CONSTRAINED SECONDARY STRUCTURE PREDICTION USING STEM DETECTION

A Thesis

by

VENKATA VIKAS VARMA NALLAPARAJU

Submitted to the Office of Graduate and Professional Studies of
Texas A&M University
in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE

Chair of Committee,	Byung-Jun Yoon
Committee Members,	Xiaoning Qian
	Yang Shen
	Ruihong Huang
Head of Department,	Miroslav M. Begovic

December 2018

Major Subject: Electrical Engineering

ABSTRACT

RNA sequence analysis and structure prediction are classical topics of computational biology and a powerful tool to examine complex genomic data. Over the decades, various tools have been developed to predict RNA secondary structures and sequence alignments, a majority of which utilize one of the two characteristic approaches: (a) thermodynamic minimum free energy or (b) probabilistic maximum likelihood prediction. However, despite numerous takes on modeling these approaches, the computational complexity of the developed algorithms hasn't seen significant improvements. Most algorithms still operate with a polynomial time complexity of $O(N^3)$. This cost is significantly large while processing large RNA sequences with hundreds of bases.

In this thesis, a constrained structure prediction algorithm is presented that aims to diminish the computational overhead of traditional RNA structure prediction methods to $O(N^2)$. The proposed algorithm employs pattern recognition methods to devise rules for constructing a confined space of possible secondary structures. This confined structure space is then searched to find a secondary structure that satisfies the optimality criterion. Through this document, we present the design details of the proposed algorithm implemented using the minimum free energy (MFE) model. Later, we compare its performance to Zuker's algorithm which is the conventional dynamic programming equivalent of the MFE model. The proposed algorithm provides a significant reduction in CPU time to process longer sequences which can be attributed to its lower computational complexity.

CONTRIBUTORS AND FUNDING SOURCES

This work was supervised by a thesis committee consisting of Professor Byung-Jun Yoon [Faculty Advisor], Professor Xiaoning Qian of the Department of Electrical and Computer Engineering, Professor Yang Shen of the Department of Electrical and Computer Engineering and Professor Ruihong Huang of the Department of Computer Science and Engineering.

All work for the thesis was completed by the student, in collaboration with Professor Byung-Jun Yoon of the Department of Electrical and Computer Engineering.

There are no outside funding contributions to acknowledge related to the research and compilation of this document.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1

INTRODUCTION

Studies have ascertained that nucleic acids (DNA & RNA) are the central stimulant of the flow of genetic information within a biological system [10]. This developed a lot of interest in nucleic acid research among biologists and computer scientists leading to the assembly of an abundance of genetic information to process.

In this document, we explore a sub-problem of nucleic acid analysis – RNA secondary structure prediction. RNA is a polymer composed of four nucleotides: adenine, cytosine, guanine, and uracil, abbreviated as A, C, G, and U respectively. RNA molecules often fold into complex secondary and tertiary structures [11] due to the formation of hydrogen bonds between these nucleotides called complementary base-pairs. Two classes of widely modeled base-pairs are: canonical Watson-Creek A-U and G-C pairs [11], Wobble pairs G-U pairs [11].

The Secondary structure of an RNA molecule can be determined through x-ray crystallography, which is expensive and time-consuming. Moreover, many RNA molecules cannot be crystallized. These limitations paved the need for mathematical models that perform efficient RNA secondary structure prediction based solely on the knowledge of its primary sequence. The computational methods developed so far use different approaches, mainly: energy minimization methods [2], probabilistic models [1] which search the structure space for optimal secondary structures and comparative sequence analysis methods. These widely used algorithms, given a sequence of length N, predict the structure in $O(N^3)$ time. Since their introduction, extensions to these methods have been proposed to curb their polynomial time complexity [12]. However, while some enhancements

come at the expense of accuracy or a worst-case time-complexity of $O(N^3)$, some information theory based algorithms [13] require additional data for the prediction task.

In this document, we propose a constrained structure prediction algorithm developed within the principles of existing structure prediction algorithms that instead locates optimal substructure-motifs from the RNA sequence and iteratively stitches them to form the final secondary structure. This approach restricts the search space of the prediction algorithm to the one spanned by the motifs, thereby improving the time complexity involved with the structure prediction to $O(N^2)$.

The following chapters will detail the concepts and terminology involved with RNA secondary structures, dynamic programming [1][2] algorithms for structure prediction, stem detection methodology, and the proposed constrained prediction algorithm developed using the thermodynamic principles of the Zuker algorithm. Finally, we provide a comparative performance analysis of our constrained prediction model against Zuker's conventional structure prediction algorithm.

CHAPTER 2

BACKGROUND

2.1 RIBONUCLEIC ACIDS & STRUCTURES

Ribonucleic Acids (RNA) is a linear macromolecule made up of nucleotide bases: adenine (A), cytosine (C), guanine (G), and uracil (U). RNA is one of the three major polymers that facilitates encoding and decoding of genetic information in all life forms. Unlike the other major polymer, DNA which is a double-stranded paired molecule, RNA occurs as a single-strand that folds onto itself due to interactions between its nucleotides. These interactions occur at different factions leading to complex multi-dimensional structure of the RNA.

1. *Primary Structure*: The primary structure is simply the complete sequence of nucleotide bases arranged in 5'-3' order.

2. *Secondary Structure*: Secondary structure of an RNA is the set of all interactions between the bases in the primary structure located in different sub-strands of the molecule.

3. *Tertiary Structure*: RNA's tertiary structure is a dimension higher than its secondary structure. It is a representation of the arrangement of the atoms of the molecule in a three-dimensional space.

The primary sequence of an RNA determines the covalent structure of RNA and is the basis for determining how the RNA folds into its secondary and tertiary structures [11]. Figure 1 illustrates the primary, secondary and tertiary structures of a *t-RNA* molecule. Primary structure is illustrated

in Figure 1(a) as a linear strand of nucleotides arranged in 5'-3' direction. Interactions between nucleotides forming secondary and tertiary structures are subsequently illustrated in Figure 1(b) and 1(c).

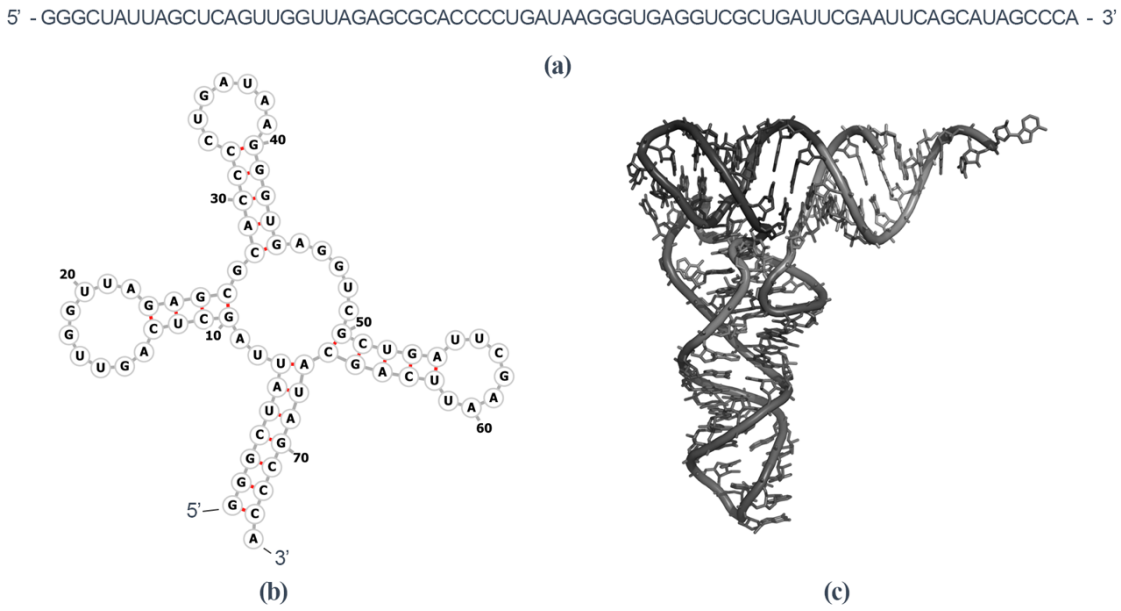


Figure 1: *(a) Primary, (b) Secondary and (c) Tertiary structures of RNA molecules.*

Base-pairs refer to the set of hydrogen-linked nucleotides connecting the complementary strands of a molecule of DNA or RNA and consisting of a purine linked to a pyrimidine by hydrogen bonds. Certain nucleotides in the RNA sequence associate to form base-pairs. In DNA, base pairing occurs between the two parallel strands to produce the helix. However, due to the single stranded structure of RNA, it loops onto itself forming intramolecular base-pairs among its sub-strands. Different types of base-pairs occur in RNA including the *canonical* Watson-Crick pairs (A-U & G-C) often observed in DNA [14]. However, unlike DNA, RNAs also have the ability to form *non-canonical base-pairs*, like the Wobble pairs: G-U.

