MODELING AND SEARCHING FOR NCRNA SECONDARY STRUCTURE

by

YONG WU

(Under the Direction of Liming Cai)

ABSTRACT

The discovery of functional non-coding RNAs (ncRNAs) has led to an increasing interest in efficient algorithms related to ncRNA secondary structure prediction and search for new ncRNA in genomes. The hidden Markov model and covariance model have been introduced to perform such tasks, but their limitations of modeling and computational complexity have compromised their practical application. Therefore, a tree-decomposition-based graph approach has been proposed to efficiently conduct the structure-sequence alignment, which underlies our computational tool, RNATOPS. As an essential part, the modeling and searching for accurate component candidates in a structure become one of major issues in the search process. In this thesis, a simplified model and many heuristic techniques have been proposed and exploited to address the issue. Comparisons between RNATOPS and Infernal have been conducted on several types of ncRNAs, which show the better performance of RNATOPS.

INDEX WORDS: ncRNA, secondary structure, hidden Markov model, covariance model
MODELING AND SEARCHING FOR NCRNA SECONDARY STRUCTURE

by

YONG WU

B.E., Zhejiang University, China, 2001

M.E., Shanghai Jiao Tong University, China, 2004

A Thesis Submitted to the Graduate Faculty of The University of Georgia in Partial Fulfillment

of the Requirements for the Degree

MASTER OF SCIENCE

ATHENS, GEORGIA

2009
MODELING AND SEARCHING FOR NCRNA SECONDARY STRUCTURE

by

YONG WU

Major Professor: Liming Cai
Committee: Russell L. Malmberg
           John A. Miller

Electronic Version Approved:

Maureen Grasso
Dean of the Graduate School
The University of Georgia
May 2009
ACKNOWLEDGEMENTS

I would like to thank everyone who helped make this thesis possible, especially, Dr. Cai for his insight in this field and his efforts to make my research better and always in the right direction, Dr. Malmberg for teaching me knowledge of biology and providing so many test data, and Dr. Miller for giving helpful suggestions. And last, but far from least, I would like to thank all of my friends at the Computer Science department and the RNA informatics lab for discussing with them and learning a lot from them.
# TABLE OF CONTENTS

<table>
<thead>
<tr>
<th>ACKNOWLEDGEMENTS</th>
<th>iv</th>
</tr>
</thead>
<tbody>
<tr>
<td>LIST OF TABLES</td>
<td>vii</td>
</tr>
<tr>
<td>LIST OF FIGURES</td>
<td>viii</td>
</tr>
<tr>
<td><strong>CHAPTER</strong></td>
<td></td>
</tr>
<tr>
<td>1 INTRODUCTION</td>
<td>1</td>
</tr>
<tr>
<td>1.1 RNA SECONDARY STRUCTURE AND PREDICTION</td>
<td>1</td>
</tr>
<tr>
<td>1.2 ALGORITHMS FOR NCRNA PREDICTION</td>
<td>3</td>
</tr>
<tr>
<td>1.3 RNATOPS</td>
<td>4</td>
</tr>
<tr>
<td>1.4 OUTLINE OF THE THESIS</td>
<td>6</td>
</tr>
<tr>
<td>2 MODELING STEMS AND LOOPS</td>
<td>9</td>
</tr>
<tr>
<td>2.1 PSEUDOKNOTTED STRUCTURES OF RNA</td>
<td>9</td>
</tr>
<tr>
<td>2.2 TRAINING SEQUENCE ANALYSIS</td>
<td>10</td>
</tr>
<tr>
<td>2.3 PROFILE HIDDEN MARKOV MODEL FOR LOOPS</td>
<td>12</td>
</tr>
<tr>
<td>2.4 COVARIANCE MODEL FOR STEMS</td>
<td>14</td>
</tr>
<tr>
<td>3 SEARCHING FOR THE TOP CANDIDATES</td>
<td>17</td>
</tr>
<tr>
<td>3.1 AN HMM FILTER</td>
<td>17</td>
</tr>
<tr>
<td>3.2 VITERBI ALGORITHM</td>
<td>18</td>
</tr>
<tr>
<td>3.3 CYK ALGORITHM</td>
<td>21</td>
</tr>
<tr>
<td>3.4 APPLIED HEURISTIC TECHNIQUES</td>
<td>23</td>
</tr>
</tbody>
</table>
# LIST OF TABLES

Table 2.1: Seven states and production rules for CM .......................................................... 14
Table 2.2: Match states and production rules for RNATOPS .............................................. 15
Table 4.1: tmRNA search results and comparison between RNATOPS and Infernal .......... 32
Table 4.2: RNaseP RNA search results and comparison between RNATOPS and Infernal .... 33
Table 4.3: Telomerase RNA search results and comparison between RNATOPS and Infernal ... 33
Table 4.4: 16s rRNA search results and comparison between RNATOPS and Infernal ......... 34
Table 4.5: Structure alignment accuracy and comparison between RNATOPS and Infernal ...... 35
<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Figure 1.1</td>
<td>RNATOPS architecture</td>
<td>5</td>
</tr>
<tr>
<td>Figure 1.2</td>
<td>Pipeline for modeling and searching</td>
<td>6</td>
</tr>
<tr>
<td>Figure 2.1</td>
<td>A simple RNA secondary structure</td>
<td>9</td>
</tr>
<tr>
<td>Figure 2.2</td>
<td>Training sequence pasta format</td>
<td>10</td>
</tr>
<tr>
<td>Figure 2.3</td>
<td>Stem elements</td>
<td>11</td>
</tr>
<tr>
<td>Figure 2.4</td>
<td>HMM transition structure</td>
<td>13</td>
</tr>
<tr>
<td>Figure 2.5</td>
<td>SCFG transition structure</td>
<td>15</td>
</tr>
<tr>
<td>Figure 2.6</td>
<td>Transition structure of left insertion with bulges</td>
<td>16</td>
</tr>
<tr>
<td>Figure 2.7</td>
<td>Transition structure of right insertion with bulges</td>
<td>16</td>
</tr>
<tr>
<td>Figure 3.1</td>
<td>HMM filter scanning genome</td>
<td>18</td>
</tr>
<tr>
<td>Figure 3.2</td>
<td>Pseudocode for local Viterbi algorithm</td>
<td>20</td>
</tr>
<tr>
<td>Figure 3.3</td>
<td>Pseudocode for CYK algorithm</td>
<td>22</td>
</tr>
<tr>
<td>Figure 3.4</td>
<td>Restricted region for a stem</td>
<td>24</td>
</tr>
<tr>
<td>Figure 3.5</td>
<td>Restricted regions for each arm of a stem</td>
<td>24</td>
</tr>
<tr>
<td>Figure 3.6</td>
<td>Stem-level regions for non-scanning region</td>
<td>28</td>
</tr>
<tr>
<td>Figure 4.1</td>
<td>Bacterial tmRNAs structure</td>
<td>30</td>
</tr>
<tr>
<td>Figure 4.2</td>
<td>RNaseP (bacterial B) RNA structure</td>
<td>30</td>
</tr>
<tr>
<td>Figure 4.3</td>
<td>Yeast telomerase RNA structure</td>
<td>31</td>
</tr>
<tr>
<td>Figure 4.4</td>
<td>Bacterial 16s rRNA structure</td>
<td>31</td>
</tr>
</tbody>
</table>
CHAPTER 1
INTRODUCTION

1.1 RNA SECONDARY STRUCTURE AND PREDICTION

Non-coding RNAs (ncRNAs) are functional RNA molecules, which are transcribed but not translated into protein [1]. Even though the number of ncRNAs encoded within the human genome is unclear for now, recent biology studies have indicated that there may be thousands of types of ncRNAs [2]. It has been shown that ncRNAs are diverse and play important roles in many biological processes, such as in gene regulation, chromosome replication, and RNA modification [3].

