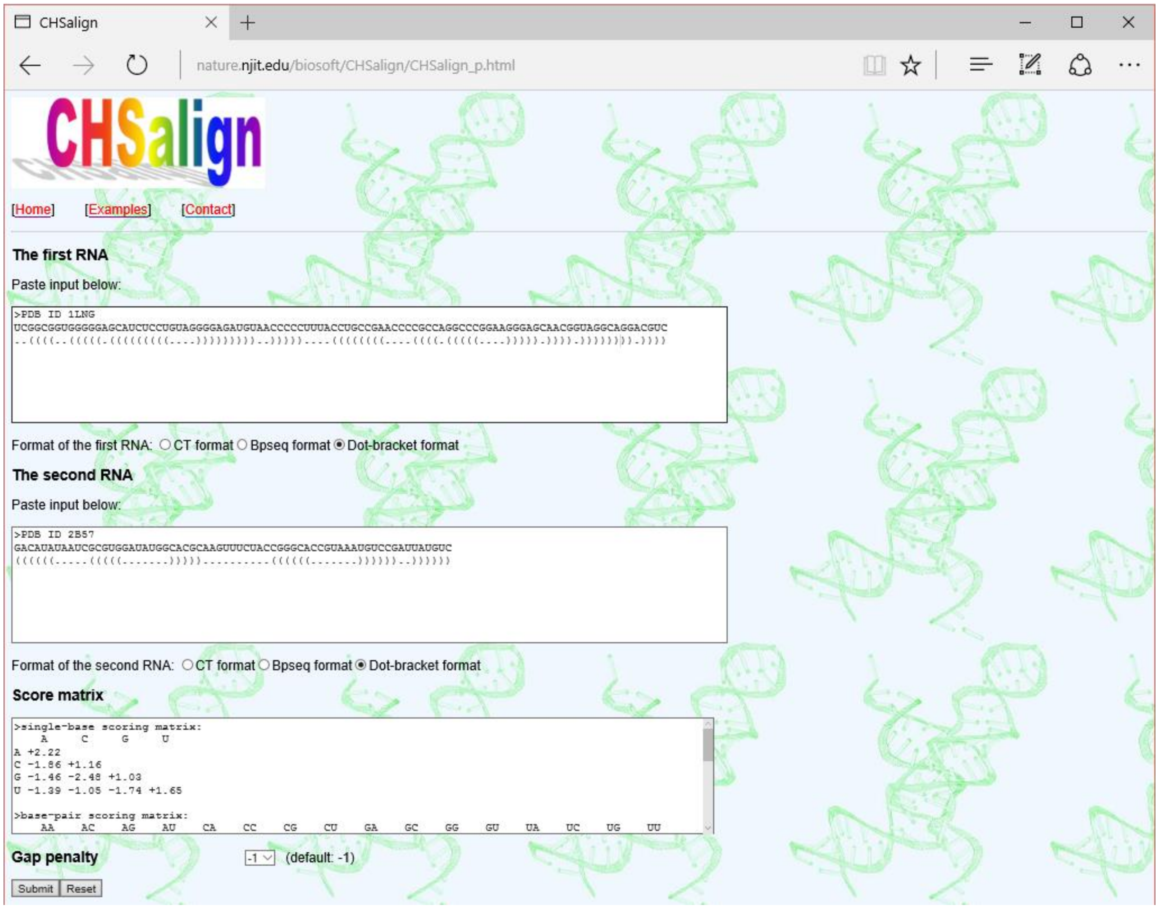
Gap penalty -1 (default: -1)

Figure 4.17 The screenshot of input for CHSalign_u.



```
  Helix 3: U(2712)-A(2758)
  CHS 1: 1,2

Junction 32:
  Junction type: 3-way junction
  Helix 1: C(2783)-G(2882)
  Helix 2: G(2788)-C(2840)
  Helix 3: G(2841)-U(2880)
  CHS 1: 2,3

--------------------------------------------------------------------------------
Alignment details
--------------------------------------------------------------------------------


Alignment score: 1015.63

Aligned junctions:
Junction 11 --- Junction 9
Junction 12 --- Junction 10
Junction 13 --- Junction 11
Junction 14 --- Junction 12
Junction 15 --- Junction 13
Junction 16 --- Junction 14

Aligned nucleotides:
G(696) --- G(589)
U(697) --- C(590)
C(698) --- C(591)
A(699) --- G(592)
G(700) --- G(593)
C(701) --- C(594)
G(702) --- G(595)
A(703) --- A(596)
C(704) --- C(597)
U(705) --- U(598)
U(706) --- C(599)
A(707) --- A(600)
U(708) --- G(602)
A(709) --- G(603)
U(710) --- U(604)
C(712) --- C(605)
U(713) --- G(606)
G(714) --- U(607)
U(715) --- G(608)
A(716) --- G(609)
G(717) --- G(610)
C(718) --- C(611)
A(719) --- A(613)
G(721) --- G(614)
G(722) --- C(615)
U(723) --- U(617)
U(724) --- A(618)
A(726) --- G(620)
C(727) --- C(621)
C(728) --- C(622)
G(734) --- G(628)
G(735) --- G(629)
G(736) --- C(630)
A(738) --- G(632)
G(739) --- A(633)
```

Figure 4.18 The screenshot of the result for CHSalign_u in Figure 4.6.


CHSalign
nature.njit.edu/biosoft/CHSalign/CHSalign_p.html
CHSalign
[Home] [Examples] [Contact]

The first RNA

Paste input below:

```
>PDB ID 1LNG
UCGGCGGUGGGGGAGCAUCUCCUGUAGGGGAGAUGUAACCCCCUUUACCUGCCGAACCCCGCCAGGCCCGGAAGGGAGCAACGGUAGGCAGGACGUC
--((((..(((((.(((((((((....)))))))))..)))))....((((((((....((((.(((((....))))).)))).)))))))).))))
```

Format of the first RNA: ○ CT format ○ Bpseq format ● Dot-bracket format

The second RNA

Paste input below:

```
>PDB ID 2B57
GACAUAUAAUCGCGUGGAUAUGGCACGCAAGUUUCUACCGGGCACCGUAAAUGUCCGAUUAUGUC
((((((.....(((((.......))))).........((((((.......))))))..))))))
```

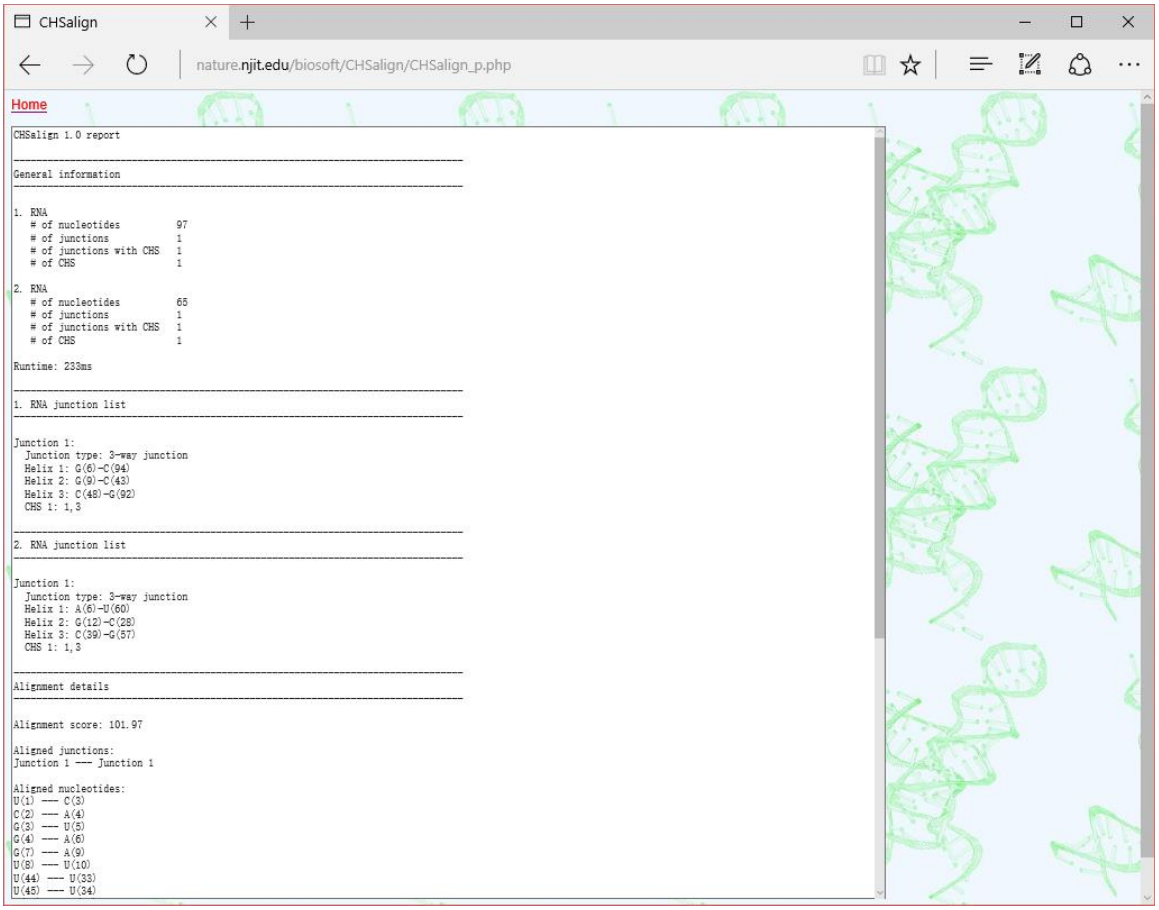
Format of the second RNA: ○ CT format ○ Bpseq format ● Dot-bracket format