2.2 RNA SECONDARY STRUCTURE PREDICTION

RNA secondary structure is a schematic representation of its base-pairs. Various computational biologists have addressed this problem with different models. Nussinov *et al.* [4], proposed a dynamic programming algorithm which maximized the number of base-pairs in the secondary structure. In 1981, Zuker *et al.* [2], proposed a refined dynamic programming approach that models the nearest neighbor energy interactions. This algorithm took a minimum free-energy approach that identifies the secondary structure with the least free-energy. However, the drawback of such an approach was its profound sensitivity towards the utilized thermodynamic parameters [15].

Subsequently, Stochastic Context-Free Grammars (SCFGs) [3] based probabilistic models were introduced to overcome this limitation. Probabilistic prediction models find the most likely structure of an RNA using stochastic parameters extracted from pre-computed secondary structures. Despite the improved parametric independence, SCFG models were slow to beat the accuracy of their thermodynamic counterparts. Decades later, SCFG models have caught up to the accuracy and reliability of free-energy algorithms owing to the use of advanced learning models like Hidden Markov Models (HMMs) [1] and sophisticated grammars.

Recent advances have seen a surge in the use of machine learning models to solve the structure prediction problem significantly – statistical sampling [16], back-propagation neural networks and deep learning frameworks [17]. Although some of the recent RNA modeling techniques have worked towards improving the accuracy of the predictions, none have significantly addressed the polynomial $O(N^3)$ time complexity involved with this problem.

2.2.1 STRUCTURE MODEL

Structure prediction algorithms share a common set of norms and notations while modeling RNA secondary structures. Consider a sequence S with n characters belonging to the finite set $\{A, C, G, U\}$, each representing a nucleotide in the RNA primary structure. A *base-pair* between two nucleotides S_i and S_j is denoted by $S_i.S_j$ or simply $(i.j)$. The secondary structure is the set of all *base-pairs* $K = \{\, i.j \mid 1 \leq i < j \leq n\}$ that satisfy the following rules:

- A base can only be associated with one pair in the structure, i.e., if $(i.j) \in K$ and $(i'.j') \in K$, then $i = i'$ *iff* $j = j'$ and vice versa.
- If $(i.j) \in K$, then $j - i > d$ where d is an optional parameter that indicates the minimum hairpin loop size.
- All base-pairs in the secondary structure are nested i.e., if $(i.j)$ & $(i'.j') \in K$, then both pairs follow $i < i' < j' < j$, if $i < i'$, and $i' < i < j < j'$ otherwise. Secondary structures that violate this rule are termed *'pseudoknots'* and are known to occur often.

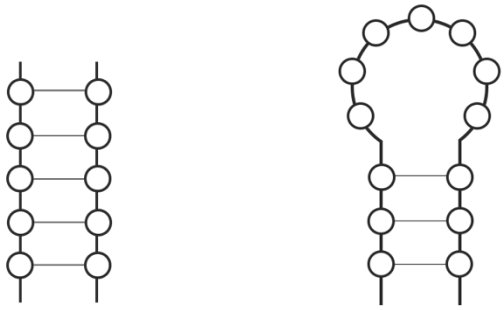
2.2.2 MOTIFS

Depending on how the *base-pairing* occurs, the secondary structure consists of various commonly recurring sub-structure motifs – hairpin-loops, stems, bulges, internal-loops, multi-loops and pseudoknots [1].

1. *Stem*

   The structure motif resulting from a contiguous set of multiple base-pairs that form a double-helix type structure is known as a stem, Figure 2(a). Stems are the key building

blocks of all secondary structures. Therefore, our constrained structure prediction algorithm introduced in the following chapters utilizes stems as the primary motif for modeling secondary structures.



(a) (b)

Figure 2*: Base-pairs in (a) stem structures and (b) hairpin-loops.*

2. *Hairpin-loop*

Stem structures that culminate in a sequence of unpaired bases that form a hairpin like loop are called hairpin-loops or simply H-loops, as illustrated in Figure 2(b). The formation of a hairpin-loop depends on the stability of the resulting stem and loop sub-structures.

3. *Bulge or Internal Loop*

Another class of commonly observed structure motif are internal loops which is an imperfection or an unpaired section of the stem. Therefore, I-loops appear between two sections of the stem and span a certain set of bases on the $5'$ and $3'$ ends of the sequence. I-loops that have the same number of unpaired bases on the $5'$ end (n_1) and $3'$ end (n_2), i.e., $n_1 = n_2$ are called symmetric I-loops as shown in Figure 3(b) and those that violate this property, $n_1 \neq n_2$ are called asymmetric I-loops.

Bulges are a special case of internal-loops where the loop occurs only on one end of the folded structure. For example, $n_1 > 0, n_2 = 0$ or $n_2 > 0, n_1 = 0$ as shown in Figure 3(a).

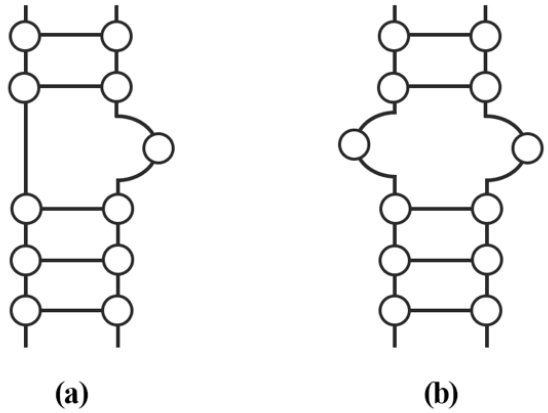


Figure 3*: Illustration of stem structures with (a) a bulge on the right strand and (b) a symmetric internal loop.*

4. *Multi-loop*

Multi-loop structures are the most complicated common structure motif. M-loop structures occur when bases in the hairpin-loop of a stem also close other hairpin-loops forming a nested series of stems. This nested secondary structure is called the M-loop, shown in Figure 4, and occurs often in structure prediction.

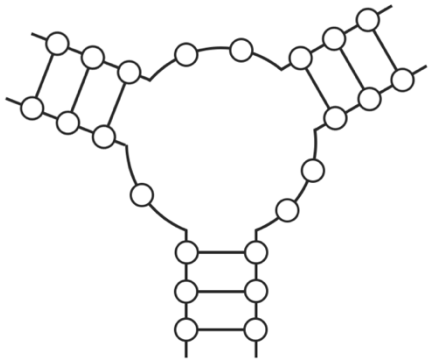


Figure 4*: A multi-loop structure with three stem branches.*

5. *Pseudoknot (PK)*

Pseudoknots are structures where there are interactions between bases from different stem or loops, as illustrated in Figure 5. They violate the third rule of secondary structure formation and accurate modeling of pseudoknots is quite difficult. Therefore, in this document, we restrict the study to the analysis of nested secondary structures, avoiding the possibility of a pseudoknot formation altogether.

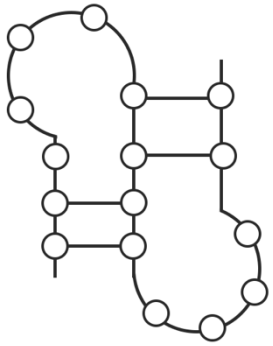


Figure 5*: Example of a pseudoknot between two hairpin-loop structures sharing a common strand.*

CHAPTER 3

DYNAMIC PROGRAMMING ALGORITHMS

Nussinov *et al.* [4] first introduced a rudimentary dynamic programming algorithm with an approach to maximize the base-pairs in the predicted secondary structure. This algorithm was weighed down by its simplicity, but its mechanism was carried forward or extended in the more sophisticated energy minimization or probabilistic SCFG based algorithms.

3.1 NUSSINOV ALGORITHM

Nussinov's algorithm calculates the optimal structure of subsequences recursively by finding optimal structures for smaller subsequences and working its way up to larger ones. Formally, given an RNA sequence x of length L, let $\delta(i,j)$ be an indicator function for complementary base-pairs x_i and x_j ($i < j$). The recursion populates a score matrix $\gamma(i,j)$ that records the maximum number of base pairs in the $x_i \ldots x_j$ subsequence.

Algorithm

1. Initialization $\gamma(i,j) = 0, \quad \forall\, i,j \ \in (1,L)$

2. Recursion: For all subsequence, starting with length 2 to L,

$$\gamma(i,j) = \ min \begin{cases} \gamma(i+1,j) \\ \gamma(i,j-1) \\ \gamma(i+1,j-1) + \ \delta(i,j) \\ \max\limits_{k \in (i,j)} \ [\,\gamma(i,k) + \ \gamma(k,j)\,] \end{cases}$$

3. Traceback

Initialize – $push(1, L)$ onto the stack

Recursion – while the stack is not empty

$pop(i, j)$

$if\ i \geq j, break;$

$else\ if\ \gamma(i+1, j) = \gamma(i, j),\ push\ (i+1, j)$

$else\ if\ \gamma(i, j-1) = \gamma(i, j),\ push\ (i, j-1)$

$else\ if\ \gamma(i+1, j-1) + \delta(i, j) = \gamma(i, j)$

$record\ (i, j)\ pair$

$push\ (i+1, j-1)$

$else\ for\ k = i - j,\ if\ \gamma(i, k) + \gamma(k, j) = \gamma(i, j)$

$push\ (i, k)$

$push\ (k, j)$

The drawbacks of the Nussinov algorithm are that the underlying principle assumption that the optimal secondary structure is the structure with the maximum possible base pairs is not always true and its uniform scoring scheme for all base-pairs. Many algorithms have thus addressed these drawbacks using different approaches.