RNA is generally a single-stranded molecule consisting of four nucleotides A, C, G, and U, and typically folds onto itself to form secondary structure by base-pairing interactions between these nucleotides [4]. These base pairs are generally the Watson-Crick base-pairs (C \cdot G and A \cdot U) and the wobble base-pair (G \cdot U) [5, 6]. In the course of evolution, RNA sequences can mutate in nucleotides while keeping the structures that are important to the functions. If some base pair is disrupted by the nucleotide in one position, the evolutionary mechanism may correct this with a compensatory mutation [7]. Generally it is believed that the secondary structure may be conserved across related species but the consecutive nucleotides in a sequence have changed significantly.

Research has shown that the functions of ncRNAs are determined by tertiary structures, which in turn depend on secondary structures [8]. To better understand functions of ncRNAs,
knowing their secondary structure is therefore becoming increasingly important. However, it is
time consuming and expensive to obtain RNA structural data with experimental methods, such as
NMR and crystallography [9, 10]. Computational methods to predict ncRNA secondary structure
are drawing more and more attention and expected to expedite the process of discovering novel
ncRNAs and functions.

The computational determination of RNA secondary structure becomes exciting and has
been extensively studied. However, it has been proven to be a challenging and difficult task and
different approaches have been proposed to improve the performance. The methods developed
can be mainly divided into two broad categories:

1) Single sequence structure prediction, which is to optimally determine the secondary
structure given a single nucleotide sequence. Most methods, such as Mfold and RNAfold [11, 12],
icorporate stacking energy into prediction and minimize the free energy [13, 14]. The total
free energy is estimated for each possible structure, then choosing the one with the lowest free
energy. Unfortunately the real structure is not necessarily the one with lowest free energy.
Moreover, their time and space requirements are prohibitive for the pseudoknotted structure.

2) Comparative analysis, which infers a conserved structure from a set of related RNA
sequences. Since the secondary structure of most functional RNA molecules is strongly
conserved in evolution, a consensus structure can be determined across these molecules. Then a
multiple sequence alignment problem is introduced to align structurally equivalent residues from
these sequences in the same column. Many methods have used multiple sequence alignment to
predict the structure of the unknown sequences. There are three major classes for the
comparative analysis-based approach. The first class is to predict consensus structure and
alignment simultaneously, like Sankoff’s algorithm [15]. Since the problem is intractable,
Foldalign[16] and Dynalign[17] are restricted versions of Sankoff’s methods to improve the efficiency. The second class of methods folds the sequences using single sequence structure prediction algorithms and then aligns the resulting structures among sequences. Their performance is largely affected by the single sequence structure prediction algorithms used [18]. The third class of methods builds up a structural profile based on the alignment of multiple annotated sequences and then detects the similar structure in unknown sequences. Rsearch and Infernal are two systems for pseudoknot-free ncRNA search on genomes or database [19, 20].

1.2 ALGORITHMS FOR NCRNA PREDICTION

In order to search for ncRNA secondary structures in genomes, a computational tool is needed to scan through the target and compute the similarity based on a set of known sequences within an ncRNA family. The underlying algorithm is required to perform a sequence-structure alignment and its efficiency and accuracy are critical to the success of the search.

Diverse computational methods have been developed with the aim of solving the sequence-structure alignment problem, but still without an efficient solution. Up to now, stochastic context-free grammars (SCFGs) have been widely used to describe the RNA secondary structure [21]. Given a group of sequences with the known consensus structure, the SCFG-based covariance model (CM) [22], introduced by Eddy and Durbin, is popular for the development of computational tools for predicting the ncRNA secondary structure and structural homolog search [19]. The optimal sequence-structure alignment can be performed with a dynamic programming algorithm in O(WN^3) time, where N is the size of the model and W is the length of the target sequence. Both RSEARCH and Infernal are two representative computational tools, which can perform structure predictions and searches. Even though these CM-based methods can achieve
high searching accuracy, due to the time complexity for sequence-structure alignment, the more complex RNA structures are, the longer time these methods will require.

Furthermore, the issue of high time complexity becomes even more computationally prominent when allowing for a certain type of substructures, namely pseudoknot, which occur in real life RNA structures and therefore are important to model. However, the structure with pseudoknots can not be modeled by CM. A number of creative approaches have been tried to model the crossing stems of pseudoknots [23, 24, 25]; however, the time and space complexities for aligning sequence with these models are $O(N^4)$ or $O(N^5)$, which limits practical applications. Meanwhile, some heuristic methods have been developed to deal with RNAs containing pseudoknots. For example, ERPIN [26] considers the stem loops contained in a secondary structure. The genome is then scanned to find the possible hit locations for each stem loop. A hit for the overall structure is reported when there exists a combination of hit locations for different stem loops that conform with the overall structure. However ERPIN does not allow gaps in the alignment and thus may have low sensitivity when the target is a remote homolog of the query structure model.

1.3 RNATOPS

A graph modeling method was proposed by Cai [27], that can profile the secondary structure of a RNA family including pseudoknots. The topology of an RNA secondary structure is specified with a mixed graph, with non-directed edges denoting stems and directed edges for loops.

Based on this graph model, we have developed a practical computational tool called RNATOPS [28], for efficient RNA structure searching including pseudoknots, by exploiting the small tree width in the graph model. The time complexity has been reduced into $O(k^{t+1}n)$, where
n is the number of stems, k is the number of candidates for each stem, and t is the tree width for a decomposed structure graph. Furthermore, to achieve higher efficiency, some heuristic techniques for the candidate-searching step have been applied to obtain small values of parameters k and t without compromising search accuracy. In particular, parameter k can be chosen relatively small (e.g., k = 10) to ensure both accuracy and efficiency of the search.

**Figure 1.1 RNATOPS architecture**

RNATOPS has been implemented with the following three parts (see Figure 1.1). Firstly, the filtering part automatically identifies a conserved region in training data [29], constructs a hidden Markov model (HMM) of that region, and uses the HMM as a fast filter to scan a long genome and locate potential positions, where a structure motif is likely to be present [30, 31]. Secondly, the modeling and searching part will use the annotated training data to build up a statistical model for each stem and loop contained in a consensus structure, obtains top K
candidates for each stem and loop through the corresponding sequence-model alignment algorithm. At last, the assembly part uses the tree decomposition-based graph model to compute the valid and optimal combinations among all components’ candidates in an efficient way. Hits with high score will be reported to indicate potential motifs.

1.4 OUTLINE OF THE THESIS

My research mainly focuses on the structural modeling and candidate searching part, an essential part of RNATOPS. The goal is to produce high-quality candidates (see Figure 1.2), as accurate candidates are essential for RNATOPS to detect structure motifs accurately.