Score matrix

```
>single-base scoring matrix:
    A     C     G     U
A +2.22
C -1.86 +1.16
G -1.46 -2.48 +1.03
U -1.39 -1.05 -1.74 +1.65

>base-pair scoring matrix:
    AA    AC    AG    AU    CA    CC    CG    CU    GA    GC    GG    GU    UA    UC    UG    UU
```

Gap penalty -1 (default: -1)

Figure 4.19 The screenshot of CHSalign_p.



```
CHSalign 1.0 report

--------------------------------------------------------------------------------
General information
--------------------------------------------------------------------------------

1. RNA
   # of nucleotides         97
   # of junctions           1
   # of junctions with CHS  1
   # of CHS                 1

2. RNA
   # of nucleotides         65
   # of junctions           1
   # of junctions with CHS  1
   # of CHS                 1

Runtime: 233ms

--------------------------------------------------------------------------------
1. RNA junction list
--------------------------------------------------------------------------------

Junction 1:
  Junction type: 3-way junction
  Helix 1: G(6)-C(94)
  Helix 2: G(9)-C(43)
  Helix 3: C(48)-G(92)
  CHS 1: 1,3

--------------------------------------------------------------------------------
2. RNA junction list
--------------------------------------------------------------------------------

Junction 1:
  Junction type: 3-way junction
  Helix 1: A(6)-U(60)
  Helix 2: G(12)-C(28)
  Helix 3: C(39)-G(57)
  CHS 1: 1,3

--------------------------------------------------------------------------------
Alignment details
--------------------------------------------------------------------------------

Alignment score: 101.97

Aligned junctions:
Junction 1 --- Junction 1

Aligned nucleotides:
U(1) --- C(3)
C(2) --- A(4)
G(3) --- U(5)
G(4) --- A(6)
G(7) --- A(9)
U(8) --- U(10)
U(44) --- U(33)
U(45) --- U(34)
```

Figure 4.20 The result of CHSalign_p in Figure 4.8.

4.3.2 Performance Evaluation Using RMSD

We conducted a series of experiments to evaluate the performance of the algorithms. In the first experiment, we divided Dataset2 into three disjoint subsets Dataset2-1, Dataset2-2 and Dataset2-3, with 35, 18, and 23 junctions, respectively. These three subsets contain, respectively, three-way junctions whose coaxial helical stacking status is H_1H_2, H_2H_3, or H_1H_3. We performed pairwise alignment of junctions in each subset. There are $(35\times34/2 + 18\times17/2 + 23\times22/2) = 1,001$ pairwise alignments produced by CHSalign. Commonly used ways for evaluating the accuracy of these structural alignments include the calculation of distance matrices or RMSD (root-mean-square deviation) [46,74,77,107-111]. We adopt the RMSD measure [46,74] to evaluate the performance of our algorithms; specifically we use the method for computing RMSDs of tree graphs [46]. It has been shown that RMSDs of tree graphs and RMSDs of atomic models are positively correlated and indicate similar trends [46]. The average of the RMSD values of the 1,001 pairwise alignments was calculated and plotted.

One important parameter in our algorithms is the weight w used in Equation (4.3) for calculating the alignment score of two junction nodes. This parameter is introduced to favor the alignment between two junctions with the same number of branches and the same coaxial helical stacking status. Experimental results show that when w is sufficiently large (e.g., $w > 50$), our algorithms work well. In subsequent experiments, we fixed the weight w in Equation (4.3) at 100.

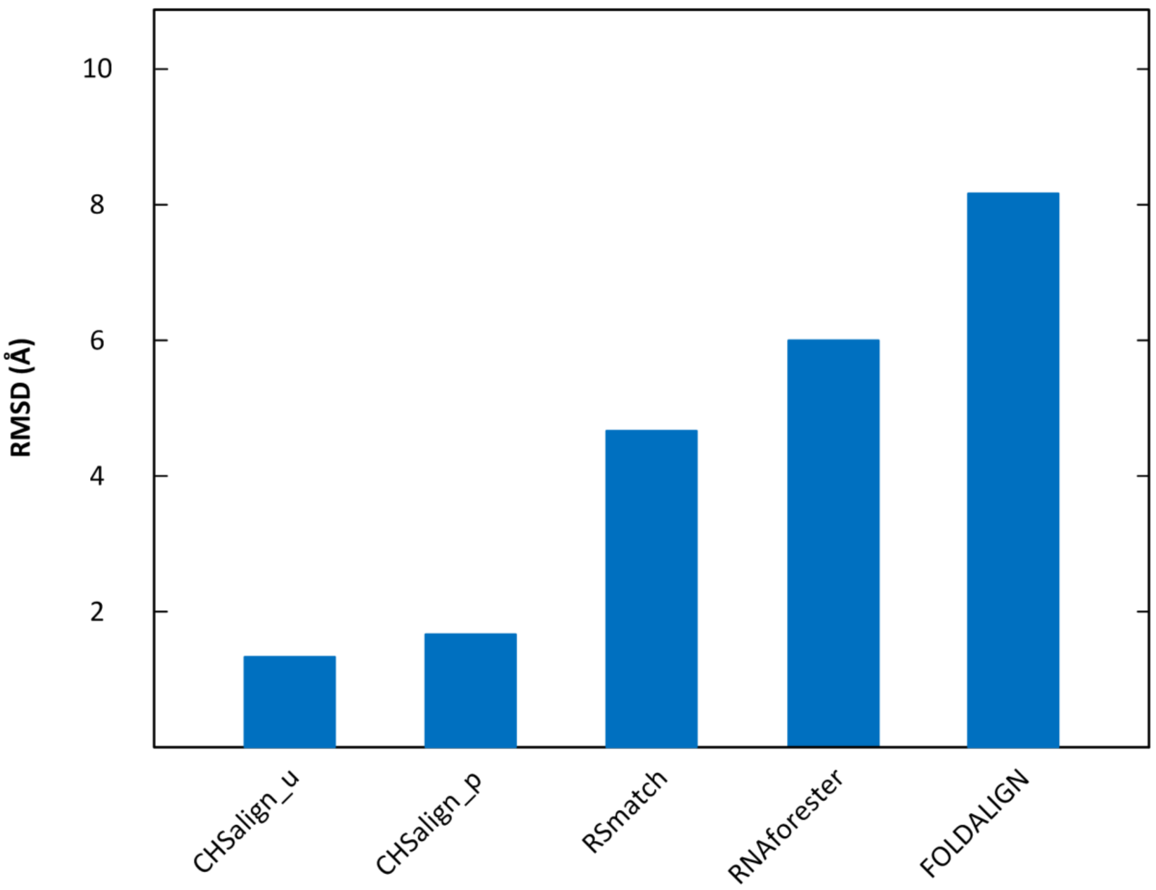


Figure 4.21 Comparison of the RMSD values obtained by CHSalign_u, CHSalign_p, RSmatch, RNAforester and FOLDALIGN.