3.2 ENERGY MINIMIZATION

Much of the secondary structure prediction's literate contains various revisions of the *free energy minimization* method, aka the *thermodynamic* approach. Zuker *et al.* [2] first introduced the dynamic programming approach to energy minimization. This design sought to find the secondary structure with the least total free energy by iteratively calculating free energies of substructures such as stems, bulges and loops.

Despite nearly three decades of research, these energy minimization algorithms still have a long journey to efficacy and reduced time complexity. The deficiency of such break-throughs in this realm can be attributed to its parametric limitations and the role of external agents in the RNA folding process that is neglected by the approach. However, many packages have been developed using such energy minimization algorithms – `Mfold` and `RNAFold` [2]. We discuss Zuker's folding algorithm here as our implementation of our constrained prediction algorithm retains most of the principles laid down by this algorithm.

3.2.1 FREE ENERGY MODEL OF RNA

Let x be an RNA sequence of length L and K be the set of base-pairs representing its secondary structure. A base $1 \leq i \leq L$ is unpaired in K if there is no complimentary base j, such that (i, j) & $(j, i) \notin K$. Based on the enclosing structure, closing base-pairs in K are subdivided further

1. The base pair (i, j) closes a hairpin-loop if the subsequence it encloses doesn't have a base-pair; $\forall\, i < k < j, k$ is unpaired in K.
2. Pair (i, j) closes a stem if it encloses its immediate base-pair i.e., $(i + 1, j - 1) \in K$.
3. The set of base pairs (i, j) & $(i', j') \in K$ close an internal loop iff $i < i' < j' < j$, (i, j) doesn't close a stacking loop and $i \ldots i', j' \ldots j$ are all unpaired bases.
4. A set of M base pairs comprise a m-multiloop if it contains $m\,(> 1)$ base-pairs $(i_1, j_1) \ldots (i_m \ldots j_m)$ that individually enclose different loops and a closing pair (i, j). All bases that occurs between the base-pairs and the closing pair should be unpaired i.e., $i + 1 \ldots i_1 - 1; j_1 + 1 \ldots i_2 - 1; \ldots . j_m + 1 \ldots j - 1$ are unpaired.

3.2.2 ZUKER FOLDING ALGORITHM

The Zuker folding algorithm adapts the energy minimization approach to search for the secondary structure with the lowest equilibrium free energy, ΔG. It is analogous to Nussinov's base-pair maximization approach, using dynamic programming recursion to compute minimum free energies of substructures and iteratively extending this to estimate minimum free energy of the entire sequence. The traceback algorithm then identifies base-pairs which contribute to the secondary structure with least free energy.

Consider an RNA sequence x of length L, four matrices W, V, VBI and VM are defined to hold computed minimal free energies of various types of substructures. These energies are calculated by means of experimentally determined free energy parameters $-eH, eS, eBI$ and eM, for hairpin-loops, stacks, internal-loops and multi-loops respectively [2].

The Zuker recursion [2] matrices capturing contributions from different loop structures are defined below $\forall 1 \leq i < j \leq L$.

- W holds the energy of the optimal substructure $(1, i)$.

$$W(i) = \min \begin{cases} W(i-1) \\ \min\limits_{1<j\leq i} [\, W(j-1) + V(j,i) \,] \end{cases}$$

- V identifies the free energy of the substructure s, spanned between (i, j), with the constraint that (i, j) is the closing base-pair for s.

$$V(i,j) = \min \begin{cases} eH(i,j) \\ eS(i,j) + V(i+1, j-1) \\ VBI(i,j) \\ VM(i,j) \end{cases}$$

Here, $eH(i,j)$ is the hairpin-loop energy for a closing pair (i,j). $eS(i,j)$ is the stacking energy of stacking (i,j) and $(i+1,j-1)$.

- VBI defines the energy of the internal loop or bulge closed by the (i,j) base-pair.

$$VBI(i,j) = \min_{\substack{i'-i+j-j'>2 \\ i<i'<j'<j}} \{ eBI(i,i',j,j') + V(i',j') \}$$

Where $eBI(i,j,i',j')$ is the energy of a bulge or an internal loop for the base pairs (i',j') closing a stacking stem structure s.

- VM hold the energy of the optimal multi-loop structure closed by the base pair (i,j) at $VM(i,j)$. Therefore, it can be computed using $-$

$$VM(i,j) = \min_{\substack{i<i_1<j_1<\cdots \\ \ldots<j_k<j}} \{ eM(i,j,i_1,i_1,\ldots,i_k,i_k) + \sum_{n=1}^{k} V(i_n,i_n) \}$$

Where eM is the energy of the multi-loop consisting of closing base-pairs $\{(i_1,j_1),(i_2,j_2),\ldots,(i_k,j_k)$ & $(i,j)\}$. Most implementations of the Zuker algorithm simplify the multi-loop recursion to use an approximation of the multi-loop energy as a linear function of the number of unpaired bases in the loop like below,

$$eM(i,j,i_1,i_1,\ldots,i_k,i_k) = a + bk + c(i_1 - i - 1 + j - j_k - 1 + \sum_{n=1}^{k-1} [i_{n+1} - j_n - 1])$$

Where a,b,c are constants. This enables the algorithm to introduce a multi-loop energy matrix WM for the subsequence (i,j) such that

$$WM(i,j) = \min \begin{cases} V(i,j) + b \\ WM(i,j-1) + c \\ WM(i+1,j) + c \\ \min_{i<k\leq j} WM(i,k-1) + WM(k,j) \end{cases}$$

The calculation of multi-loop energy can then be restated as

$$VM = \min_{i+1<k<j-1}\{WM(i+1,k-1) + WM(k,j-1) + a\}$$

After the recursion terminates, the optimal minimum free energy of the RNA structure is $W(L)$. The secondary structure corresponding to this optimal energy is found by tracking back the steps taken to calculate it.

3.2.3 TIME COMPLEXITY

The $W - V$ recursion takes $O(L^2), VBI$ recursion completes in $O(L^4)$ and the linear approximation of the multi-loop energy function reduces VM's time complexity to $O(L^3)$. Another constraint, which is often imposed, restricts the maximum internal loop size to a constant d (usually 30), dropping the complexity of the VBI recursion to $O(L^3)$. Therefore, the overall time complexity of the algorithm is $O(L^3)$ [2][1].

3.3 PROBABILISTIC STRUCTURE ANALYSIS

Transformational grammars [3] are a generative set of rules governing the production of sequences in a formal language. These grammars have been extensively used in natural language processing and have subsequently found their application in RNA sequence analysis. Chomsky sub-categorized transformational grammars into: *regular, context-free, context sensitive* and *unrestricted* [1] based on the amount of restrictions placed on the generating rules, illustrated in Figure 6.

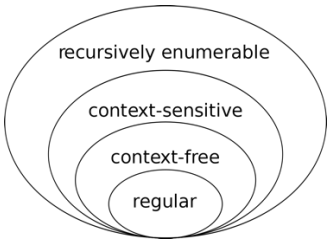


Figure 6: *Chomsky hierarchy of transformational grammars, nested according to the amount of restrictions placed on the production rules by the grammar, reprinted from [1].*

3.3.1 STOCHASTIC CONTEXT-FREE GRAMMARS

Context-free grammars are a subclass of transformational grammars suitable for modeling palindromic languages [3]. They differ from *regular* grammars due to their ability to produce two symbols simultaneously.

Formally, a *context-free grammar* G, is characterized by three distinctive sets – (1) *alphabet* set Σ, a finite set of *terminal* symbols observed in generated sequences. For RNA sequences, it is simply the set of four nucleotides $\{A, C, G, U\}$. (2) a *non-terminal* set N, containing n non-terminal symbols $W_1 \dots W_n$ and a start symbol S_0. (3) production set P defining rules of generating subsequences from non-terminals. A sequence s generated by G can be associated a specific *derivation* D, which is the set of sequentially applied production rules (starting from the nonterminal S_0) to obtain s. *Parse-tree* [1] is a tree structured representation of the derivation, with S_0 as its root node and leaf nodes representing terminal emissions. Grammars that have more than one legal derivation for any sequence belonging to the language are called *ambiguous* [3] and are a particularly difficult to model. Therefore, unambiguous grammars are predominantly used in RNA

sequence analysis which maintain a one-to-one relationship between a sequence and its derivation. Therefore, in RNA analysis, the derivation of an RNA sequence, more particularly, its parse-tree represents its secondary structure derived by the associated grammar.

3.3.2 PROBABILISTIC STRUCTURE PREDICTION MODEL

Stochastic context-free grammars (SCFGs) [3][1] that represent RNA sequences primarily utilize three types of production rules: (1) $S \rightarrow aS\hat{a}$ pair-emission rules emitting a complementary base-pair $(a, \hat{a})$ towards opposite ends of the sequence and a non-terminal symbol. (2) $S \rightarrow aS \mid S \rightarrow Sa$ single emission rules used to emit unpaired bases in the sequence. (3) $S \rightarrow SS \mid S \rightarrow S$ rules emitting non-terminal symbols to model branched secondary structures. Facilitating algorithm design, Chomsky designed a restrained form of SCFGs called *Chomsky normal form* (CNF) [1]. Any SCFG can be rewritten in CNF by modifying its production rules to only have two forms $-\ W_u \rightarrow W_y W_Z$ and $W_v \rightarrow a$.