According to the searching process, my thesis consists of the four parts:

(a) Analyzing the ncRNA training sequences that have already been aligned to a consensus structure. In order to improve prediction accuracy, such information as restricted regions, average length, and clustering for stem and loop are calculated from training sequences.

![Figure 1.2 Pipeline for modeling and searching](image)

**Figure 1.2 Pipeline for modeling and searching**
(b) Building statistical models for each stem and loop. Full probabilistic models are proposed to consider all statistically possible situations. Position-specific CMs and the profile HMMs are generated for stems and loops respectively from a given set of multiple sequences with the consensus structure. Every stem consisting of two base-paired regions is modeled one simplified CM without the bifurcation rules. One profile HMM is generated from every two neighboring base regions; this can be the unpaired region connecting two stems. The parameters of these probabilistic models are computed from the multiple structural alignments using the maximum likelihood method. In addition, background frequencies are also weighted and incorporated into the models to avoid overfitting training sequences.

(c) Locating possible positions with a HMM filter. To greatly improve search speed, a conserved region will be manually or automatically determined from training sequences and a profile HMM model will be constructed for this region. Then this HMM is used as a filter to quickly scan the target genome such that the regions unlikely to contain a motif are removed.

(d) Searching candidates for each stem model. Within a scanning window, the top k candidates for each stem are identified by a simplified dynamic programming algorithm of Cocke-Younger-Kasami (CYK) [32]. Loop regions between stem region candidates are calculated by Viterbi algorithm based on the corresponding HMM models [33, 34]. To handle some unstable stems and diverse loops, several strategies have been implemented, including restricted regions, training sequences clustering, candidates merging, and length penalties.

According to Figure 1.2, the content of thesis will be organized as follows:

Chapter 2 mainly focuses on how to model stems and loops. Before that, training sequence format and some extracted features will be discussed. Then the profile HMM and CM models will be introduced.
Chapter 3 describes and discusses candidates-searching algorithm – simplified CYK and different candidate finding strategies. It gives details about how to apply heuristics techniques to improve the performance.

Chapter 4 presents several tests on real data and compares RNATOPS with Infernal, another well-known tool for RNA secondary structure search.

Chapter 5 provides a summary of my current research work and discusses further work that could be interesting to pursue.
CHAPTER 2
MODELING STEMS AND LOOPS

2.1 PSEUDOKNOTTED STRUCTURES OF RNA

RNA is typically a single-stranded molecule consisting of four types of residues: adenine(A), cytosine(C), guanine(G) and uracil(U). Usually, a stem is formed with stable base pairs stacking together, which are the canonical pairs including A •U, U •A, G •C, C •G, U •G, and G •U; a loop is a single stranded subsequence; a pseudoknot is an RNA secondary structure containing at least two stem-loop structures in which half of one stem is intercalated between the two halves of another stem [4, 35]; a bulge is an unpaired region embedded in either arm of a stem. There are many other types of loops in RNA, like hairpin loop, interior loop and multi-branched loop. These stem-loop structures together compose a secondary structure of RNA (see Figure 2.1).

Figure 2.1 A simple RNA secondary structure
2.2 TRAINING SEQUENCE ANALYSIS

The pasta format (pairing with fasta) has been proposed to describe multiple aligned RNA sequences and their consensus structure. Figure 2.2 shows the pasta format of training sequences corresponding to the secondary structure in Figure 2.1.

> Pairs
AAAAAAA.............aaa...aaaa......DDDDDD...EEEE....ddddddd....eeee..
>Aqu.aeo
-GGGGGCGGA-AGG&UU--CGACGCGGACCG---UGUAGGAG--CAGGCCGG--UGGU--CCCCAA
>The.ear
AGGGGGCGGA--CGGGUU--CGACGCGGAGUG---CCUGGGAAGGG&GC&GGGCCC&CU--CUCGUC
>The.the
-GGGGGUGAA--ACGUCU--CGACGCGGUGCA--GGGGGCUG---CGC&CC&GGGGGGUCU--CUCGUG
>Del.rad
CGGGGGUGAC--CGGUUU--CGACAGGGG&AA--GGAUGUG--CGUGUCAG--GGCGUUGG--CUCGUA
>Por.gin
-GGGGGUGA--CCGCUU--UGACAG--CGUGUA--GGC&UGAAG--CAUCUACUGGGGGCUU--CACUAU
>Chl.tep
AGGGGAUGA--CGGCUAUCG&CAAGGA&UG--AG&UGU---ACAUCGGAAUGCAUGG&GCGCGCUU
>Chl.tra
-GGGGGUG&JA--AGGUUU--CG&CUUAGAAAAG--CGA&UGU--CAUGGC&GGGUUGGCUGGCUCCAA

Figure 2.2 Training sequence pasta format

Every line beginning with ‘>’ is a comment line for the following line of RNA sequence. In the first two lines of the pasta file, the ‘>pair’ indicates the next line is an annotation of the consensus structure; the rest of the lines in pasta represent all sequences aligned with the consensus structure containing gaps (‘-’) for deletion, where the line beginning with ‘>’ indicates the human-readable identification for the next line. For the structure annotation, the left arm of a stem is labeled as upper case letters and the right arm as corresponding lower case letters. The rest of the line is labeled with dots (‘.’) for unpaired nucleotides. In this simplified perspective, the structure can be considered as a collection of stems and loops, which are able to denote arbitrary RNA structures, including pseudoknots and bulges.
From the training sequences, some potential features are exploited to help improve the performance of our search algorithm. After parsing the pasta file, RNATOPS obtains many statistic features for each component except positions:

1) For a loop, the average length (AVG) and its standard deviation (SD) are calculated based on the loop region across all training sequences. This region is likely to have a diversity of length and alignment. Considering that it is not advisable to build a single model to represent many different individuals, an alternative model approach is proposed to group the loop region based on the similarity and build a specific model for each group. The performance of this approach primarily depends on the detection of similar loops.

2) For a stem, the average length and its standard deviation are also calculated for each element: left arm, right arm, lead offset, tail offset, and middle loop (see Figure 2.3).

![Figure 2.3 Stem elements](image)

Because of loop diversity, stem positions possibly demonstrate a diverse property for one subset of training sequences as compared with another subset. In other words, the lead offset and tail offset of a stem in some sequences will be much different from some others, which results in
a large standard deviation. So it is better for the training sequences to be partitioned into clusters, each with a small standard deviation for the positions of left arm and right arm of a stem. One or more small regions for arms can be calculated from these groups, where the stem motif may be identified in more accurate manner than single large region.

2.3 PROFILE HIDDEN MARKOV MODEL FOR LOOPS

A profile Hidden Markov Model (HMM) is a statistical model consisting of a series of nodes, each of which corresponds to match, insertion, or deletion states [36]. Two types of probabilities are associated with HMM. One is the transition probabilities for transitioning from one state to another. The other is the emission probabilities associated with match and insertion states, based on the probability of a given residue existing at that position in the alignment.