Figure 4.21 compares CHSalign_u and CHSalign_p with three other alignment programs: RNAforester [83], RSmatch [15] and FOLDALIGN [85]. Like CHSalign, both RNAforester and RSmatch produce an alignment between two input RNA 2D structures. FOLDALIGN differs from the other programs in Figure 4.21 in that it performs 2D structure prediction and alignment simultaneously. When running the FOLDALIGN tool, the structure information in the datasets was ignored and only the sequence data was used as the input of the tool. In addition, when experimenting with CHSalign_u, the coaxial stacking patterns were provided along with the input RNA 2D structures. When running the other programs including CHSalign_p, RNAforester, RSmatch and FOLDALIGN, these coaxial stacking patterns were absent in the input. CHSalign_p automatically predicts the coaxial stacking patterns and then aligns the predicted structures.

Figure 4.21 shows that CHSalign_u performs the best, achieving an RMSD of 1.78 Å. The drawback of CHSalign_u, however, is that it requires the user to annotate the input RNA structures with coaxial stacking patterns manually. Manually annotating coaxial stacking patterns on RNA structures requires domain related expertise. On the other hand, CHSalign_p does not require any manual processing and achieves a reasonably good RMSD of 1.83 Å. Since the predicted coaxial stacking patterns may be imperfect, the RMSD of CHSalign_p is larger than that of CHSalign_u. RSmatch and RNAforester have even larger RMSDs of 4.41 Å and 6.13 Å, respectively. This happens because RSmatch and RNAforester ignore coaxial stacking configurations when aligning RNA 2D structures. FOLDALIGN has the largest RMSD of 8.26 Å, partly because it does not consider coaxial helical stacking either, and partly because there are errors in its predicted 2D structures.

4.3.3 Performance Evaluation Using Precision

In the next experiment, we adopt *precision* as the performance measure, defined below, to evaluate how junctions and coaxial stacking patterns are aligned by different programs using the 24 structures in Dataset1. We say a junction J_1 in structure R_1 is aligned with a junction J_2 in structure R_2, or more precisely there is a junction alignment between J_1 and J_2, if there exist a nucleotide n_1 on a loop region of J_1 and a nucleotide n_2 on a loop region of J_2 such that n_1 is aligned with n_2. A junction alignment between J_1 and J_2 is a true positive if J_1 and J_2 have the same number of branches and the same coaxial helical stacking status. A junction alignment between J_1 and J_2 is a false positive if J_1 and J_2 have different numbers of branches or different coaxial helical stacking statuses. The precision (PR) of an alignment between R_1 and R_2 is defined as

$$PR = TP / \left(TP + FP\right), \tag{4.13}$$

where *TP* equals the number of true positives and *FP* equals the number of false positives in the alignment. The higher PR value a program has, the more precise alignment that program produces. In the experiment, we also included a closely related RNA 3D alignment tool (SETTER) [76].

We calculated the precision of each alignment produced by a program, took the average of the precision values of the pairwise alignments of the 24 structures in Dataset1, and plotted the average values. Figure 4.22 shows the result. We can see that CHSalign_u performs the best, achieving a PR value of 1. CHSalign_p achieves a PR value of 0.85, not 1, because some coaxial stacking patterns were not predicted correctly by Junction Explorer [68] used in CHSalign_p. The other programs in Figure 4.22 did not consider coaxial helical stacking while performing pairwise alignments, and hence achieved low PR values. Specifically, the PR values of RNAforester, SETTER, RSmatch, and FOLDALIGN were 0.54, 0.42, 0.33, and 0.31, respectively. Unlike the CHSalign method, these programs occasionally align two junctions with different numbers of branches or different coaxial helical stacking statuses, hence yielding false positives. However, SETTER is a general-purpose structure alignment tool capable of comparing two RNA 3D molecules with diverse tertiary motifs, while CHSalign can only deal with the 2D structures of the 3D molecules that contain coaxial helical stacking motifs.

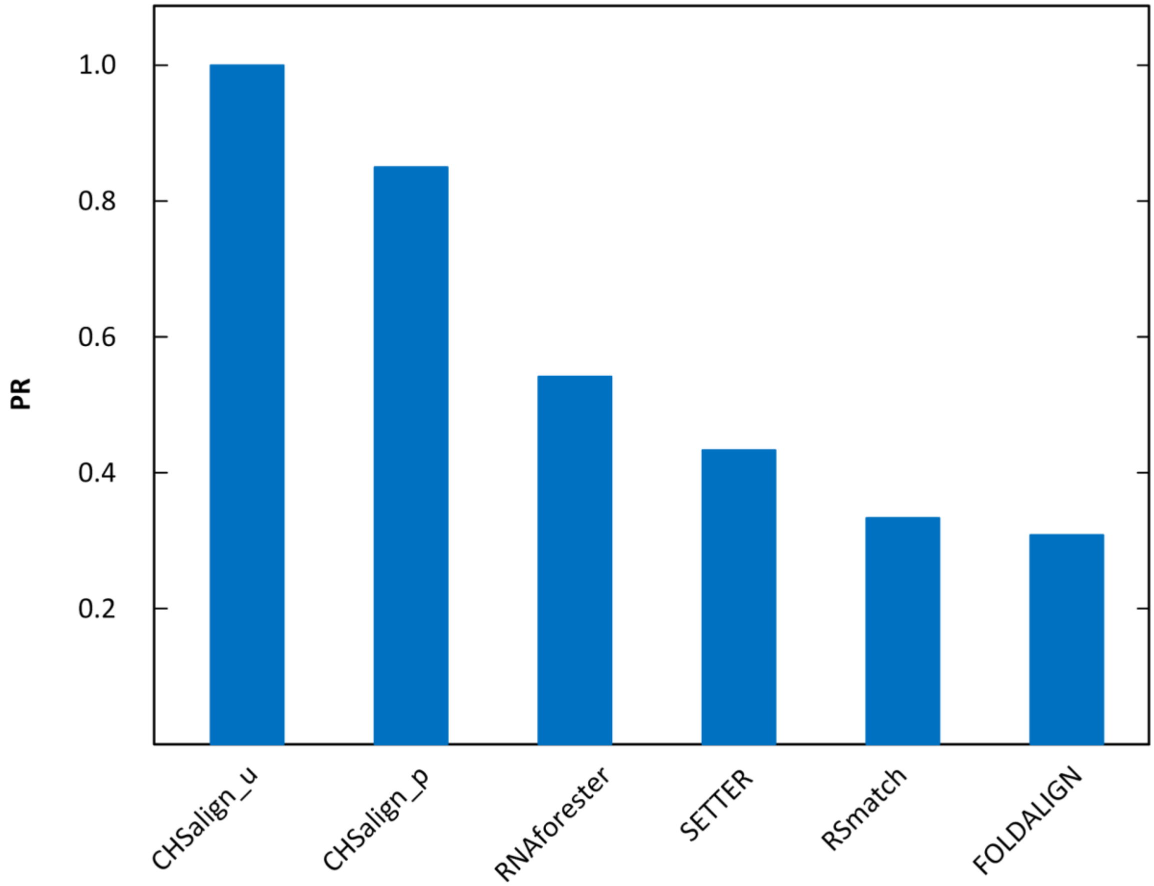


Figure 4.22 Comparison of the PR values obtained by CHSalign_u, CHSalign_p, RNAforester, SETTER, RSmatch and FOLDALIGN.

4.3.4 Potential Application of CHSalign

To demonstrate the utility of the CHSalign tool, we applied CHSalign to the analysis of riboswitches that regulate gene expression by selectively binding metabolites [112]. Table 4.2 lists six riboswitches that bind to different metabolites (purine, guanine, thiamine pyrophosphate [TPP], and S-Adenosyl methionine [SAM]) found in different organisms. Since such binding and gene regulation activities are correlated to junction structures, the results of junction alignments could help suggest structural similarity (and thus possibly function) of these riboswitches. For each riboswitch, Table 4.2 also lists the junction type and coaxial helical stacking status within the junction in that riboswitch. Figure 4.23 illustrates the coaxial stacking patterns in the six riboswitches. Figure 4.23 (A) is Artificial purine riboswitch (PDB code: 2G9C) with a three-way junction and a CHS motif of type H_1H_3 in the junction. Figure 4.23 (B) is Artificial guanine riboswitch (PDB code: 3RKF) with a three-way junction and a CHS motif of type H_1H_3 in the junction. Figure 4.23 (C) is *A. thaliana* TPP riboswitch (PDB code: 3D2G) with a three-way junction and a CHS motif of type H_1H_2 in the junction. Figure 4.23 (D) is *E. coli* TPP riboswitch (PDB code: 2GDI) with a three-way junction and a CHS motif of type H_1H_2 in the junction. Figure 4.23 (E) is *T. tengcongensis* SAM-I riboswitch (PDB code: 2GIS) with a four-way junction and a CHS motif of type H_1H_4, H_2H_3 in the junction. Figure 4.23 (F) is *H. marismortui* SAM-I riboswitch (PDB code: 4B5R) with a four-way junction and a CHS motif of type H_1H_4, H_2H_3 in the junction. We tested several combinations of junctions in these six riboswitches to determine whether the CHSalign results confirm known structural and functional similarity in existing RNAs. Table 4.3 summarizes the test results.