SCFGs define a probability distribution over all possible sequences by associating production rules with a probability. For a given RNA sequence x and a SCFG G that models RNA sequences, the most likely secondary structure of x is essentially the most probable parse-tree for x under G.

3.3.3 COCKE–YOUNGER–KASAMI (CYK) ALGORITHM

The *Cocke–Younger–Kasami* (CYK) [1] algorithm is used to find the most likely parse-tree for an RNA sequence x. Assuming the grammar G is in CNF and has a non-terminal set $W = \{W_1, \ldots, W_N\}$ with W_1 being the start non-terminal. Let u, v, y and z represent indices

corresponding to the non-terminal set W. The probabilistic parameters for the grammar G are categorized into: *transition* probability $t_u(y,z)$ associated with the production rule $W_u \rightarrow W_y W_z$ and *emission* probability $e_v(a)$ for the production rule $W_v \rightarrow a$, where $a \in \Sigma$.

CYK is a dynamic programming algorithm operating on the variable $\gamma(i,j,u)$ which is the maximum likelihood measure of a parse subtree rooted at W_u non-terminal and generating the subsequence $x_i, \ldots, x_j \ \forall\, i,j \ \in (1,L)$ and $\forall\, u \ \in (1,N)$. The initialization and recursion steps for γ are given below.

1. Initialization: $\forall\, i \in (1,L)$ and $u \ \in (1,N)$

$$\gamma(i,i,u) = e_u(x_i)$$

2. Recursion: $\forall\, i \in (1,L-1),\ j \in (i+1,L)$ and $u \ \in (1,N)$

$$\gamma(i,j,u) = \max_{y,z} \max_{k=i\ldots j-1} \{\gamma(i,k,y) + \gamma(k+1,j,z) + t_u(y,z)\}$$

Once the recursion is complete, $\gamma(1,L,1)$ corresponds to the likelihood of the most probable parse-tree of the sequence x. The most likely parse-tree can be obtained by incorporating a traceback variable into the recursion step that notes all the non-terminal transitions and terminal emissions belonging to the derivation of x.

CHAPTER 4

STEM DETECTION

In retrospect, all nested RNA secondary structures are an amalgamation of stem structures, some contributing to hairpin-loops and some culminating in a multi-loop structure. Stems, as described earlier, are a contiguous set of multiple base-pairs resembling a double-helix. These helix structures exert a large influence on the stability and formation of RNA secondary structures. The stability of a helix is determined by factors such as its length, presence of bulges or internal loops and base-pair compositions. Naturally, stems are the most important motifs in RNA structure analysis. Accurate estimation of stem motifs can constrain the search space for prediction algorithms and reduce their polynomial time complexity.

Aldwairi *et al*. [18] developed a search heuristic to identify *hairpin-loop* motifs using sequence alignments, but this heuristic doesn't extend its approach to find the secondary structure of an isolated RNA sequence. In this chapter, we propose a classifier design routine to detect stem motifs from their primary sequences. The proposed workflow is overlaid in Figure 8.

Machine learning based pattern recognition methods are widely used to identify patterns and regularities in data. The design of a supervised learning model to detect stem motifs can be segmented into four steps: sequence/data acquisition, feature extraction, feature selection and model training. The overview of the design methodology is presented in Figure 7. Since we are only interested in stem structures, primary sub-sequences that fold into stems are extracted from RNA sequences along with sub-sequences that don't contain a contiguous stack of base-pairs. These sequences are extracted from various RNA families in the Rfam database [20]. A maximum

of four unpaired bases are arbitrarily concatenated at each end to add randomness to the obtained samples.

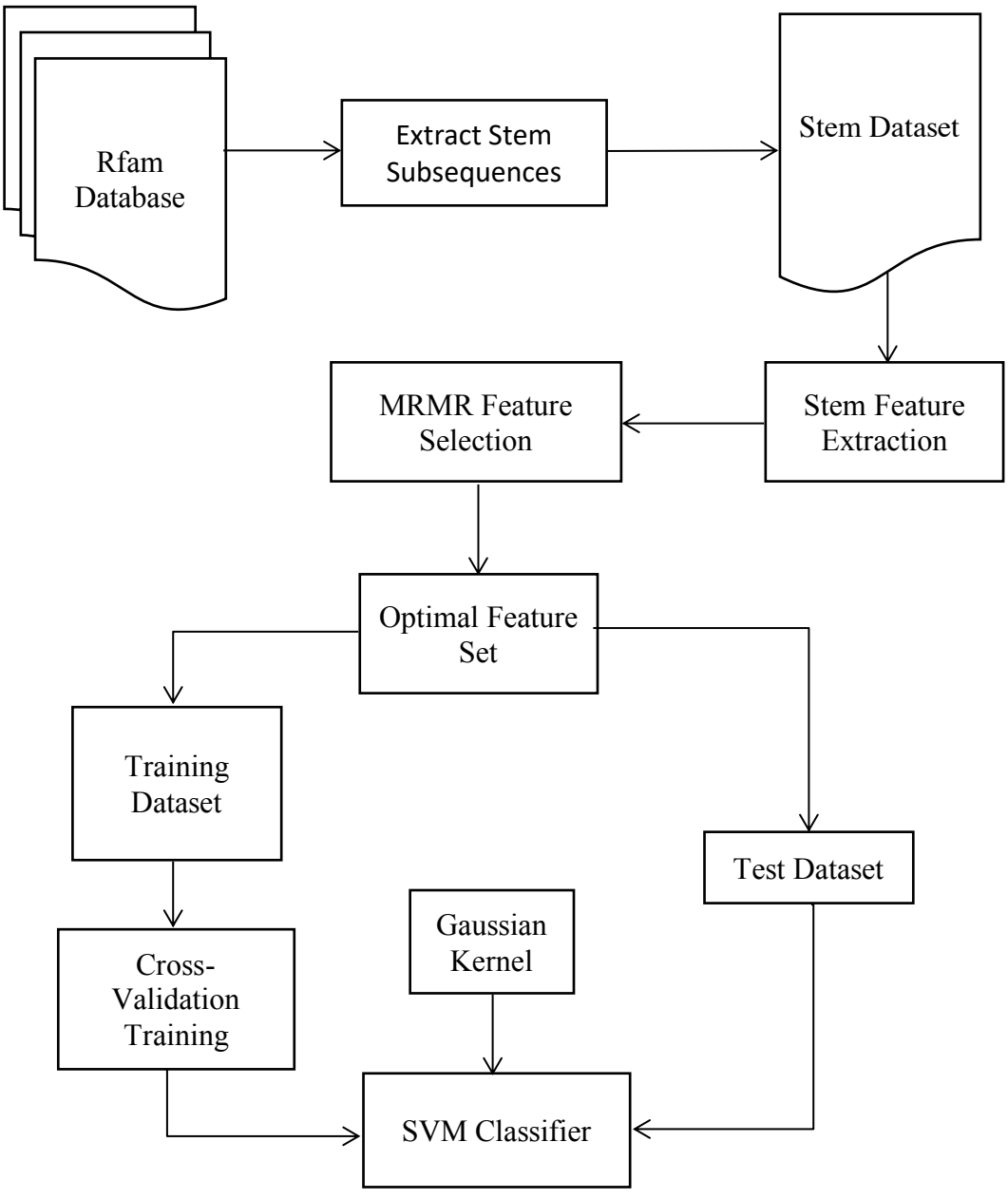


Figure 7: *Overview of the workflow of designing the stem detection classifier.*

The labeled dataset thus curated is denoted by, $X = \{x_1^0, \dots, x_n^0, x_1^1, \dots, x_n^1\}$, where $x_i^c;\ \forall i \in (1, n)$ is an extracted primary sequence that folds into a stem for $c = 1$ (*positive)* and doesn't form a stem for $c = 0$ (*negative*). The dataset has $2n$ data points and a uniform class distribution. It is further segmented into training and test sets, consisting of $2n/3$ and $4n/3$ samples respectively, retaining the uniform distributed of class in both the subsets.

4.1 STEM FEATURE EXTRACTION

Curated primary stem sub-sequences are not characteristic to their folding structures and cannot be directly used to train classification models. So, features that are deterministic to stem motifs are extracted from the primary sequence data. Given an RNA primary sequence s of length $|s|$, divide the sequence into two equal halves s_1 and s_2 corresponding to 5'-to-3' and 3'-to-5' directional branches of the helix respectively such that $s = (s_1, s_2)$. Subsequently, the following factors are computed for each sequence sample.

4.1.1 THE COMPOSITIONAL FACTOR

The *compositional* factor [21] is a single strand operator quantifying the proportion of nucleotides $\{A, C, G, U\}$ in the sample sequence s. For nucleotide a, let $n(a)$ denote the number of instances of this nucleotide in the sample sequence s. The nucleotide's compositional factor is therefore the ratio of the count of a to the total length of s given by $|s|$.

$$CF = \frac{n(a)}{|s|}\ ;a \in \{A, C, G, U\}$$

The compositional factor is four elements long as RNA sequences are made up of four different nucleotides.