To construct a full probabilistic HMM model, pseudocounts are introduced to account for emissions and transitions which were not present in the alignment when calculating the emission probabilities and transition probabilities, respectively. Even though there is no deletion in a column, a pseudocount will be assigned for the deletion as same as residues of A, C, G, and U. To build up a model of full probabilities, the transitions between insertion and deletion states are also added, although these are usually impossible. A column with the number of nucleotides below the number of gaps will be treated as an insertion column, while the rest of columns will be modeled as match columns with deletion.

A typical transition structure is shown as Figure 2.4 (from Durbin’s Book Biological Sequence Analysis), where squares are for match states, diamonds for insert states, circles for delete states, and arrows for transitions. For the insertion state, it can transit into itself to repeatedly generate consecutive nucleotides, which belong to the same insertion state. There are also many states before and after the i\textsuperscript{th} state, which are not illustrated in the figure.
Our training data in pasta format can provide a set of alignments of independent sequences. We can estimate the parameters directly using the following formula [36], where the number of times of transition or emission is counted up:

\[
\begin{align*}
    a_{kl} &= \frac{A_{kl}}{\sum_{l'} A_{kl'}} \\
    e_k(a) &= \frac{E_k(a) + PC(a)}{\sum_{a'} E_k(a') + PC(a')}
\end{align*}
\]

Here, \( k \) and \( l \) are indices over states, \( a_{kl} \) is the transition probability, \( e_k \) is the emission probability, PC is the corresponding pseudocount, and \( A_{kl} \) and \( E_k \) are the corresponding frequencies. In the practical application, the log-odds scores of the probabilities are preferred.

A special loop model, called the NULL model, is also introduced in the case of no nucleotide between any neighboring pairing regions in the consensus structure. It merely serves as a placeholder in the RNA secondary structure to indicate that there is a loop without any nucleotides.
2.4 COVARIANCE MODEL FOR STEMS

A covariance model (CM) [22] is a complicated statistical model based on stochastic context free grammars, SCFG for short. Typically, a CM is used to transform an RNA secondary structure into a set of SCFG terminals, nonterminals and production rules, which construct a grammar tree-like graph. Each node represents a number of states, each for a certain type of alignment.

Table 2.1: Seven states and production rules for CM

<table>
<thead>
<tr>
<th>State Type</th>
<th>Description</th>
<th>Production</th>
<th>Δₗ</th>
<th>Δᵣ</th>
<th>Emission Probability</th>
<th>Transition Probability</th>
</tr>
</thead>
<tbody>
<tr>
<td>P</td>
<td>Pair emitting</td>
<td>P → aYb</td>
<td>1</td>
<td>1</td>
<td>eₗ(a,b)</td>
<td>tᵣ(Y)</td>
</tr>
<tr>
<td>L</td>
<td>Left emitting</td>
<td>L → aY</td>
<td>1</td>
<td>0</td>
<td>eₗ(a)</td>
<td>tᵣ(Y)</td>
</tr>
<tr>
<td>R</td>
<td>Right emitting</td>
<td>R → Ya</td>
<td>0</td>
<td>1</td>
<td>eₗ(a)</td>
<td>tᵣ(Y)</td>
</tr>
<tr>
<td>B</td>
<td>Bifurcation</td>
<td>B → S</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>D</td>
<td>Delete</td>
<td>D → Y</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>tᵣ(Y)</td>
</tr>
<tr>
<td>S</td>
<td>Start</td>
<td>S → Y</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>tᵣ(Y)</td>
</tr>
<tr>
<td>E</td>
<td>End</td>
<td>E → ε</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

According to the convention of CM [37], a CM is composed of seven different types of states with corresponding production rules as Table 2.1. In this table, Y is new state; a and b are emitting residues; Δₗ and Δᵣ: the number of residues emitted to the left and right of state v respectively.

A modified CM has been introduced and implemented with consideration of the tree decomposition based approach and the requirement of efficient genome search. Since RNATOPS takes apart each stem from a RNA secondary structure with tree decomposition, the simplified CM without bifurcation rules has been proposed to greatly enhance the searching speed without loss of accuracy. Insertion states are added into production rules. When the pairing columns have
the number of pairs containing gap(s) larger than the number of pairs without any gaps, these two columns will be considered as insertion columns, denoted by left and right insertion column, respectively. Bulges appearing in a stem are also treated as loops, for which a profile HMM model are constructed to be contained within the CM. There are special production rules for the connection between the CM and the bulges.

![Figure 2.5 SCFG transition structure](image)

**Figure 2.5 SCFG transition structure**

The transition structure is showed as Figure 2.5, where B is for the start state, E for the end state, M for the match state, L/R for left/right insertion states without bulges between any two consecutive match states, and arrows for transitions. For a match state, it represents a pair of columns and can be further categorized into four types of match in Table 2.2, where MN is state type of none emitting, which can be seen as deletion of two nucleotides simultaneously.

**Table 2.2 Match states and production rules for RNATOPS**

<table>
<thead>
<tr>
<th>State Type</th>
<th>Description</th>
<th>Production</th>
<th>( \Delta^L )</th>
<th>( \Delta^R )</th>
<th>Emission Probability</th>
<th>Transition Probability</th>
</tr>
</thead>
<tbody>
<tr>
<td>M</td>
<td>Pair emitting</td>
<td>M ( \rightarrow ) aYb</td>
<td>1</td>
<td>1</td>
<td>( e_c(a,b) )</td>
<td>( t_r(Y) )</td>
</tr>
<tr>
<td>ML</td>
<td>Left emitting</td>
<td>ML ( \rightarrow ) aY</td>
<td>1</td>
<td>0</td>
<td>( e_c(a) )</td>
<td>( t_r(Y) )</td>
</tr>
<tr>
<td>MR</td>
<td>Right emitting</td>
<td>MR ( \rightarrow ) Yb</td>
<td>0</td>
<td>1</td>
<td>( e_r(b) )</td>
<td>( t_r(Y) )</td>
</tr>
<tr>
<td>MN</td>
<td>None emitting</td>
<td>MN ( \rightarrow ) Y</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>
If there are bulges in the left side between consecutive match states, L will become LB (Left insertion with Bulges) as Figure 2.6. The same thing happens for right side, where R will become RB (Right insertion with Bulges) as Figure 2.7. In order to build a full probability model, it is also considered that the insertion columns can interweave with bulge regions, even though the situation is unlikely to occur in biology.

**Figure 2.6 Transition structure of left insertion with bulges**

**Figure 2.7 Transition structure of right insertion with bulges**
3.1 AN HMM FILTER

It will take a very long time to completely scan one genome, which contains millions of nucleotides, by moving a scanning window forward one by one residue. In the scanning process, a large portion of search time will be apparently wasted by the computation of aligning those segments to a secondary structure model, where it is unlikely to contain the desired pattern.

To reduce the search time, some filtering methods [30, 31] have been introduced and significantly speeded up the homolog search of RNA secondary structure. The reason is the filtering process can efficiently remove an amount of genome segments which cannot contain the target when the filter is simply a light-weight HMM model, which can quickly scan through whole genomes. They have been incorporated into many search tools, which greatly improve the computational efficiency of genome searches.

In RNATOPS, the filtering idea has also been implemented as the following: based on the conserved region selected from training sequences [29], the constructed HMM model is used to scan a whole genome and then report those hits with score above the threshold (e.g., 0), where it is more likely to contain the similar secondary structure. For each of the reported filtering hits, the appropriate region can be extended according to the relative position of the conserved region within the whole consensus structure in training sequences. Then the segment within the extended region will be aligned with the structure model. Figure 3.1 shows such a process. If
more false positives can be identified correctly by the filter, then less computation will be performed in the whole structure alignment.