Table 4.2 The Six Riboswitches Selected from the Protein Data Bank (PDB) to Demonstrate the Utility of Our Web Server.

	PDB Code	Molecule Name	Length	Junction	CHS
1	2G9C	Artificial purine riboswitch	68	3-way	H_1H_3
2	3RKF	Artificial guanine riboswitch	68	3-way	H_1H_3
3	3D2G	*A. thaliana* TPP riboswitch	77	3-way	H_1H_2
4	2GDI	*E. coli* TPP riboswitch	80	3-way	H_1H_2
5	2GIS	*T. tengcongensis* SAM-I riboswitch	95	4-way	H_1H_4 H_2H_3
6	4B5R	*H. marismortui* SAM-I riboswitch	95	4-way	H_1H_4 H_2H_3

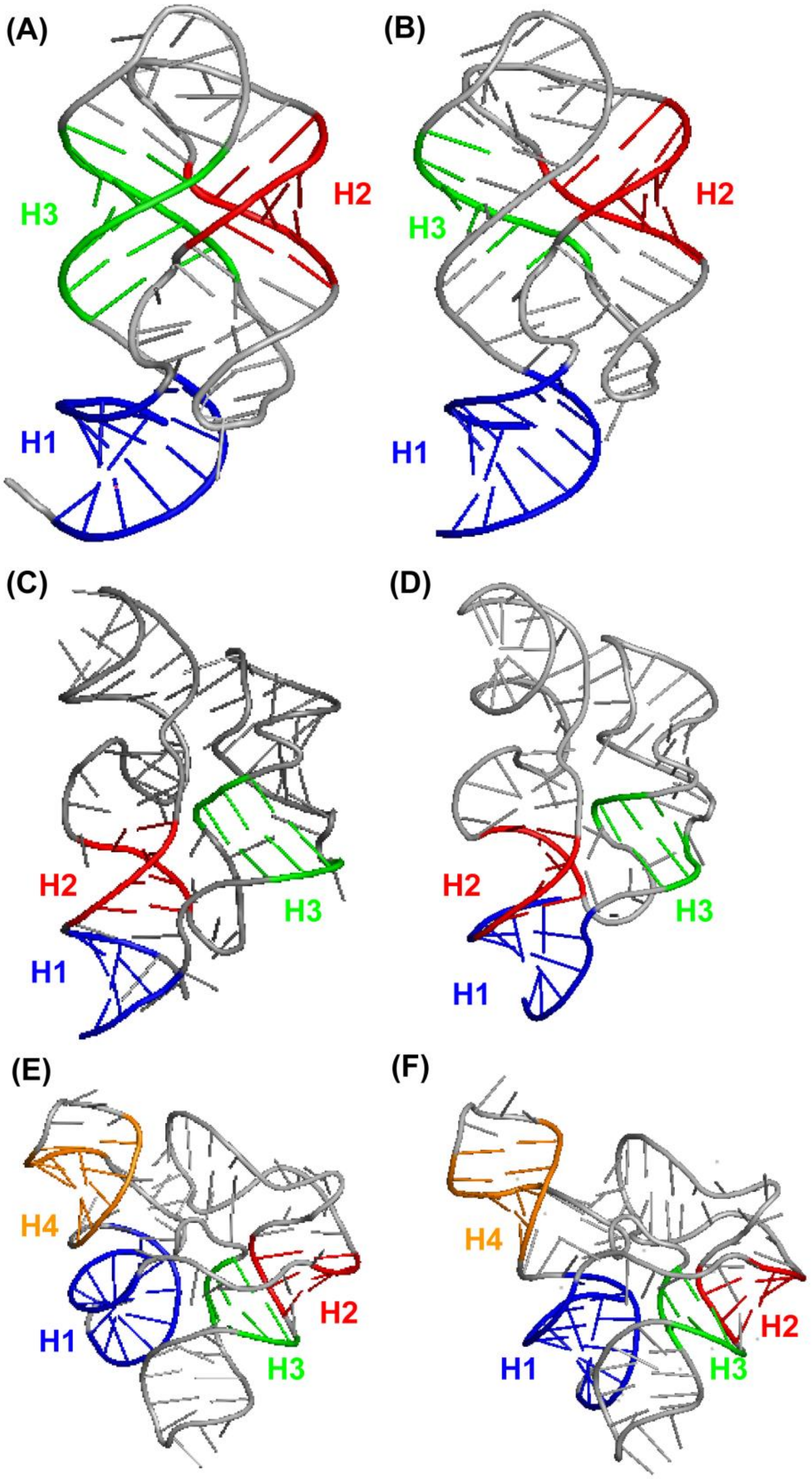


Figure 4.23 Illustration of the coaxial stacking patterns in the six riboswitches used to demonstrate the utility of our web server.

Table 4.3 Results Obtained by Aligning Seven Pairs of Riboswitches from Table 4.2.

Program	Molecule 1	Molecule 2	Alignment Score
CHSalign_p	2GIS *T. tengcongensis* SAM-I riboswitch ($H_1H_{4,}$ H_2H_3)	4B5R *H. marismortui* SAM-I riboswitch ($H_1H_{4,}$ H_2H_3)	252.61
CHSalign_p	2G9C Artificial purine riboswitch (H_1H_3)	3RKF Artificial guanine riboswitch (H_1H_3)	179.68
CHSalign_p	2GDI *E. coli* TPP riboswitch (H_1H_2)	3D2G *A. thaliana* TPP riboswitch (H_1H_2)	191.06
CHSalign_p	2GIS *T. tengcongensis* SAM-I riboswitch ($H_1H_{4,}$ H_2H_3)	2G9C Artificial purine riboswitch (H_1H_3)	20.40
CHSalign_p	2G9C Artificial purine riboswitch (H_1H_3)	2GDI *E. coli* TPP riboswitch (H_1H_2)	13.65
CHSalign_u	2G9C Artificial purine riboswitch (H_1H_3)	2G9C Artificial purine riboswitch (H_1H_2)	36.69
CHSalign_u	2G9C Artificial purine riboswitch (H_1H_2)	3RKF Artificial guanine riboswitch (H_1H_2)	179.68

Without knowledge of junction helical arrangements, we first tested the following cases using CHSalign_p, where the two aligned junctions had the same coaxial stacking patterns. We used SAM riboswitches in different organisms (PDB codes 2GIS and 4B5R in Table 4.2) as input. CHSalign_p predicted that the two riboswitches had helical arrangements of four-way junctions both with coaxial stacking helices 1 and 4 and helices 2 and 3, and produced a very high alignment score of 252.61, as calculated by the equations in the subsection ‘Alignment scheme” in the section ‘Materials and Methods’. This high score implies that the two riboswitches have highly similar helical arrangements. This corroborates our expectations, because the two tested riboswitches have similar structures and functionality, binding to SAM. Next, when we used purine and guanine riboswitches

(PDB codes 2G9C and 3RKF), we obtained a high alignment score of 179.68 for three-way junction alignment of the two witches with predicted coaxial stacking of helices 1 and 3 in both riboswitches, indicating high similarities of their three-way junction structures. We also tested two TPP riboswitches with three-way junctions in different organisms (PDB codes 2GDI and 3D2G), which produced a high alignment score of 191.06, again indicating that these two TPP riboswitches have similar three-way junction structures.

We next compared very different junction structures using CHSalign_p. When we aligned two different riboswitches – SAM riboswitch with a four-way junction and purine riboswitch with a three-way junction (PDB codes 2GIS and 2G9C, respectively), we obtained a low alignment score of 20.40. We also tested a pair of purine and TPP riboswitches (PDB codes 2G9C and 2GDI), which are in different riboswitch classes and have different coaxial stacking patterns in their three-way junctions. We obtained a low alignment score of 13.65. These experiments suggest that CHSalign_p, based only on secondary structural information, is useful for inferring tertiary structural features regarding helical arrangements.