4.1.2 THE TRI-TRANSITIONAL FACTOR

Analogous to the compositional factor, the *tri-transitional* factor [21] is also a single strand operator specifying the transition and composition of triplets in the sample sequence s. Let $n(a, b, c)$ denote the number of occurrences of the triplet $'abc'$ in s with consecutive transitions between a, b and c respsectively. The tri-transitional factor is computed as below.

$$TTF = \frac{n(a, b, c)}{|s| - 2} \; ; a, b, c \in \{A, C, G, U\}$$

a, b and c can represent 4 different nucleotides. Therefore, the total length of the tri-transitional factor is 64, considering all possible combinations of (a, b, c).

4.1.3 THE POTENTIAL BASE-PAIR FACTOR

The *potential base-pairing* factor [21] is a measure of the probability of occurrence of the Watson-Creek and Wobble base-pairs; $(A, U), (C, G)$ and (G, U) between both the strands. It is a coarse estimate of the maximum length of the sample sequences' stem structures. For all pairs $-(a, \hat{a})$, the factor is computed by counting the number of instances of nucleotide a in strands s_1 and $\hat{a}$ in strand s_2 and dividing the minimum of these counts with half the length of s. The potential base-pair factor for the pair $(a, \hat{a})$ is given by,

$$BPF = \frac{\min\{ n_1(a), n_2(\hat{a})\}}{|s|/2},$$

where $n_i(.)$ is a counting function operating on strand s_i and $\min(x, y)$ is the minimum operator between the two numbers x and y. Since the pairs $(a, \hat{a})$ and $(\hat{a}, a)$ are treated to be identical, the total number of elements in this factor is 3.

4.1.4 THE LARGEST STACK SIZE FACTOR

The *largest stack-size* factor is a stability estimate of the stem motif. Large stacks of contiguous base-pairs add stability to the stem and therefore are a good testament to the presumed stem structure. Let $a_i, \dots, a_{i+m-1}$ denote a segment of strand s_1 such that there exists a complementary segment $\hat{a}_j, \dots, \hat{a}_{j+m-1}$ within strand s_2 i.e., $\{(a_i, \hat{a}_j), \dots (a_{i+m-1}, \hat{a}_{j+m-1})\}$ are all valid base-pairs. The largest stack size factor is thus the maximum value of m, for which the aforementioned segments exist.

$$LSF = \frac{1}{|s|/2} \underset{m}{\text{argmax}}\ \exists\ (\hat{a}_j, \dots, \hat{a}_{j+m-1} : I^c([a_i, \dots, a_{i+m-1}], [\hat{a}_j, \dots, \hat{a}_{j+m-1}]);\ \forall i, j$$

where $I^c(X, Y)$ is an indicator function which takes the value of 1 if Y is a compliment of X and 0 otherwise.

4.1.5 THE NUCLEOTIDE RATIO FACTOR

The *nucleotide ratio* factor [21] inspects the effects of different ratios of nucleotides in strands s_1 and s_2 on the folded stem structure. This factor is always computed as a ratio

between the count of nucleotide a in s_1; $n_1(a)$ to the count of nucleotide b in s_2; $n_2(b)$. This factor comprises of 12 ratios based on the constraints on a and b described above.

$$NRF = \frac{n_1(a)}{n_2(b)} \; ; \forall a, b \in \{A, C, G, U\} \text{ and } a \neq b$$

The composition of the feature set constructed using these factors is give in Table 1. However, not all the features are *strongly* correlated to the stem classification label c as some of these features are either weak attributes of stem motifs or are redundant features. The most characterizing subset of features is selected from this set to minimize the classification error.

Identifier	Factor Name	Size
CF	Compositional factor	4
TTF	Tri-transitional factor	64
BPF	Potential base-pair factor	3
LSF	Largest stack size factor	1
NRF	Nucleotide ratio factor	12
	Total	84

Table 1: Complete feature set composition and factor sizes used in the design of stem detector.

4.2. MRMR FEATURE SELECTION

Feature selection methods are essentially a search algorithm operating under an optimization criterion, on the set of all spaces spanned by the features under scrutiny. Given a data set D consisting of N samples of M features $X_M = \{ x_i \; ; i \in (1, M) \}$ and classification label c, the feature selection problem finds an subspace of m features R^m from the observational space R^M such that $X_m = \{ x_k \; ; k \in (1, m) \}$ optimally characterize c.

Most algorithms ignore the relationship between features and solely select features that are maximally relevant based on F-scores or mutual information metrics. Overcoming this limitation, we employ the *minimal-redundancy-maximal-relevance* (mRMR) algorithm [22] which is a two-stage feature selection technique that accommodates both relevance to target label and dependency among features. The correlation between two random variables x and y can be captured using their mutual information which is defined in terms of the probability densities $p(x), p(y)$ and joint density $p(x, y)$ as:

$$I(x; y) = \iint p(x, y) \log \frac{p(x, y)}{p(x)p(y)} \, dx \, dy$$

The maximal relevance (dependency) criteria to select a feature subset X_m which captures the maximum mutual information between the selected features and class label c is given by:

$$D(X_m) = \frac{1}{m} \sum_{x_k \in X_m} I(x_k; c)$$

where x_k represents a feature in the selected feature set X_m containing m features. Analogous to the maximal relevance criteria, the minimal redundancy criteria can be defined for any two features $-(x_i,\ x_j)$ in the selected feature set X_m as

$$R(X_m) = \frac{1}{m^2} \sum_{x_i\ ;\ x_j \in X_m} I(x_i;\ x_j)$$

Combining these constraints, the MRMR operator is defined as $\Phi(X_m) = D(X_m) - R(X_m)$ and the optimization criterion to find the optimal subset X_m is given by

$$\text{argmax}\, \Phi(X_m)\ ;\ \Phi(X_m) = D(X_m) - R(X_m)$$

However, to reduce the complexity involved in the exhaustive search for feature subset X_m, an incremental search method is used to find the *near-optimal* features defined by Φ. In principle, assuming a feature subset of $m-1$ features is already selected in X_{m-1}, the search algorithm's task is to find the m^{th} feature from the set $\{X - X_{m-1}\}$. The incremental search's [21] optimization criteria given a selected set X_{m-1} is

$$x_m = \underset{x_k \in \{X - X_{m-1}\}}{\operatorname{argmax}} \left[I(x_k;\, c) - \frac{1}{m-1} \sum_{x_i \in X_{m-1}} I(x_k;\, x_i) \right]$$

The optimal feature subset selected for stem detection is illustrated in Table 2. We used these features to design the classification model from the training samples.

Identifier	Factor Name	Selected Features	Effective Size
CF	Compositional factor	G	1
TTF	Tri-transitional factor	GGG, GGC, CCC, AAA, GGA, UUU, AUA	7
BPF	Potential base-pair factor	GC, AU, GU	3
LSF	Largest stack size factor	-	1
NRF	Nucleotide ratio factor	G/C, U/C, G/U, A/G	4

Table 2: Selected optimal feature subset composition and effective factor sizes.

4.3 SUPPORT VECTOR MACHINES

The selected training set is handed down to train a binary support vector machine (SVM) classifier. SVMs are a widely used class of supervised learning models and have a well-established literature for data classification and regression. In principle, for a p-dimensional dataset, the SVM classifier

locates a $p-1$ dimensional hyperplane which separates the two classes such that margin between the plane and the nearest training data point (functional margins) is maximized. The optimal hyperplane known as the *maximum-margin hyperplane* is linear in p-dimensional space and the training samples which lie on its margins are called support vectors. Non-linear classification boundaries can be obtained through a kernel function that implicitly maps the data into a higher dimensional space where it is linearly separable. Once the classifier is trained, a new data sample is assigned a class label based on which side of the plane it falls.

In our classifier design, we used a non-linear gaussian kernel function as it produced the least classification error. The accuracy of the designed classifier and its cross-validation metrics are shared in Table 3.

Metric	Value
Sensitivity	0.934
PPV	0.905

Class Predictions	0	1
0	451	49
1	33	467

Table 3: Classification accuracy and confusion matrix of the designed stem detector.

In the next chapter, we demonstrate how the designed stem detection model can be used to predict the complete secondary structure of an RNA sequence and illustrate why this method is computationally much less expensive.

CHAPTER 5

CONSTRAINED STRUCTURE PREDICTION

Machine learning models have slowly begun to gain an impetus in RNA sequence analysis. Neural network-based structure prediction algorithms were introduced by Steeg et al [23] and extended by Liu et al [17] subsequently. Bindewald and Shapiro et al [13] developed *KNetFold* that utilizes a network of K-nearest neighbor classifiers to predict RNA structures using sequence alignment information. In this document, we introduce a hybrid constrained space algorithm; which retains the principles of traditional dynamic programming structure prediction methods while reducing their time complexity through pattern (motif) recognition. The ability of machine learning models to identify RNA structural motifs [18] can be used to construct a restricted search space for existing prediction algorithms. Imposing restrictions on the search space based on the identified motifs will improve computational efficiency due to fewer secondary structures to process. This chapter details the methodology and strictures enforced in our constrained approach which constructs a confined search space and the algorithm used to predict the final secondary structure.