![HMM filter scanning genome](image)

**Figure 3.1 HMM filter scanning genome**

### 3.2 VITERBI ALGORITHM

Once the candidates for all stems have been identified, each segment between any neighboring pairing regions can be aligned to the corresponding loop model in order to calculate the overall score for the whole secondary structure of RNA. Here, the Viterbi algorithm is used to compute the maximum likelihood estimates of the successive states in the loop HMM given a sequence of RNA nucleotides.

Let $V^M_j(i)$ be the optimal score aligning sequence $x_{i-1}$ up to state $j$ with $x_i$ emitting by state $M_j$; Let $V^I_j(i)$ be the optimal score aligning sequence $x_{i-1}$ up to state $j$ with $x_i$ emitting by state $I_j$; Let $V^D_j(i)$ be the optimal score aligning sequence $x_{i-1}$ up to state $j$ ending with state $D_j$. The Viterbi recursion equations are shown as below.
Based on these recursion functions, a dynamic programming is implemented to avoid repeated computation and reduce the running time. There are two versions of the Viterbi algorithm applied in RNATOPS for different purposes: one is a global version for aligning the whole sequence to a model, the other is a local version for optimally aligning a part of sequence to a model (see Figure 3.2). In particular, the local Viterbi is applied in the HMM filter to scan genomes, while the global one is used to calculate the overall score for the whole structure alignment.

Since the consecutive bases in a subsequence can be optimally aligned to a model for the local version while the whole sequence is expected to be aligned to the model for the global version, the difference between these two versions will lead to different initialization and different traceback, in which the maximal probability is obtained from different regions.
Input:
A HMM H and RNA sequence s

Output:
The optimal alignment of a sequence to a model H.

Initialization:
n = the length of s,
m = the number of match states in H,
Vt[n, m, 3] = 3-D array of probabilities // 0: Match; 1: Insertion; 2: Deletion
Tb[n, m, 3] = 3-D array of previous state for traceback,

for i=1 TO n
for j=1 TO m
for v=1 TO 3
Vt[i, j, v] = 0
Tb[i, j, v] = None

Iteration:
for nt=1 TO n // the nucleotide in the sequence
for mid=1 TO m // the index of match states in model
for state=1 TO 3 // match or insertion or deletion state
for prestate=1 TO 3
// nt’, mid’, and state’ will differ according to previous state
// p(state’ → state) includes the transmission and/or emission probabilities
curprob= Vt[nt’, mid’, state’] + p(state’ → state)
update maxprob and maxstate
Vt[nt, mid, state] = maxprob
Tb[nt, mid, state] = maxstate

Traceback for optimal alignment:
// B: beginning nonterminal for G
for nt=1 TO n
for state=1 TO 3
if ( Vt[nt, m, state] > maxprob )
maxprob= Vt[nt, m, state]
maxtuple= (nt, m, state)
Ttraceback maxtuple

Figure 3.2 Pseudocode for local Viterbi algorithm
3.3 CYK ALGORITHM

The CYK algorithm [23] can be used to calculate the most probable parse tree of a sequence according to a covariance model. It operates on context-free grammars given in Chomsky normal form (CNF). The time complexity of CYK is $O(n^3)$, where $n$ is the length of the RNA sequence.

In order to obtain the actual parse trees of top K candidates, an additional data structure is provided to record the derivation path so as to make it possible to trace back and retrieve the optimal parse tree. The data structure will contain information about a production rule, a nonterminal, and a contained subsequence, all of which indicates how the current probability is calculated from which state using what production rule.

When the CYK alignment algorithm finishes, the top K cells with highest scores, which are specific to the beginning nonterminal, will be obtained in a 3-dimension table of probabilities. For each, it is possible to construct the probable parse tree by following the traceback links. Figure 3.3 is the pseudocode of CYK algorithm.

The memory usage of the algorithm corresponds to the size of the 3-D array of probability, $P$, and the 3-D array of the traceback data structure, $G$. If there are $M$ nonterminals in the SCFG, then the space of either $P$ or $G$ is $O(n^2M)$, so the total memory usage will be $O(n^2M)$. For the running time, the iteration part will take the largest portion in the CYK computation. Suppose the SCFG has $T$ production rules, the time complexity becomes $O(n^2T)$. And the traceback part will take constant time to retrieve the parse tree for the top K candidates, since it just visits a constant number of nodes for an alignment.
**Input:**
A simplified SCFG G and RNA sequence s

**Output:**
The top K candidates.

**Initialization:**
\[ n = \text{length of } s, \]
\[ V = \text{a set of nonterminals in } G, \]
\[ P[n,n,V] = 3\text{-D array of probabilities}, \]
\[ G[n,n,V] = 3\text{-D array of tuple (beginning position, span length, nonterminal, production rule )} \]
for \( i = 1 \text{ TO } n \)
for \( j = 1 \text{ TO } n \)
for nonterminal \( v \) in \( V \)
\[ P[i, j, v] = \text{invalid} \]
\[ G[i, j, v] = \text{invalid} \]

**Iteration:**

// wd: the width of subsequence
// ps: the beginning position of subsequence
// v: each production rule of G
for \( wd = \text{min width} \text{ TO } \text{max width} \)
for \( ps = \text{beginning position} \text{ TO } \text{ending position} - \text{wd} \)
for each production \( r \) in \( G \)

// \( ps', wd', v' \) will differ according to different production rules
\[ \text{newprob} = P[ps', wd', v'] + p(r) \]
if (newprob > \( P[ps, wd, v] \))
\[ P[ps, wd, v] = \text{newprob} \]
\[ G[ps, wd, v] = (ps', wd', v', r) \]

**Traceback for Top K candidates:**
//B: beginning nonterminal for G
for \( wd = \text{min width} \text{ TO } \text{max width} \)
for \( ps = \text{beginning position} \text{ TO } \text{ending position} - \text{wd} \)
if ( \( P[ps, wd, B] \) in top K probabilities )
push \( P[ps, wd, B] \) into stack
for each one in stack
traceback

---

**Figure 3.3 Pseudocode for CYK algorithm**

---

22
In the CYK algorithm, the allowed width of subsequence (wd) is calculated from statistic information in the training sequences as below:

\[ \text{minimal width} = \text{average middle loop length of a stem} - 3 \times \text{corresponding standard deviation} \]
\[ \text{maximal width} = \text{average span length of a stem} + 3 \times \text{corresponding standard deviation} \]

Since the allowed width (wd) is restricted between minimal middle loop and maximal stem span, there are many useless cells in P and G. Suppose \( w \) is the difference value between them, i.e.,

\[ w = \text{maximal width} - \text{minimal width}, \]

the memory can further reduced into \( O(nwM) \). For a stem with a long middle loop, the \( w \) will be much smaller than \( n \). Also the same thing happens for the running time \( O(nwT) \). Therefore, in the practical stem searches, the running time and memory space will be much less than indicated by our theoretical analysis.