Finally, we tested CHSalign_u, which requires prior information about junction arrangement and produces a structural similarity score for two given RNAs. Here, we tested two cases. First, we considered the same RNA structure (purine riboswitch with PDB code 2G9C) but annotated it with different helical arrangement patterns where one had coaxial stacking helices 1 and 3 (H_1H_3) and the other had coaxial stacking helices 1 and

2 (H_1H_2). Second, we considered two RNAs with different structures (purine riboswitch with PDB code 2G9C and guanine riboswitch with PDB code 3RKF, respectively) but annotated them with the same helical arrangement pattern, namely coaxial stacking helices 1 and 2 (H_1H_2). Note that this manually annotated H_1H_2 pattern is different from the H_1H_3 pattern that naturally occurs, and is also predicted by CHSalign_p, in the purine and guanine riboswitches.

In the first case, the score produced by CHSalign_u was very low (36.69), due to the different helical arrangements. This result shows the large conformational range of structural arrangements that the purine riboswitch can have, from naturally preferable arrangements (H_1H_3, as predicted by CHSalign_p) to unnatural arrangements (H_1H_2, as manually set by us). In the second case, CHSalign_u produced a high score of 179.68, which indicates the possibility that two different RNA structures can have very similar helical arrangements when we manually set these arrangements. Thus, CHSalign_u could help investigate the structural diversity of all possible helical arrangements, including natural or hypothetical conformations for two RNA 2D structures.

4.4 Conclusions

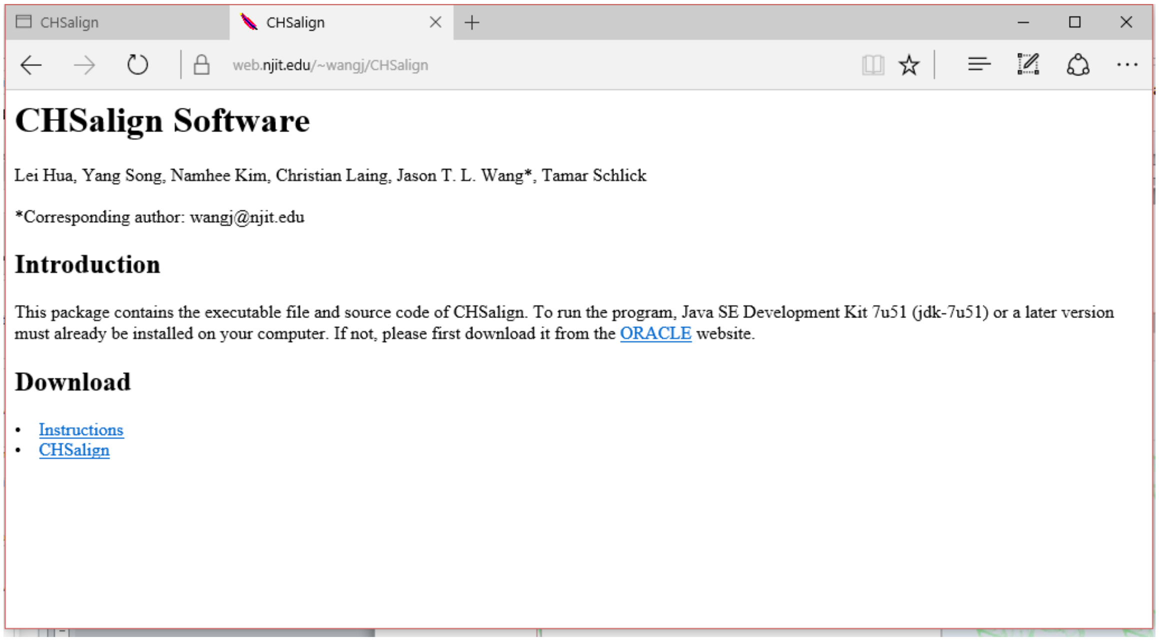
We have presented a novel method (CHSalign) capable of producing an optimal alignment between two input RNA secondary (2D) structures with coaxial helical stacking, based on the previously developed Junction Explorer [68] and RNAJAG [46]. The method is junction-aware, CHS-favored in the sense that it assigns a weight to the alignment of two RNA junctions with the same number of branches and the same coaxial helical stacking status while prohibiting the alignment of two junctions that do not have the same number of branches or the same coaxial helical stacking status. The method transforms each input RNA 2D structure to an ordered labeled tree, and employs dynamic programming techniques and a constrained tree matching algorithm to align the two input RNA 2D structures. CHSalign has two versions; CHSalign_u requires the user to manually annotate the coaxial stacking patterns in the input structures while CHSalign_p automatically predicts the coaxial stacking patterns in the input structures. Experimental results demonstrate that both versions outperform the existing alignment programs that do not take into account coaxial stacking configurations in the input RNA structures.

It has been observed that several functional RNA families such as tRNA, RNase P, and large ribosomal subunits have conserved structural features while having very diverse sequence patterns. RNA structure alignment tools such as CHSalign can help measure the structural similarity between these RNAs, even without sequence relevance in the RNAs. Similar RNA structural motifs are encountered on a variety of RNAs. While these motifs exist in different contexts, their functions are related. For instance, sarcin-ricin motifs often bind to proteins, and GNRA tetraloops act as receptors for RNA-RNA long-range interactions. Furthermore, examples of larger structure-function similarity are observed in the tRNA-like structure found in the transfer-messenger RNA (tmRNA), whose structure similarity with tRNA helps identify the functional role of tmRNAs to aid in translation via stalled ribosome rescue. Other tRNA-like structures found in viruses such as HIV and internal ribosome entry sites (IRES) mimic the 3D “L-shape” of tRNAs to take control of the host ribosome.

As our knowledge on RNA structure progresses, more sophisticated secondary structure alignment tools are required that allow for comparison of tertiary motifs such as coaxial stacking patterns. Indeed, experimental probing techniques such as RNA SHAPE chemistry, SAXS, NMR, and fluorescence resonance energy transfer (FRET), can often provide sufficient information to determine coaxial helical stacking [91,113,114]. Because the structure and function of RNA are highly interrelated, a tool that addresses coaxial stacking patterns can assist the comparison of structures with high functional relevance.

CHSalign is the first tool that can compute an RNA secondary structure alignment in the presence of coaxial helical stacking. When coaxial stacking configurations are available from experimental data such as FRET, NMR or SAXS data, the user can input such information to aid in the alignment. However, if no knowledge of coaxial stacking configurations is available, CHSalign can infer this information by employing Junction Explorer [68], which predicts coaxial helical stacking with 81% accuracy.

Existing RNA secondary structure alignment tools [15,83] do not distinguish between structural elements such as helices, junctions and hairpin loops. However, each element type has its special property and function. In contrast, CHSalign only matches structural elements of the same type. Furthermore, the tool imposes a constraint that a junction of RNA1 can be aligned with a junction of RNA2 only if they have the same number of branches and the same coaxial helical stacking status. We also implemented an extension of CHSalign, which relaxes this constraint. This extension is able to align two junctions with different numbers of branches and simply requires that coaxially stacked helices be aligned with coaxially stacked helices when matching a p-way junction with a q-way junction for p different than q. The source code of both CHSalign and its extension can be downloaded from the web server site. Figure 4.24 shows the download page.



CHSalign Software

Lei Hua, Yang Song, Namhee Kim, Christian Laing, Jason T. L. Wang*, Tamar Schlick

*Corresponding author: wangj@njit.edu

Introduction

This package contains the executable file and source code of CHSalign. To run the program, Java SE Development Kit 7u51 (jdk-7u51) or a later version must already be installed on your computer. If not, please first download it from the ORACLE website.

Download

- Instructions
- CHSalign

Figure 4.24 The download page of CHSalign.

CHAPTER 5

CONCLUSIONS

5.1 Summary for DiscoverR

In the first part of this dissertation, we presented a new quadratic-time dynamic programming algorithm, called DiscoverR, for pattern mining in RNAs. There are many potential applications suitable for DiscoverR. We presented two applications in this dissertation:

The first application is finding repeated regions in an RNA secondary structure. Most previous work focused on detecting repeats in sequences (Sokol 2007), (Wexler 2005). In contrast, DiscoverR is able to locate structural repeats or repeated regions in an RNA secondary structure.

The other application is the discovery of conserved RNA secondary structures in the human genome. By examining how the discovered structures differ from the results obtained from other studies that were recently carried out to search conserved RNA secondary structures in the human genome (Washietl 2005), (Pedersen 2006), (Khaladkar 2008), one can conclude that DiscoverR not only is a powerful tool for RNA motif discovery, but also presents unique searching capability that other current algorithms cannot provide. What's more exciting here is that this research finding indicates there may exist much more conserved RNA secondary structures in the human genome that remain to be explored. And DiscoverR can play a critical role.