5.1 CONSTRAINED SEARCH SPACE

Conventional dynamic programming algorithms, as discussed in Chapter 3, predict optimal secondary structures by modeling a search space spanned by all possible secondary structures of the RNA sequence x and choosing the structure that satisfies the optimality criterion. Therefore, for each subsequence $x_{ij} = \{ x_i \dots x_j \};\ i, j \in (1, L)$ where L is the length of sequence x, these algorithms presume that x_{ij} could fold into any of the commonly known motifs described in Section 2.2.2. The dynamic programming recursion evaluates the scoring metrics for all these

substructure motifs, but indeed selects the substructure that contributes to the optimal state. Such an exhaustive search usually comes at the cost of a large computational overhead as seen in the case of Zuker [2] and CYK [1] algorithms, which exhibit polynomial time complexity of the third order $O(L^3)$. Constrained structure prediction overcomes this computational overhead by constructing a restricted search space which confines parts of the sequence to only fold into a single type of motif – *Stems*, called the *primary* motif.

Therefore, the first step towards constructing a reduced search space S_r is to identify all the subsequences that comply with the primary motif constraint. The stem detector designed in Chapter 4 is plugged into this phase to identify such qualifying subsequences. Although, for any given RNA sequence, subsequences could range from a few bases in length to the complete set of L bases, transforming the stem detector's job into an exhaustive search operation. However, our algorithm avoids such exhaustive computations by employing a sliding window-based search heuristic. The window search approach is feasible due to a distinctive property of stem structures – given a stem formed by the contiguous set of base-pairs $\{(x_i, x_j), (x_{i+1}, x_{j-1}), \dots (x_{i+n}, x_{j-n})\}$, any contiguous subset of these pairs can also be called a stem structure. Therefore, stems that contain a large enough number of base-pairs can be broken down into a series of smaller individual stems. This property is the substantial incentive behind selecting stems as the primary motif.

5.1.1 SLIDING WINDOW SEARCH

The sliding window search heuristic is a frequently used tool in pattern recognition to process contiguous datasets. The principle involved in this approach is to slide a window W of fixed-size

across different portions of the RNA sequence, determining whether the subsequence masked by the window satisfies the primary motif criteria at each iteration.

Formally, given an RNA sequence x of length L, let W_i and W_j denote equally sized 5'-to-3' and 3'-to-5' directional strands of the window starting at i and j respectively, such that $|W| = 2w = |W_i| + |W_j|$ and $|W_i| = |W_j| = w$, where $|\,.\,|$ denotes the length of the window strand. The search proceeds by sliding the window strands across different portions of the RNA sequence, at increments of w, as shown in Figure 8. For each (W_i, W_j) configuration, optimal stem features (Table 2) are extracted from the spanned subsequence and the constructed feature vector f_{ij} is classified using the stem detector and subsequently the window is added to S_r.

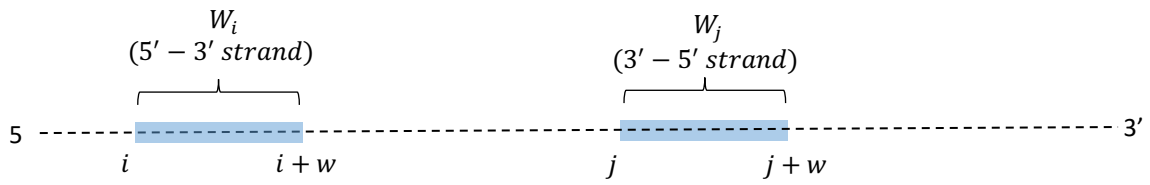


Figure 8: *The sliding window search heuristic demonstrated on an RNA sequence for the window configuration* $(\boldsymbol{W_i, W_j})$ *starting at* $\boldsymbol{i}$ *and* $\boldsymbol{j}$ *respectively.*

Algorithm

For $i = (1, w + 1, 2w + 1 \ldots L - 2w)$ and $j = (i + w, i + 2w \ldots L - w)$

Define $W_i = x(i, i + w),\ W_j = x(j, j + w)$ and $W = (W_i, W_j)$

$$f_{ij} \leftarrow \hat{X}_m(W)$$

$$S_r \leftarrow \begin{cases} S_r \cup W & , \quad if\ \Phi(f_{ij}) = 1 \\ S_r & , \quad if\ \Phi(f_{ij}) = 0 \end{cases}$$

Where, $\hat{X}_m$ is the optimal features used to train the stem detector and $\Phi(f)$ denotes the decision function of the SVM stem classifier.

5.1.2 OPTIMAL WINDOW SIZE

The fundamental parameter in the design of the sliding window search heuristic is the length of each window slice – w. Selecting a small window size w adds to the computational complexity and increases the number of incorrectly classified motifs due to the lack of adequate bases in the subsequence to conjure a strong prediction. On the opposing end, very large windows produce ambiguous results as the window might not fit a stem properly, inevitably leading to poor prediction outcomes. Therefore, selection of the optimal window size becomes a highly sensitive design decision which should accommodate most bulges, internal-loops and hairpin-loops sizes.

In order to come up with a practically optimum window size, we derived the size distributions of hairpin-loops, stems, bulges and internal-loops [18] from a dataset of 1000 RNA substructures randomly extracted from sequences pertaining to 150 different Rfam families. Inspecting these density estimates, illustrated in Figure 9, we noticed that hairpin-loop sizes mostly range from 3–11; stems are sized anywhere between 2–12 base-pairs long; bulges and internal loops are often not more than 8 bases in length.

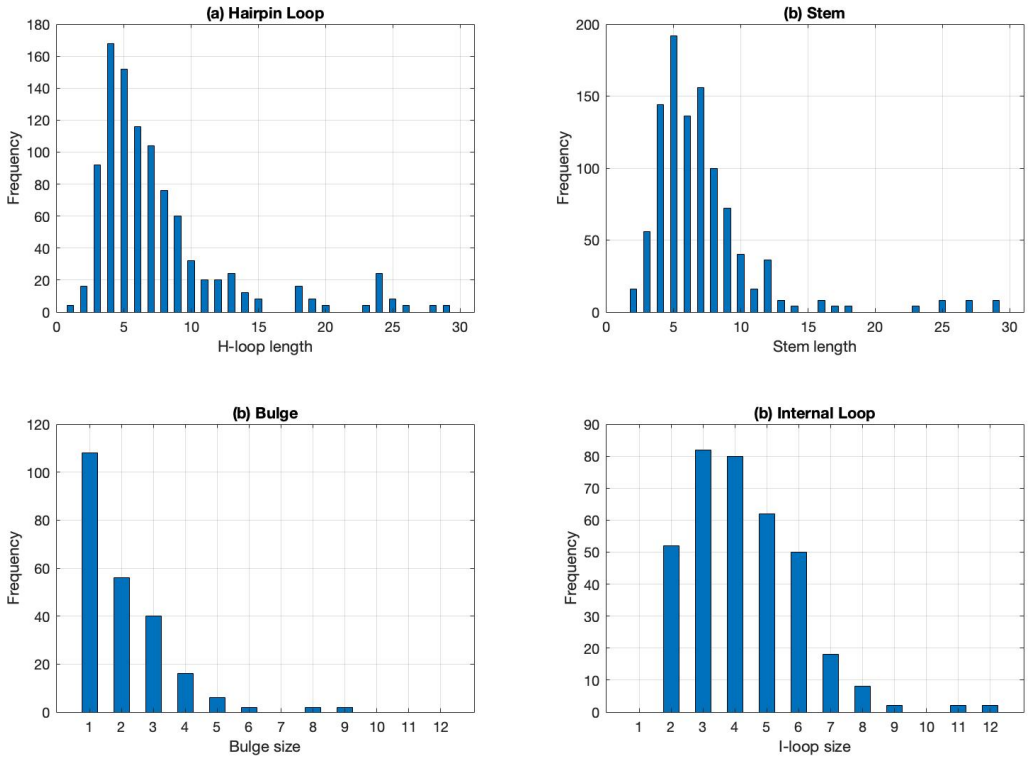


Figure 9: *Distribution of the sizes of (a) hairpin-loops, (b) stems, (c) bulges and (d) internal loop structures.*

The chosen optimal window size should overlap a majority of the stem portion while at the same time accommodating for loops and bulges that are usually not longer than 10 bases. Naturally, based on the inside provided from these distributional factors, a good choice of the window size would then be $w = 10$. This choice of w, checks all constraints discussed above while still remaining computationally efficient. Figure 10 plots the execution times of the sliding window search algorithm for a sequence of 300 bases at different window sizes. This execution time captures time taken in feature extraction and classification for every possible window configuration.