Also, a modified CYK algorithm, namely two-region CYK, is implemented as a result of applying the restricted region technique. The two arms of a stem can only be allowed to generate from two restricted regions, which will be reflected by the restriction of “ps” in the algorithm shown in Figure 3.3.

### 3.4 APPLIED HEURISTIC TECHNIQUES

Even though the simplified SCFG-CYK produces the expected results, how to get better stem candidates as well as reducing the searching time is still of major interest. Without accurate stem candidates, RNATOPS will be unable to make a good prediction of RNA secondary
structure. After testing and analyzing a lot of experiment data, several strategies have been exploited and implemented, such as region restriction, length penalty, and candidate merging.

### 3.4.1 REGION RESTRICTION

With the consideration of a whole secondary structure present in a scanning window, it is not necessary to search the whole window for each stem. Because each stem will not appear within a certain offset from both ends, we exclude the head offset and tail offset regions (see Figure 3.4) for the corresponding stem to reduce the search time, as well as enhancing accuracy.

![Figure 3.4 Restricted region for a stem](image)

After applying this technique to both arms of a stem, the search region for each arm becomes more restrained. The left arm can only be produced within the left arm region and the right arm within the right arm region. The smaller the regions, the less time the search program takes, and more accurate the identified candidates will be.

![Figure 3.5 Restricted regions for each arm of a stem](image)
The restricted regions are calculated according to the statistical distribution of the consensus stem in the training sequences. Especially, a Gaussian distribution is taken for the position of the consensus stem in the RNA structure. The region for the correct motif of a stem is within a certain number of standard deviation units of the average position. Currently, we use 3 standard deviation units to assure the region contains expected candidate with high confidence.

3.4.2 LINEAR COMPUTATION

The computation of the sequence-structure alignment in HMM or SCFG costs a large part of time in the candidate-searching process. Considering that the scanning window will be moved one or more nucleotides forward each time, the data from the previous scanning window frame can be reused. In terms of the dynamic programming table, the current scanning window will keep most of the values in the table except the cells related to residues out of the window scope.

A reindexing method is adopted to keep the same memory blocks for each new scanning window; only new residues-related cells are computed. For the HMM filter, the linear computation has been implemented; for CM, there is no need to apply this technique since two consecutive window frames are unlikely to share nucleotides in common as a result of filtering.

3.4.3 CANDIDATE MERGING

In many search results, some of the candidates for some stem are very similar to each other and the difference is some nucleotides at the beginning or ending of either arm of the stem. Their pair regions are heavily overlapping in their positions, and most of them may have decent alignment scores with respect to a stem model. It suffices to record only one representative for these candidates. Based on these observations, a candidate merging strategy has been introduced to eliminate these similar candidates and select representatives to ensure a low value for K, the
number of top candidates. In particular, a small top K is anticipated to reduce running time caused by the dynamic programming combination.

If two candidates of a stem have a similarity of nucleotides of both arms above the threshold, they are treated as a same group. Eventually, for a group, a candidate will be chosen according to this strategy:

1. If there is a top candidate with its score much higher than others, it will be selected as a representative.
2. If there are several top candidates with their scores close to each other, the one with the shortest length of stem will be selected as a representative.

3.4.4 BACKGROUND MATRIX

The parameters of these stochastic models (HMM and CM) are computed from the multiple structural alignment using the maximum likelihood method. When a training set is small and biased, the information about pairs like pair frequencies can not be obtained fairly and adding pseudocounts for base pairs does not help too much. Especially, the training set is not well-conserved and includes many noncanonical pairs so as to make it impossible to achieve the high accuracy just based on the biased or small training set. To avoid over-fitting the training set, we take background frequencies into consideration. Here, the background probability matrix of pairs is computed from the whole family of RNA.

For a profile HMM, we allow pseudocounts for nucleotides in the match, insertion, and deletion states of the profile HMM. After many tests on real RNA data, 0.001 was chosen.

For a simplified CM, a 4×4 prior probability matrix $P_p$ for base pairs is introduced so that the probability of a base pair $P(x, y)$ is calculated as below:

$$P(x,y)=wP_1(x, y) + (1-w)P_p(x, y)$$
Here, $P_i$ is the base pair probability matrix obtained from the training data and $w$ is a weighting parameter.

### 3.4.5 LENGTH PENALTY

Each stem is an integral component of a secondary structure of RNA and carries the information about its relative position within the whole structure. Through statistical distributions of various length parameters for a consensus stem, a mechanism of penalty can be established according to the difference between the actual value and the corresponding expected value. That is called length penalty for this purpose. For each stem candidate, the length penalty is calculated based on the stem length, the middle loop length between the two arms, and the head and tail offsets. Let $P(c, M)$ represent the penalty for the covariance model $M$, which is computed based on the following formula:

$$
P(c, M) = \log\left(\frac{1}{cK^2}\right)
$$

Here, $K = \left|l - \mu\right|/\sigma \geq 1$ for the length $l$ deviating from mean $\mu$ with a standard deviation $\sigma$ and $c$ is a given constant to adjust the strength of the penalty.

The score of every possible candidate $c$ for $M$ is recalculated according to the formula below:

$$
S(c, M) = wA(c, M) + (1-w)P(c, M)
$$

Here, $A(c, M)$ is the logodds score from the alignment of a sequence with a model, $P(c, M)$ is the penalty function for the deviations of all lengths list above from their means, and $w$ ($0 \leq w \leq 1$) is a weighting parameter.
3.4.6 NON-SCANNING REGION

For each hit of HMM filtering, a small region is expected to do the sequence-structure alignment for whole structure search. The tighter the region is; the less time the search will spend. To achieve the least number of scanning windows, the non-scanning region is proposed to do exactly one alignment between the CM model and the sequence for a HMM filtering hit. Since the conserved region for HMM filter is located in training sequences, the relative region of each stem can be calculated as compared with the position of the conserved region. Once a hit of HMM filtering is reported, the region of each stem can be determined based on the hit position and then the top K candidates can be identified within the stem’s region. In this way, the candidates of all stems can be calculated only once for a filtering hit; this technique reduces multiple scanning windows into exactly one and hence decreases the searching time. In essence, the region is determined at the stem level instead of the whole structure level (see Figure 3.6).

![Figure 3.6 Stem-level regions for non-scanning region](image)
CHAPTER 4
TESTING RESULTS AND ANALYSES

4.1 EXPERIMENT OVERVIEW

Several tests have been conducted to evaluate the RNATOPS efficiency and accuracy. We chose 4 types of RNAs with different sizes: bacterial tmRNA, bacterial RNaseP (type B) RNA, yeast telomerase RNA, and bacterial 16s rRNA. RNATOPS performance has been compared with Infernal, one of the well known computational tools for RNA secondary structure search. Infernal was installed to conduct the tests with same data set in the same computers as RNATOPS. Both Infernal and RNATOPS use multiple structural alignments for model training and use filters to speed up search.

4.2 RNA DATA COLLECTION

For these tests, a cross-validation approach has been applied: if a sequence is present in a genomic sequence, it will be removed from the set of training sequences. The rest of training sequences will be used as a training set for current search on that genome.