5.2 Summary for CHSalign

In the second part of this dissertation, we have presented a novel method, called CHSalign, which is capable of producing an optimal alignment between two input RNA secondary (2D) structures with coaxial helical stacking. This method transforms each input RNA 2D structure to an ordered labeled tree, and employs dynamic programming techniques and a constrained tree matching algorithm to align the two input RNA 2D structures. The algorithm also assigns a weight to the alignment of two RNA junctions with the same number of branches and the same coaxial helical stacking status while prohibiting the alignment of two junctions that do not have the same number of branches or the same coaxial helical stacking status. There are two versions of CHSalign: CHSalign_u, which requires the user to manually annotate the coaxial stacking patterns in the input structures, and CHSalign_p, which automatically predicts the coaxial stacking patterns in the input structures. Experimental results demonstrate that both versions outperform the existing alignment programs that do not take into account coaxial stacking configurations in the input RNA structures.

Scientists have found that several functional RNA families have conserved structural features while having very diverse sequence patterns. Similar RNA structural motifs are encountered on a variety of RNAs. While these motifs exist in different contexts, their functions are related. RNA structure alignment tools such as CHSalign can help measure the structural similarity between these RNAs, even without sequence relevance in the RNAs.

As our knowledge on RNA structure progresses, more sophisticated secondary structure alignment tools are required that allow for comparison of tertiary motifs such as coaxial stacking patterns. Indeed, experimental probing techniques such as RNA SHAPE chemistry, SAXS, NMR, and fluorescence resonance energy transfer (FRET), can often provide sufficient information to determine coaxial helical stacking [91,113,114]. Because the structure and function of RNA are highly interrelated, a tool that addresses coaxial stacking patterns can assist the comparison of structures with high functional relevance.

CHSalign is the first tool that can compute an RNA secondary structure alignment in the presence of coaxial helical stacking. When coaxial stacking configurations are available from experimental data such as FRET, NMR or SAXS data, the user can input such information to aid in the alignment. However, if no knowledge of coaxial stacking configurations is available, CHSalign can infer this information by employing Junction Explorer [68], which predicts coaxial helical stacking with 81% accuracy.

Existing RNA secondary structure alignment tools [15,83] do not distinguish between structural elements such as helices, junctions and hairpin loops. However, each element type has its special property and function. In contrast, CHSalign only matches structural elements of the same type. Furthermore, the tool imposes a constraint that a junction of RNA1 can be aligned with a junction of RNA2 only if they have the same number of branches and the same coaxial helical stacking status. We also implemented an extension of CHSalign, which relaxes this constraint. This extension is able to align two junctions with different numbers of branches and simply requires that coaxially stacked helices be aligned with coaxially stacked helices when matching a p-way junction with a q-way junction for p different than q. The source code of both CHSalign and its extension can be downloaded from the web server site.

5.3 Future Work

Junctions with coaxial helical stacking are very important motifs in RNA. In CHSalign, the RNA secondary structure is transformed to an ordered labeled tree. However, some features of the tree model become insufficient when aligning RNA tertiary structures. In the future, we plan to develop new graph models to tackle alignment problems of RNA structures with more complicated tertiary motifs. In addition, we plan to design and implement new graph mining algorithms capable of finding biologically significant patterns in the complex tertiary structures.

BIBLIOGRAPHY

1. Altschul S.F., Gish W., Miller W., Myers E.W., Lipman D.J. (1990) Basic local alignment search tool. J Mol Biol 215: 403-410.

2. Pearson W.R., Lipman D.J. (1988) Improved tools for biological sequence comparison. In: Proc Natl Acad Sci USA. pp. 2444-2448.

3. Akmaev V.R., Kelley S.T., Stormo G.D. (1999) A phylogenetic approach to RNA structure prediction. In: Proc Int Conf Intell Syst Mol Biol pp. 10-17.

4. Gulko B., Haussler D. (1996) Using multiple alignments and phylogenetic trees to detect RNA secondary structure. Pac Symp BioComput: 350-367.

5. Hofacker I.L., Fekete M., Stadler P.F. (2002) Secondary structure prediction for aligned RNA sequences. J Mol Biol 319: 1059-1066.

6. Knudsen B., Hein J. (2003) Pfold: RNA secondary structure predection using stochastic context-free grammars. Nucleic Acids Res 31: 3423-3428.

7. Zuker M. (1989) Computer prediction of RNA structure. Methods Enzymol 180: 262-288.

8. Zuker M. (1989) On finding all suboptimal foldings of an RNA molecule. Science 244: 48-52.

9. Hofacker I.L. (2003) Vienna RNA secondary structure server. Nucleic Acids Res 31: 3429-3431.

10. Schuster P., Fontana, W., Stadler, P. F., and Hofacker, I. L. (1994) From sequences to shapes and back: a case study in RNA secondary structures. Proc Biol Sci 255: 279-284.

11. Rivas E., Eddy S.R. (1999) A dynamic programming algorithm for RNA structure prediction including pseudoknots. J Mol Biol 285: 2053-2068.

12. Rijk P.D., Wuyts, J., and Wachter, R. D. (2003) RnaViz2: an improved representation of RNA secondary structure. Bioinformatics 19: 299-300.

13. Ambros V., Bartel, B., Bartel, D. P., Berge, C. B., Carrington, J. C., Chen, X., Dreyfuss, G., Eddy, S. R., Griffiths-Jones, S., Marshall, M., Matzke, M., Ruvkun, G., and Tuschl T. (2003) A uniform system for microRNA annotation. RNA 9: 277-279.

14. Griffiths-Jones S., Bateman A., Marshall M., Khanna A., Eddy S.R. (2003) Rfam: an RNA family database. Nucleic Acids Res 31: 439-441.

15. Liu J., Wang J.T.L., Hu J., Tian B. (2005) A method for aligning RNA secondary structures and its application to RNA motif detection. BMC Bioinformatics 6: 89.

16. Pesole G., Liuni S., Grillo G., Licciulli F., Mignone F., Gissi C., et al. (2002) UTRdb and UTRsite: specialized databases of sequences and functional elements of 5' and 3' untranslated regions of eukaryotic mRNAs. Nucleic Acids Res 30: 335-340.

17. Hofacker I.L., Stadler, P. F., and Stocsits, R. R. (2004) Conserved RNA secondary structures in viral genomes: a survey. Bioinformatics 20: 1495-1599.

18. Kuersten S., Goodwin E.B. (2003) The power of 3'UTR: translational control and development. Nat Rev Genet 4: 626-637.

19. Mazumder B., Seshadri V., Fox P.L. (2003) Translational control by the 3'-UTR: the ends specify the means. Trends Biochem Sci 28: 91-98.

20. Thurner C., Witwer C., Hofacker I.L., Stadler P.F. (2004) Conserved RNA secondary structures in Flaviviridae genomes. J Gen Virol 85: 1113-1124.

21. Washietl S., Hofacker I.L., Lukasser M., Huttenhofer A., Stadler P.F. (2005) Mapping of conserved RNA secondary structures predicts thousands of functional noncoding RNAs in the human genome. Nat Biotechnol 23: 1383-1390.

22. Khaladkar M., Liu J., Wen D., Wang J.T., Tian B. (2008) Mining small RNA structure elements in untranslated regions of human and mouse mRNAs using structure-based alignment. BMC Genomics 9: 189.

23. Wang J.T.L., Shapiro, B.A., Shasha, D., Zhang, K., and Currey, K. M. (1998) An Algorithm for Finding the Largest Approximately Common Substructures of Two Trees. IEEE TPAMI 20: 889-895.

24. Zuker M. (2003) Mfold web server for nucleic acid folding and hybridization prediction. Nucleic Acids Res 31: 3406-3415.

25. Spirollari J., Wang J.T., Zhang K., Bellofatto V., Park Y., Shapiro B.A. (2009) Predicting consensus structures for RNA alignments via pseudo-energy minimization. Bioinform Biol Insights 3: 51-69.

26. Zhang K., Shasha D. (1989) Simple fast algorithms for the editing distance between trees and related problems. SIAM J Comput 18: 1245-1262.

27. Hua L., Wang, J. T.L., Ji, X., Malhotra, A., Khaladkar, M., Shapiro, B. A., and Zhang, K. (2012) A method for discovering common patterns from two RNA secondary structures and its application to structural repeat detection. J Bioinform Comput Biol 10: 1250001.