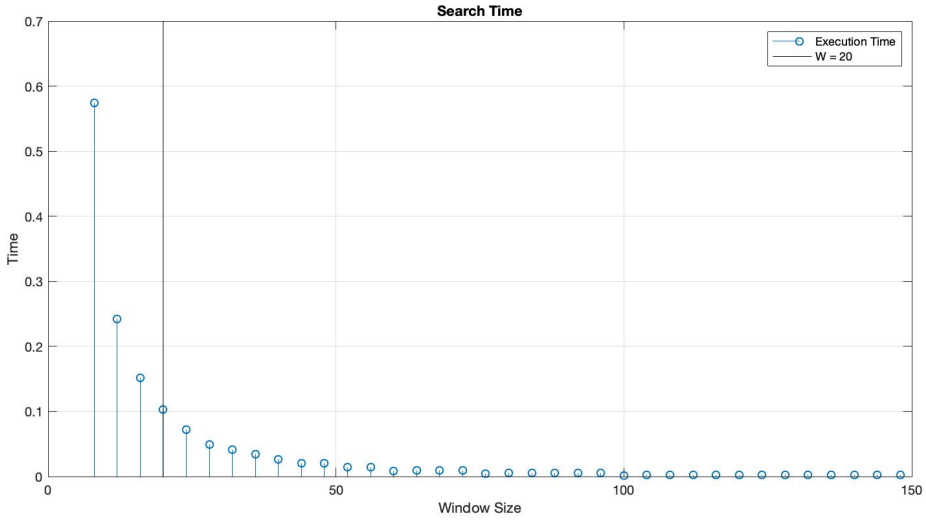


Figure 10: *Effects of the window size parameter 'w', on CPU search time for a fixed RNA sequence.*

5.2 STRUCTURE PREDICTION USING MESSAGE PASSING

Once a constrained structure space S_r is identified through the window search technique, the constrained prediction algorithm employs a message passing scheme over a graphical network of constrained subsequences to search for the optimal secondary structure in S_r. This section provides the network construction methodology and subsequently details its message passing infrastructure for structure prediction. In this document, we present an implementation of our constrained prediction method using a message passing algorithm [5][6] that utilizes the underlying principles of the energy minimization model. Interestingly, our method can also be constructed using a probabilistic model and the network can be configured as per the maximum likelihood principles.

5.2.1 NETWORK TOPOLOGY

Consider an RNA sequence x of size L, the constrained structure space S_r is defined by the set of N pairs of qualified window strand configurations $(i_1, j_1), (i_2, j_2), \ldots, (i_N, j_N)$, where i_k and j_k denote the starting indices of the 5'-to-3' directional window strand W_i and 3'-to-5' directional window strand W_j respectively $\forall k \in (1, N)$. Each window configuration (i_k, j_k) is regarded as a node in the graphical network G. The nodes in the network are connected to each other following a series of inheritance rules:

1. Node $A = (i_a, j_a)$ *inherits* from node $B = (i_b, j_b)$, $A \leftarrow B$, if its window configuration is enclosed by that of B i.e., $i_b < i_a < j_a < j_b$, where $a \in (1, N)$ and $b \in (1, N) - \{a\}$.
2. If this inheritance holds, then there exists a directed edge in G following the direction of inheritance $B \rightarrow A$ and A is called a *child* of *parent* B.
3. Each child node in G may inherit from more than one parent node, which in turn can have multiple child nodes.
4. G is a directed acyclic graph (DAG), which implies no two nodes can inherit from each other; either directly i.e., if $A \leftarrow B$ then $B \nleftarrow A$; or through cyclic inheritance i.e., if $A \leftarrow B \leftarrow \ldots \leftarrow C \leftarrow D$, then $D \nleftarrow A$.

Interpretation of the network topology: For any graphical structure G, following the inheritance rules described above, the nodes of its network can be classified into three types:

1. *H-loop Node*: Hairpin-loop nodes, as the name suggests, model hairpin-loop sub-structures. These nodes do not have any child nodes associated and often contain a

consecutive window configuration i.e., considering the node has a window configuration $(i_k,\ j_k)$, with each window slice of length w, its configuration satisfies the equality:

$$j_k = i_k + w$$

2. *Stem Nodes:* These nodes model stem sub-structures and must have at least one child node in G. If a stem node has multiple child nodes, then all pairs of child nodes should have overlapping window configurations. So, for any two child nodes A and B of *stem* node C, with window configurations (i_a, j_a) and (i_b, j_b) respectively, $i_a < i_b < j_a < j_b$ provided $i_a < i_b$.

3. *M-loop Nodes:* Multi-loop nodes are used to model multi-branched loop structures. An m-loop node should always have more than one child node. The principle difference between m-loop and stem nodes are that, unlike stem nodes, m-loop nodes have at least one pair of non-overlapping child nodes. If C is a m-loop node, then there exists at least one pair of A and B child nodes which (provided $i_a < i_b$) satisfy the non-overlapping criteria

$$i_a < j_a < i_b < j_b$$

where $(i_a,\ j_a)$ and $(i_b,\ j_b)$ denote the window configurations of nodes A and B respectively.

Let us consider the network topology G illustrated in Figure 11, which models a constrained structure space S_r defined by the set of 6 constrained subsequences represented as the nodes of $G = \{ A, B, C, D, E, F \}$. The edges in the graph symbolize the influence of child substructure on that of its parent. In G, A and B are the children of C $(C \rightarrow A; C \rightarrow B)$ which inherits from E $(E \rightarrow C)$; A is also a child of D $(D \rightarrow A)$ and F inherits from E $(E \rightarrow F)$.

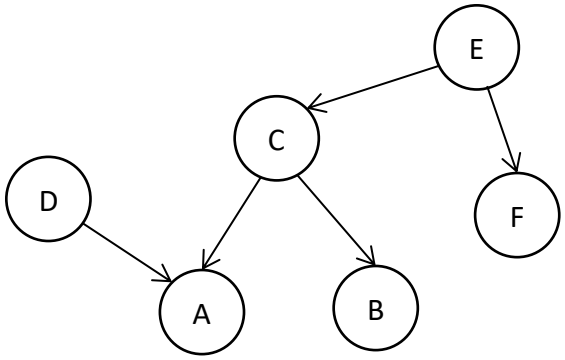


Figure 11: An example graphical parse network consisting of six nodes each corresponding to one of the identified stem motifs during window search.

Consequently, A, B and F are *h-loop* nodes and the substructures spanned by their window slices have stem structures which culminate in hairpin-loops whereas D is a *stem* node as it only has a single child node A. C and E could either be *stem* nodes or *m-loop* nodes based on their children's window configurations. If A and B have a non-overlapping window configuration, C is a m-loop node, else it is considered to be a stem node. Similarly, E is considered a m-loop node if C and F are non-overlapping and a stem node if they are. The behavior and role of each node in the structure prediction task is dictated by their kin.

5.2.2 MESSAGE PASSING SCHEME

This section introduces a novel scheme for searching the constrained search space S_r based on a simple message-passing approach, where messages are exchanged between the child and parent nodes of the graphical model G which describes the structure space S_r. Given a fully constructed network G whose edges corroborate the inheritance pattern between its nodes, the message passing scheme solves for the minimum free energy structure using the constraints established in S_r.

Messages in the network are sent opposite to the direction of inheritance [6], i.e., from children to parents as shown in Figure 12. Every node in the graphical model spans a certain sub-section of the RNA sequence which is presumed to contain a stem structure. Therefore, each node in G computes a corresponding sub-section of the energy matrix V that holds the minimum free energy of the substructures modeled by network G. Each h-loop node in the network initializes its own copy of the energy matrix V and populates its corresponding sub-matrix V_{ij}. Subsequently, this copy of V is sent across to respective parent nodes, which utilize the local sub-optimal energy calculations to put together the free energy of their subsequences. This exchange of messages across the graph is terminates when the energy matrix is passed over to the *root* nodes, which don't inherit from any other node in the graph.

In this message passing approach, nodes of the graphical model exhibit different behavioral and energy modeling patterns based on the class of nodes that they belong to. For an RNA sequence x of length L, the behavioral patterns of the three kinds of nodes introduced in the previous section is described henceforth.

1. H-Loop Nodes

   This class of nodes model hairpin-loop structures using free energy parameters eH, eS and eBI defined by the Zuker algorithm. Let (i, j) denotes the window configuration of the h-loop node H, then the free energy matrix V is initialized and updated using:

   - Initialization: $\forall\, m \in (1, L)$ and $n \in (1, L);\ V(m, n) = \infty$
   - Recursion: $\forall\, m \in (i, i + w)$ and $n \in (j, j + w)$

$$V(m,n) = \min \begin{cases} eH(m,n) \\ eS(m,n) + V(m+1,n-1) \\ VBI(m,n) \end{cases}$$

$$VBI(m,n) = \min_{\substack{2w-2 > m'-m+n-n' > 2 \\ m<m'<n'<n}} \{ eBI(m,m',n,n') + V(m',n') \}$$

where, $eH(m,n)$ is the hairpin-loop energy for the closing pair (m,n). $eS(m,n)$ is the energy of stacking base-pairs (m,n) and $(m+1,n-1)$. VBI models the energy of the internal loop or bulge closed by the (m,n) base-pair using $eBI(m,n,m',n')$, which describes the energy of a bulge or internal loop between the base pairs (m,n) and (m',n').

At the end of the recursion, the h-loop node modeling the substructure h, constructs the message comprising of 1) the energy matrix V, 2) the least free energy of h, $e_{\hat{m}\hat{n}}$ 3) corresponding closing base-pair $(\hat{m},\hat{n})$.