We collected data mainly from the seed alignment of Rfam database [38] and other resources. For each type of RNA, many sequences are available, but there are some structure variations with some stem-loops in some sequences and not in others. Some data-cleaning steps have been performed based on these data. First, we chose a subset of sequences of each type so that the structure of each sequence does not differ from each other. Second, we tried to assure the corresponding genomic data of each sequence in the set is available. Finally, we remove the
columns consisting entirely of gaps, since the sequences with nucleotides in these columns have been removed in the previous step.

Bacterial tmRNAs [39] contain 178 molecules with an average length of 364 nucleotides and have a complex structure containing 4 pseudoknots (see Figure 4.1). After performing data-cleaning work, 43 sequences have been chosen from the 178 molecules and their corresponding bacterial genome sequences are available.

![Figure 4.1: Bacterial tmRNAs structure](image)

Bacterial RNaseP (type B) RNAs [40] contain 31 sequences with an average length of 367 nucleotides and have one sophisticated pseudoknot, which crosses many stems (see Figure 4.2). A subset of 10 sequences has been kept as training data; the full genome sequence was available for 7 out of 10. The other 3 without genome sequences were also kept; otherwise, the training set is too small and probably too biased to train a model.

![Figure 4.2: RNaseP (bacterial B) RNA structure](image)
Yeast telomerase RNAs [41] contain six sequences with an average length of 834 nucleotides as training data, where a conserved pseudoknot within a large stem loop exists (see Figure 4.3). While the genome of *S. cerevisiae* has been completely sequenced, those of the other *Saccharomyces* species have different levels of completeness and assembly. Out of these six *Saccharomyces* species telomerase RNAs, we finally collected four *Saccharomyces* genomes to conduct the searches.

![Figure 4.3: Yeast telomerase RNA structure](image)

The bacterial 16s rRNA were collected from the ribosomal RNA database [42]. The dataset contains only 12 sequences with a conserved structure of 1570 bps and 62 stems (see Figure 4.4), for which there exists an exact match between these sequences and their corresponding genomes.

![Figure 4.4: Bacterial 16s rRNA structure](image)
4.3 RESULT COMPARISONS

Bacterial tmRNAs: For RNATOPS, different values of parameter K (10/15/25), the number of top candidates selected for each stem, were tested and compared with Infernal. For every sequence in the training set, its corresponding genomic sequence was used as a query sequence to conduct tests. Since a leave-one-out cross validation approach was taken, there were 42 sequences in total as the training set for every round test. From the result table (see Table 4.1), we can clearly find that sensitivity becomes higher with larger K, but the search process takes longer time. For example, at K =10, the bacterial genome searches gained 88% sensitivity and 100% specificity; at K =25, the sensitivity increased to 98%, but the time increased to be 5 times more. However, Infernal achieved 100% sensitivity and specificity for these searches with comparable searching times.

Table 4.1 tmRNA search results and comparison between RNATOPS and Infernal

<table>
<thead>
<tr>
<th></th>
<th>RNATOPS</th>
<th></th>
<th>Infernal</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>K=10</td>
<td>K=15</td>
<td>K=25</td>
</tr>
<tr>
<td>Number of Training Sequences</td>
<td>42</td>
<td>42</td>
<td>42</td>
</tr>
<tr>
<td>Filter Used</td>
<td>HMM</td>
<td>HMM</td>
<td>HMM</td>
</tr>
<tr>
<td>Length of Filter</td>
<td>28</td>
<td>N/A</td>
<td>N/A</td>
</tr>
<tr>
<td>Number of Genomes Searched</td>
<td>43</td>
<td>43</td>
<td>43</td>
</tr>
<tr>
<td>Max/Min/Avg Genome Length(Mbps)</td>
<td>6.9/0.6/3.4</td>
<td>6.9/0.6/3.4</td>
<td></td>
</tr>
<tr>
<td>Max/Min of Searching Time (Minutes)</td>
<td>9.3/0.04</td>
<td>13.1/0.12</td>
<td>39.2/0.3</td>
</tr>
<tr>
<td>Avg/Std of Time per Genome(Minutes)</td>
<td>7.9/1.2</td>
<td>11.8/1.6</td>
<td>35.6/4.1</td>
</tr>
<tr>
<td>Sensitivity</td>
<td>38/43</td>
<td>40/43</td>
<td>42/43</td>
</tr>
<tr>
<td>Specificity</td>
<td>100%</td>
<td>100%</td>
<td>100%</td>
</tr>
</tbody>
</table>

Bacterial RNaseP RNAs: The number of top candidates for each stem was set to be 10 for RNATOPS, since their structures are more conserved. As compared with tmRNAs, the bacterial RNaseP RNA has similar length, but a more complex pseudoknot structure. Although both RNATOPS and Infernal had 100% sensitivity and 100% specificity in the 7 genomes tests,
RNATOPS spent much less time in searches than Infernal (see Table 4.2). The comparison result indicates Infernal took eight fold more time for searching the genomes of similar length. So RNATOPS shows a strong ability to handle the complex structure with pseudoknots while keeping the high efficiency.

Table 4.2 RNaseP RNA search results and comparison between RNATOPS and Infernal

<table>
<thead>
<tr>
<th></th>
<th>RNATOPS (K=10)</th>
<th>Infernal</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of Training Seqs</td>
<td>9</td>
<td>9</td>
</tr>
<tr>
<td>Filter Used</td>
<td>HMM</td>
<td>HMM</td>
</tr>
<tr>
<td>Filter Size</td>
<td>36</td>
<td>N/A</td>
</tr>
<tr>
<td>Number of Genomes Searched</td>
<td>7</td>
<td>7</td>
</tr>
<tr>
<td>Max/Min/Avg Genome Length (M bp)</td>
<td>5.1/1.8/3.1</td>
<td></td>
</tr>
<tr>
<td>Max/Min Time Used (Minutes)</td>
<td>18.7/9.5</td>
<td>150.4/58.5</td>
</tr>
<tr>
<td>Avg/Std Time Used per Genome (Minutes)</td>
<td>14.7/3.7</td>
<td>98/27.4</td>
</tr>
<tr>
<td>Sensitivity</td>
<td>100%</td>
<td>100%</td>
</tr>
<tr>
<td>Specificity</td>
<td>100%</td>
<td>100%</td>
</tr>
</tbody>
</table>

Table 4.3 Telomerase RNA search results and comparison between RNATOPS and Infernal

<table>
<thead>
<tr>
<th></th>
<th>S. bayanus</th>
<th>S. cerevisae</th>
<th>S. kudriavzevii</th>
<th>S. mikatae</th>
</tr>
</thead>
<tbody>
<tr>
<td># of training sequences</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td>Filter Used</td>
<td>HMM</td>
<td>HMM</td>
<td>HMM</td>
<td>HMM</td>
</tr>
<tr>
<td>Filter Size</td>
<td>47</td>
<td>N/A</td>
<td>47</td>
<td>N/A</td>
</tr>
<tr>
<td>Genome Length (M bps)</td>
<td>9.96</td>
<td>11.9</td>
<td>10.4</td>
<td>10.5</td>
</tr>
<tr>
<td>Time Used (Minutes)</td>
<td>5.7</td>
<td>295.2</td>
<td>5.5</td>
<td>654.9</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>6.4</td>
<td>372.1</td>
</tr>
<tr>
<td>Sensitivity</td>
<td>100%</td>
<td>100%</td>
<td>100%</td>
<td>100%</td>
</tr>
<tr>
<td>Specificity</td>
<td>100%</td>
<td>100%</td>
<td>100%</td>
<td>100%</td>
</tr>
</tbody>
</table>