28. Staple D.W., and Butcher, S. E. (2005) Pseudoknots: RNA structures with diverse functions. PLoS Biology 3: e213.

29. Mathews D.H., Banerjee, A. R., Luan, D. D., Eickbush, T. H., and Turner, D. H. (1997) Secondary structure model of the RNA recognized by the reverse transcriptase from the R2 retrotransposable element. RNA 3: 1-16.

30. Backofen R., Siebert S.R. (2007) Fast detection of common sequence structure patterns in RNAs. J Discrete Algorithm 5: 212-228.

31. Höchsmann M., Töller, T., Giegerich, R., and Kurtz, S. (2003) Local similarity in RNA secondary structures. In: Proc IEEE Comput Soc Bioinform Conf. pp. 159-168.

32. Mauri G., Pavesi, G. (2005) Algorithms for pattern matching and discovery in RNA secondary structure. Theor Comput Sci 335: 29-51.

33. Sokol D., Benson, G., Tojeira, J. (2007) Tandem repeats over the edit distance. Bioinformatics 23: e30-35.

34. Wexler Y., Yakhini Z., Kashi Y., Geiger D. (2005) Finding approximate tandem repeats in genomic sequences. J Comput Biol 12: 928-942.

35. Mankodi A., Takahashi M.P., Jiang H., Beck C.L., Bowers W.J., Moxley R.T., et al. (2002) Expanded CUG repeats trigger aberrant splicing of ClC-1 chloride channel pre-mRNA and hyperexcitability of skeletal muscle in myotonic dystrophy. Mol Cell 10: 35-44.

36. McLaughlin B.A., Spencer C., Eberwine J. (1996) CAG trinucleotide RNA repeats interact with RNA-binding proteins. Am J Hum Genet 59: 561-569.

37. Brook J.D., McCurrach, M. E., Harley, H. G., Buckler, A. J., Church, D., Aburatani, H., Hunter, K., Stanton, V. P., Thirion, J. P., Hudson, T., Sohn, R., Zemelman, B., Snell, R. G., Rundle, S. A., Crow, S., Davies, J., Shelbourne, P., Buxton, J., Jones, C., Juvonen, V., Johnson, K., Harper, P. S., Shaw, D. J., and Housman, D. E. (1992) Molecular basis of myotonic dystrophy: expansion of a trinucleotide (CTG) repeat at the 39 end of a transcript encoding a protein kinase family member. Cell 68: 799-808.

38. Tian B., White, R. J., Xia, T., Welle, S., Turner, D. H., Mathews, M. B., and Thornton, C. A. (2000) Expanded CUG repeat RNAs form hairpins that activate the double-stranded RNA-dependent protein kinase PKR. RNA 6: 79-87.

39. Pedersen J.S., Bejerano G., Siepel A., Rosenbloom K., Lindblad-Toh K., Lander E.S., et al. (2006) Identification and classification of conserved RNA secondary structures in the human genome. PLoS Comput Biol 2: e33.

40. Torarinsson E., Sawera, M., Havgaard, J. H., Fredholm, M., and Gorodkin, J. (2006) Thousands of corresponding human and mouse genomic regions unalignable in primary sequence contain common RNA structure. Genome Res 16: 885-889.

41. Pruitt K.D., Maglott D.R. (2001) RefSeq and LocusLink: NCBI gene-centered resources. Nucleic Acids Res 29: 137-140.

42. Hua L., Cervantes-Cervantes, M., and Wang, J. T.L. (2011) New Approach to the Discovery of RNA Structural Elements in the Human Genome. In: Laura Elnitski H.P., and Lonnie R. Welch, editor. Advances in Genomic Sequence Analysis and Pattern Discovery. Singapore: World Scientific Publishing Company. pp. 117-132.

43. Brimacombe R., Stiege W. (1985) Structure and function of ribosomal RNA. Biochem J 229: 1-17.

44. Woychik N.A., Hampsey M. (2002) The RNA polymerase II machinery: structure illuminates function. Cell 108: 453-463.

45. Zhong X., Tao X., Stombaugh J., Leontis N., Ding B. (2007) Tertiary structure and function of an RNA motif required for plant vascular entry to initiate systemic trafficking. EMBO J 26: 3836-3846.

46. Laing C., Jung S., Kim N., Elmetwaly S., Zahran M., Schlick T. (2013) Predicting helical topologies in RNA junctions as tree graphs. PLoS One 8: e71947.

47. Bindewald E., Hayes R., Yingling Y.G., Kasprzak W., Shapiro B.A. (2008) RNAJunction: a database of RNA junctions and kissing loops for three-dimensional structural analysis and nanodesign. Nucleic Acids Res 36: D392-397.

48. Ouellet J., Melcher S., Iqbal A., Ding Y., Lilley D.M. (2010) Structure of the three-way helical junction of the hepatitis C virus IRES element. RNA 16: 1597-1609.

49. Lilley D.M., Clegg R.M., Diekmann S., Seeman N.C., Von Kitzing E., Hagerman P.J. (1995) A nomenclature of junctions and branchpoints in nucleic acids. Nucleic Acids Res 23: 3363-3364.

50. Liu L., Chen S.J. (2012) Coarse-grained prediction of RNA loop structures. PLoS One 7: e48460.

51. Popovic M., Nelson J.D., Schroeder K.T., Greenbaum N.L. (2012) Impact of base pair identity 5' to the spliceosomal branch site adenosine on branch site conformation. RNA 18: 2093-2103.

52. Yuan F., Griffin L., Phelps L., Buschmann V., Weston K., Greenbaum N.L. (2007) Use of a novel Forster resonance energy transfer method to identify locations of site-bound metal ions in the U2-U6 snRNA complex. Nucleic Acids Res 35: 2833-2845.

53. Scott W.G., Murray J.B., Arnold J.R., Stoddard B.L., Klug A. (1996) Capturing the structure of a catalytic RNA intermediate: the hammerhead ribozyme. Science 274: 2065-2069.

54. Batey R.T., Gilbert S.D., Montange R.K. (2004) Structure of a natural guanine-responsive riboswitch complexed with the metabolite hypoxanthine. Nature 432: 411-415.

55. Kieft J.S., Zhou K., Grech A., Jubin R., Doudna J.A. (2002) Crystal structure of an RNA tertiary domain essential to HCV IRES-mediated translation initiation. Nat Struct Biol 9: 370-374.

56. Holbrook S.R. (2008) Structural principles from large RNAs. Annu Rev Biophys 37: 445-464.

57. Laing C., Schlick T. (2009) Analysis of four-way junctions in RNA structures. J Mol Biol 390: 547-559.

58. Cohen A., Bocobza S., Veksler I., Gabdank I., Barash D., Aharoni A., et al. (2008) Computational identification of three-way junctions in folded RNAs: a case study in Arabidopsis. In Silico Biol 8: 105-120.

59. Xin Y., Laing C., Leontis N.B., Schlick T. (2008) Annotation of tertiary interactions in RNA structures reveals variations and correlations. RNA 14: 2465-2477.

60. Kim S.H., Sussman J.L., Suddath F.L., Quigley G.J., McPherson A., Wang A.H., et al. (1974) The general structure of transfer RNA molecules. Proc Natl Acad Sci U S A 71: 4970-4974.

61. Butcher S.E., Pyle A.M. (2011) The molecular interactions that stabilize RNA tertiary structure: RNA motifs, patterns, and networks. Acc Chem Res 44: 1302-1311.

62. Byron K., Laing C., Wen D., Wang J.T.L. (2013) A computational approach to finding RNA tertiary motifs in genomic sequences: a case study. Recent Pat DNA Gene Seq 7: 115-122.

63. Kim J., Walter A.E., Turner D.H. (1996) Thermodynamics of coaxially stacked helixes with GA and CC mismatches. Biochemistry 35: 13753-13761.

64. Aalberts D.P., Nandagopal N. (2010) A two-length-scale polymer theory for RNA loop free energies and helix stacking. RNA 16: 1350-1355.

65. Shapiro B.A., Zhang K. (1990) Comparing multiple RNA secondary structures using tree comparisons. Comput Appl Biosci 6: 309-318.

66. Laing C., Jung S., Iqbal A., Schlick T. (2009) Tertiary motifs revealed in analyses of higher-order RNA junctions. J Mol Biol 393: 67-82.

67. Berman H.M., Westbrook J., Feng Z., Gilliland G., Bhat T.N., Weissig H., et al. (2000) The Protein Data Bank. Nucleic Acids Res 28: 235-242.

68. Laing C., Wen D., Wang J.T.L., Schlick T. (2012) Predicting coaxial helical stacking in RNA junctions. Nucleic Acids Res 40: 487-498.