RP: RNATOPS, INF: Infernal

Saccharomyces telomerase RNAs: As with previous RNAs, K is set to be 10 in RNATOPS. Both searching tools found the 4 Saccharomyces fungal telomerase RNAs perfectly in their
genomes. However, RNATOPS took from 5.5 to 6.4 minutes, while Infernal took from 295 to 654 minutes for the same searches (see Table 4.3). Once again, RNATOPS shows its advantages in searching for complex and large secondary structures with pseudoknots.

| Table 4.4: 16s rRNA search results and comparison between RNATOPS and Infernal |
|-------------------------------------------------|-------------|-------------|
| Number of Training Sequences                    | RNATOPS     | Infernal    |
| Filter Used                                     | HMM         | HMM         |
| Filter Size                                     | 111         | N/A         |
| Number of Genomes Searched                      | 11          | 11          |
| Max/Min/Avg Genome Length (M bp)                | 5.1/2.6/4.0 | 5.1/2.6/4.0 |
| Avg/Std Time Used per Genome (Minutes)          | 14.1/2.4    | 88/18.42    |
| Sensitivity                                     | 100%        | 100%        |
| Specificity                                     | 100%        | 100%        |

Bacterial 16s rRNAs: This set of RNAs had the longest sequences in our test with lengths of around 1500 bps. As expected, the results were similar to the telomerase and RNAseP RNAs: both RNATOPS and Infernal found the target with perfect specificity and sensitivity while RNATOPS only took one sixth time of Infernal searches in average.

After comparing the performance between RNATOPS and Infernal, we also investigated the quality of predicted secondary structure (see Table 4.5).

Firstly, we were interested in those sequences missed by RNATOPS at the low K value, especially for bacterial tmRNAs. RNATOPS missed 3 such sequences at K of 10. After outputting those stems, which caused the failure of whole structure prediction, we found that one or more stems in the predicted structure significantly deviated from the expected position in the consensus structure. These stems in these training sequences consisted of many rare, non-canonical base pairs, which were placed by the multiple alignment process. Furthermore, the
training data did not have some base pairs in specific pairing columns since the test approach had left out the only one sequence containing the base pair.

**Table 4.5 Structure alignment accuracy and comparison between RNATOPS and Infernal**

<table>
<thead>
<tr>
<th></th>
<th>tmRNA</th>
<th>RNaseP B RNA</th>
<th>Telomerase RNA</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>RP</td>
<td>INF</td>
<td>RP</td>
</tr>
<tr>
<td>Number of Structures</td>
<td>40/43</td>
<td>43/43</td>
<td>7/7</td>
</tr>
<tr>
<td>Correctly Found</td>
<td></td>
<td></td>
<td>7/7</td>
</tr>
<tr>
<td>Number of Found Structures with Stems off Position</td>
<td>4/43</td>
<td>7/43</td>
<td>2/7</td>
</tr>
<tr>
<td>Total Number of Stems off Position</td>
<td>9</td>
<td>17</td>
<td>4</td>
</tr>
<tr>
<td>Pseudoknot Regions Mistakenly Aligned</td>
<td>0</td>
<td>5</td>
<td>0</td>
</tr>
</tbody>
</table>

RP: RNATOPS, INF: Infernal

In addition to the failed cases, some potential problems are likely to occur in the successfully predicted sequences, considering the predicted structure alignment may be different from the given one. It is necessary for us to take a look at the predicted structure at the stem level. For the predicted stems deviated from expected positions, RNATOPS identified totally 9 such stems while Infernal had 17. One of the possible reasons is that Infernal only deals with pseudoknot-free structure so that region for a pseudoknot may be unexpectedly aligned to other stems. In this set of search tests, there were totally 5 such mistakes found in the search results of Infernal while it is not problem for RNATOPS. From this point of view, RNATOPS has higher quality of structure prediction than Infernal, even though Infernal found all tmRNAs sequences. For RNaseP B RNA, RNATOPS identified 2 structures whose alignments put totally 4 stems off their correct positions by more than a few nucleotides.
CHAPTER 5
CONCLUSION AND FUTURE WORK

5.1 CONCLUSION

My research focuses on how to build up stem/loop models and efficiently search a genomic sequence have been exploited and implemented as a requisite part of the RNATOPS project.

In the modeling stage, background pair frequencies were considered for stems and pseudocounts for loops to avoid over-fitting the training set. To speed up RNATOPS, a simplified SCFG without bifurcation rules has been proposed to generate a covariance model according to the essence of tree decomposition in our RNATOPS. To represent different unconserved loop regions, an alternative HMM model for a loop was also introduced and implemented to handle the variation in the loop region. If the arm positions of a stem in one subset of training sequences differ greatly from another subset, the idea of training sequence clustering has been implemented to divide these sequences into different groups according to the standard deviation of arm positions.

In the searching stage, several heuristic techniques have been proposed and implemented in this thesis. With the aim to develop a fast and accurate RNA pseudoknot search program, it is important to find accurate candidates for each stem quickly. In the HMM filter, an adapted Viterbi algorithm of local alignment has been implemented with linear computation to achieve fast genome scanning. A two arm-region based CYK algorithm has also been developed to get accurate candidates as well as other strategies, such as merging candidates, length penalty, and
restricted region. Higher speed has been achieved by a non-scanning technique for whole structure search.

After comparing the RNATOPS performance with Infernal, we can see that RNATOPS has many advantages over Infernal. It can handle pseudoknotted structures without compromising computation time no matter how complex the secondary structure is. In particular, RNATOPS has much faster speed than Infernal for the large structure. Furthermore, since RNATOPS can handle structures with pseudoknots, it is more likely to produce high quality prediction.

5.2 FUTURE WORK

However, there are still a lot of remaining issues occurring in RNATOPS. When the number of candidates for each stem is allowed to be small, it will become difficult for RNATOPS to include expected candidate within it and keep high sensitivity. There is a risk of missing real hits. This issue is related to how to find out the more accurate stem candidates. One of the possible ways is to mine more potential features from the training set. Currently we just consider the statistical data for position and length for an individual stem or loop, which is isolated from each other. There should be some information about relationship among these stems and loops, which can help improve the prediction accuracy.

Sometimes, the training set does not provide good data for training the model of some stem or loop, like tmRNAs in our test. These components often make the whole structure prediction worse or even failed. If such stems can be ignored selectively, the structure prediction will be much better. Therefore, the observation requires that our program can determine which stems are unstable and which loop regions are too diverse, and then allow them to disappear in the structure prediction intelligently.
In addition to taking the background pair frequencies into consideration, how to utilize the training set is also worth having further exploration. From time to time, the training set may be biased or can not represent the large-scale sequences of RNA. In these cases, some strategies should be taken to adjust the parameters of models.

Some issues are raised in the tests of bacteria tmRNAs, since the loop region varies a lot from each other sometimes. We observed that the some loops often contribute negative scores to the overall structure alignment. It is unreasonable to construct a single model to represent all of them. In addition to the alternative models, some ways are expected to preprocess the loop region variation in future.
REFERENCES