69. Wen D. (2012) Design and implementation of a cyberinfrastructure for RNA motif search, prediction and analysis: NJIT.

70. Yang H., Jossinet F., Leontis N., Chen L., Westbrook J., Berman H. (2003) Tools for the automatic identification and classification of RNA base pairs. Nucleic Acids Res 31.

71. Breiman L. (2001) Random Forests. Mach Learn 45: 5-32.

72. Ferre F., Ponty Y., Lorenz W.A., Clote P. (2007) DIAL: a web server for the pairwise alignment of two RNA three-dimensional structures using nucleotide, dihedral angle and base-pairing similarities. Nucleic Acids Res 35: W659-668.

73. Capriotti E., Marti-Renom M.A. (2009) SARA: a server for function annotation of RNA structures. Nucleic Acids Res 37: W260-265.

74. Chang Y.F., Huang Y.L., Lu C.L. (2008) SARSA: a web tool for structural alignment of RNA using a structural alphabet. Nucleic Acids Res 36: W19-24.

75. Wang C.W., Chen K.T., Lu C.L. (2010) iPARTS: an improved tool of pairwise alignment of RNA tertiary structures. Nucleic Acids Res 38: W340-347.

76. Hoksza D., Svozil D. (2012) Efficient RNA pairwise structure comparison by SETTER method. Bioinformatics 28: 1858-1864.

77. Sarver M., Zirbel C.L., Stombaugh J., Mokdad A., Leontis N.B. (2008) FR3D: finding local and composite recurrent structural motifs in RNA 3D structures. J Math Biol 56: 215-252.

78. Dror O., Nussinov R., Wolfson H. (2005) ARTS: alignment of RNA tertiary structures. Bioinformatics 21 Suppl 2: ii47-53.

79. Abraham M., Dror O., Nussinov R., Wolfson H.J. (2008) Analysis and classification of RNA tertiary structures. RNA 14: 2274-2289.

80. Rahrig R.R., Leontis N.B., Zirbel C.L. (2010) R3D Align: global pairwise alignment of RNA 3D structures using local superpositions. Bioinformatics 26: 2689-2697.

81. He G., Steppi A., Laborde J., Srivastava A., Zhao P., Zhang J. (2014) RASS: a web server for RNA alignment in the joint sequence-structure space. Nucleic Acids Res 42: W377-381.

82. Jiang T., Lin G., Ma B., Zhang K. (2002) A general edit distance between RNA structures. J Comput Biol 9: 371-388.

83. Hochsmann M., Voss B., Giegerich R. (2004) Pure multiple RNA secondary structure alignments: a progressive profile approach. IEEE/ACM Trans Comput Biol Bioinform 1: 53-62.

84. Chen S., Zhang K. (2014) An improved algorithm for tree edit distance with applications for RNA secondary structure comparison. J Comb Optim 27: 778-797.

85. Havgaard J.H., Torarinsson E., Gorodkin J. (2007) Fast pairwise structural RNA alignments by pruning of the dynamical programming matrix. PLoS Comput Biol 3: 1896-1908.

86. Mathews D.H., Turner D.H. (2002) Dynalign: an algorithm for finding the secondary structure common to two RNA sequences. J Mol Biol 317: 191-203.

87. Sato K., Kato Y., Akutsu T., Asai K., Sakakibara Y. (2012) DAFS: simultaneous aligning and folding of RNA sequences via dual decomposition. Bioinformatics 28: 3218-3224.

88. Hamada M., Sato K., Kiryu H., Mituyama T., Asai K. (2009) CentroidAlign: fast and accurate aligner for structured RNAs by maximizing expected sum-of-pairs score. Bioinformatics 25: 3236-3243.

89. Meyer I.M., Miklos I. (2007) SimulFold: simultaneously inferring RNA structures including pseudoknots, alignments, and trees using a Bayesian MCMC framework. PLoS Comput Biol 3: e149.

90. Tabei Y., Tsuda K., Kin T., Asai K. (2006) SCARNA: fast and accurate structural alignment of RNA sequences by matching fixed-length stem fragments. Bioinformatics 22: 1723-1729.

91. Walter F., Murchie A.I., Duckett D.R., Lilley D.M. (1998) Global structure of four-way RNA junctions studied using fluorescence resonance energy transfer. RNA 4: 719-728.

92. Hernandez-Verdun D., Roussel P., Thiry M., Sirri V., Lafontaine D.L. (2010) The nucleolus: structure/function relationship in RNA metabolism. Wiley Interdiscip Rev RNA 1: 415-431.

93. Bindewald E., Wendeler M., Legiewicz M., Bona M.K., Wang Y., Pritt M.J., et al. (2011) Correlating SHAPE signatures with three-dimensional RNA structures. RNA 17: 1688-1696.

94. Jiang T., Wang L., Zhang K. (1995) Alignment of trees - an alternative to tree edit. Theor Comput Sci 143: 137-148.

95. Wang L., Zhao J. (2003) Parametric alignment of ordered trees. Bioinformatics 19: 2237-2245.

96. Kim N., Laing C., Elmetwaly S., Jung S., Curuksu J., Schlick T. (2014) Graph-based sampling for approximating global helical topologies of RNA. Proc Natl Acad Sci U S A 111: 4079-4084.

97. Yang H., Jossinet F., Leontis N., Chen L., Westbrook J., Berman H., et al. (2003) Tools for the automatic identification and classification of RNA base pairs. Nucleic Acids Res 31: 3450-3460.

98. Shang L., Xu W., Ozer S., Gutell R.R. (2012) Structural constraints identified with covariation analysis in ribosomal RNA. PLoS One 7: e39383.

99. Klein R.J., Eddy S.R. (2003) RSEARCH: finding homologs of single structured RNA sequences. BMC Bioinformatics 4: 44.

100. Gardner P.P., Wilm A., Washietl S. (2005) A benchmark of multiple sequence alignment programs upon structural RNAs. Nucleic Acids Res 33: 2433-2439.

101. Burge S.W., Daub J., Eberhardt R., Tate J., Barquist L., Nawrocki E.P., et al. (2013) Rfam 11.0: 10 years of RNA families. Nucleic Acids Res 41: D226-232.

102. Andronescu M., Bereg V., Hoos H.H., Condon A. (2008) RNA STRAND: the RNA secondary structure and statistical analysis database. BMC Bioinformatics 9: 340.

103. Smit S., Rother K., Heringa J., Knight R. (2008) From knotted to nested RNA structures: a variety of computational methods for pseudoknot removal. RNA 14: 410-416.

104. Goody T.A., Lilley D.M., Norman D.G. (2004) The chirality of a four-way helical junction in RNA. J Am Chem Soc 126: 4126-4127.

105. Lafontaine D.A., Norman D.G., Lilley D.M. (2001) Structure, folding and activity of the VS ribozyme: importance of the 2-3-6 helical junction. EMBO J 20: 1415-1424.

106. Lescoute A., Westhof E. (2006) Topology of three-way junctions in folded RNAs. RNA 12: 83-93.

107. Zirbel C.L., Roll J., Sweeney B.A., Petrov A.I., Pirrung M., Leontis N.B. (2015) Identifying novel sequence variants of RNA 3D motifs. Nucleic Acids Res 43: 7504-7520.

108. Petrov A.I., Zirbel C.L., Leontis N.B. (2013) Automated classification of RNA 3D motifs and the RNA 3D Motif Atlas. RNA 19: 1327-1340.

109. Petrov A.I., Zirbel C.L., Leontis N.B. (2011) WebFR3D--a server for finding, aligning and analyzing recurrent RNA 3D motifs. Nucleic Acids Res 39: W50-55.

110. Zirbel C.L., Sponer J.E., Sponer J., Stombaugh J., Leontis N.B. (2009) Classification and energetics of the base-phosphate interactions in RNA. Nucleic Acids Res 37: 4898-4918.

111. Laing C., Schlick T. (2010) Computational approaches to 3D modeling of RNA. J Phys Condens Matter 22: 283101.

112. Kim N., Zahran M., Schlick T. (2015) Computational prediction of riboswitch tertiary structures including pseudoknots by RAGTOP: a hierarchical graph sampling approach. Methods Enzymol 553: 115-135.

113. McGinnis J.L., Dunkle J.A., Cate J.H., Weeks K.M. (2012) The mechanisms of RNA SHAPE chemistry. J Am Chem Soc 134: 6617-6624.

114. Yang S., Parisien M., Major F., Roux B. (2010) RNA structure determination using SAXS data. J Phys Chem B 114: 10039-10048.