2. Stem Nodes

Nodes that belong to the stem class extend their children's stem structures. A stem node S, receives the sparsely computed energy matrix V, least free energy of child substructure $e_{\hat{m}\hat{n}}$ and corresponding base-pair $(\hat{m},\hat{n})$ after which it computes the free energy of the substructure s spanned by its window configuration (i,j). The recursion equation is defined $\forall\ m \in (i, i+w)$ and $n \in (j, j+w)$ by

$$V(m,n) = \min \begin{cases} eS(m,n) + V(m+1,n-1) \\ VBI(m,n) \end{cases}$$

Where,

$$VBI(m,n) = \min_{m<m'<n'<n} \{ eBI(m,m',n,n') + V(m',n') \}$$

$$\forall\, m' \in (i, i+w) + \{\hat{m}\} - \{m\};\ \forall n' \in (j, j+w) + \{\hat{n}\} - \{n\}$$

holds the free energy of the base-pair (m,n) which closes a bulge or an internal loop. $eBI(m,m',n,n')$ is the energy of the internal loop between base-pairs (m,n) and (m',n') and $eS(m,n)$ is the energy of stacking base-pairs (m,n) and $(m+1, n-1)$.

If the stem node has multiple children, it repeats this computation each time it receives a message from one of its children. If the minimum free energy of the modeled substructure h is lower than that computes previously, S updates the energy matrix V and constructs the message containing V, minimum free energy of h, $e_{\hat{m}\hat{n}}$ and corresponding closing base-pair $(\hat{m}, \hat{n})$.

3. M-Loop Nodes

Multi-loop nodes M have two or more non-overlapping children and therefore receive multiple messages from its children. For all possible combinations of non-overlapping child nodes, M computes the multi-loop energy of the loop consisting of stems modeled by its children. Let $(m_1, n_1), \ldots, (m_k, n_k)$ denote the set of k base-pair messages received from non-overlapping children set $X = \{X_1 \ldots X_k\}$, the recursion equation to compute the free energy of the multi-loop stem structure is given below.

For window configuration (i,j) of M, then $\forall\, m \in (i, i+w)$ and $n \in (j, j+w)$,

$$V(m,n) = \min \begin{cases} eS(m,n) + V(m+1,n-1) \\ VBI(m,n) \\ VM(m,n) \end{cases}$$

Where,

$$VBI(m,n) = \min_{m<m'<n'<n} \{ eBI(m,m',n,n') + V(m',n') \}$$

$$\forall\, m' \in (i, i+w) - \{m\};\ \forall n' \in (j, j+w) - \{n\}$$

is internal-loop energy for base-pair (m,n), $eS(m,n)$ is its stacking energy and the multi-loop energy for the non-overlapping children set X is given by defining WM matrix for linear multi-loop energy function:

$$eM(m,n,m_1,n_1,\dots,m_k,n_k) = a + bk + c\Big(m_1 - m - 1 + n - n_k - 1 + \sum_{p=1}^{k-1} [\, m_{p+1} - n_p - 1 \,] \Big)$$

$$VM(m,n) = \min_{u \in \{m_1 \dots n_1, \dots, m_k \dots n_k\}} \{ W(m+1,u-1) + W(u,n-1) + a \}$$

Where the W recursion matrix holds the minimum free energy of the subsequence (m,n) updated by each node correspondingly.

$$W(m,n) = \min \begin{cases} W(m+1,n) \\ W(m,n-1) \\ V(m,n) \\ \min\limits_{m<u<n} [\, W(m,u) + W(u+1,n) \,] \end{cases}$$

The recursion is repeated for all possible sets of non-overlapping child nodes. The computed energy matrix V is the one corresponding to the least minimum free energy of the multi-loop stem modeled by M. M-loop nodes construct the message to be passed along

to its ancestors by combining the energy matrix V, minimum free energy of the multi-loop stem $e_{\hat{m}\hat{n}}$ and corresponding closing base-pair $(\hat{m}, \hat{n})$.

Once the message passing algorithm is executed on G, the optimal energy matrix is selected from the set of its *root* nodes $r(G)$ which do not inherit from any of the nodes in G.

$$E = \min_{R \in r(G)} V(R)$$

where $V(R)$ denotes the energy-matrix calculated by the node R.

The minimum free energy of the total RNA sequence is captured in $W(1, L)$ and the optimal secondary structure is obtained by employing a traceback algorithm which incorporates auxiliary tracking variables which log the steps taken to compute $W(1, L)$.

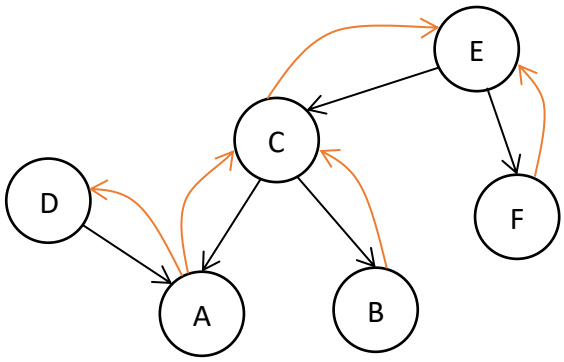


Figure 12: *Message passing scheme demonstrated on the synthetic network G. Messages are exchanged opposite the direction of inheritance starting at h-loop nodes and terminating at stem/m-loop nodes.*

Consider the example network illustrated in Figure 12. For h-loop nodes A, B and F, the message passing scheme initializes three individual energy matrices $V(A), V(B)$ and $V(F)$. These nodes

populate the sub-section of corresponding matrices and send these matrices to D, C and E. C computes the energy of its substructure based on what type of node it is and passes a message to E. E and D are *root* nodes and the message passing scheme terminates after they have completed populating their corresponding energy matrices. At the end of the message passing process, the secondary structure which exhibits the least free energy among the root nodes D and E is selected.

CHAPTER 6

PRACTICAL EXPERIMENTS

The focus of our work was to develop a fast RNA secondary structure prediction method with a lower computational complexity in comparison to conventional methods. As described in Chapter 3, dynamic programming algorithms have polynomial time complexity in the order of $O(N^3)$ for an RNA sequence of length N. This chapter discusses the time and space complexity of the constrained structure prediction model and presents a comparative analysis of the performance of Zuker's algorithm and our constrained MFE algorithm based on the experiments conducted.

6.1 COMPLEXITY

The complexity of an algorithm is a measure of its scalability and the resources it requires. The time complexity of our constrained method can be analyzed piecewise. The window search algorithm to detect possible stem structures takes $O(N^2 w)$ time to complete, where the N^2 factor is to iterate over the RNA sequence for different combinations of the window slices and the w factor is added due to the feature extraction aspect of classifying the subsequence. Similarly, the construction of the graphical network and the execution of the message passing scheme takes $O(N^2 w^3)$, since the maximum number of nodes in the network structure is N^2, which is how long it takes for the message passing scheme to execute and the energy calculations [2] at each node (which are similar to the Zuker recursion equations) adds the w^3 factor.

Therefore, the overall time complexity of the algorithm adds up to $O(N^2 w^3)$. Since the window strand parameter is a fixed constant, the true complexity of the algorithm is $O(N^2)$. Similarly, the

space complexity of the constrained prediction algorithm is $O(N^2w^2)$, and for constant w, it is $O(N^2)$. Each node in the constructed network utilizes w^2 space to compute the local energy matrix. Worst case, the N^2 nodes in the network end up using N^2w^2 memory.

6.2 PERFORMANCE MEASURES

The accuracy of a secondary structure prediction is estimated by calculating the number of base-pairs that it correctly identified and those that it did not. Therefore, Sensitivity (SN) and precision (PPV) are good statistical metrics used to analyze the performance of a structure prediction method. The SN and PPV of a prediction are given by

$$SN = \frac{TP}{TP + FN} \; ; \; PPV = \frac{TP}{TP + FP}$$

where TP denotes the number of correctly predicted base-pairs, FP denotes the number of bases that are incorrectly paired together and FN denotes the number of base-pairs which were present in the true secondary structure but are not predicted in the estimated structure.

6.3 RESULTS

The performance comparison between our constrained algorithm and other structure prediction algorithms can be made by computing the sensitivity and PPV of each algorithm's structure prediction and averaging these metrics over the set of all test sequences. To accomplish such a task, we put together a benchmark dataset comprising of all the seeded RNA sequences from 150 Rfam families [20]. The energy parameters shared by both the algorithms are obtained from the Vienna RNA structure prediction package [24]. For each family, the performance, both in terms of the accuracy of the prediction model and the execution time are recorded and averaged across the

members of the family. These family average sensitivities and PPV for a subset of the participating families comparing the constrained approach and Zuker algorithm are listed in Table 4. The overall performance measures averaging the sensitivity and PPV across all the participating families is given in Table 5.

ID	Rfam ID	Constrained Prediction		Zuker's Algorithm	
		Sensitivity	PPV	Sensitivity	PPV
1	*RF00001*	0.445	0.561	0.424	0.401
2	*RF00005*	0.713	0.739	0.685	0.655
3	*RF00007*	0.621	0.598	0.577	0.501
4	*RF00014*	0.508	0.610	0.520	0.593
5	*RF00023*	0.231	0.267	0.233	0.171
6	*RF00037*	0.659	0.903	0.666	0.843
7	*RF00057*	0.649	0.682	0.674	0.569
8	*RF00059*	0.597	0.618	0.612	0.442
9	*RF00106*	0.584	0.623	0.586	0.657
10	*RF00109*	0.411	0.400	0.269	0.296
11	*RF00162*	0.637	0.655	0.645	0.523
12	*RF00168*	0.496	0.648	0.500	0.466
13	*RF00515*	0.522	0.580	0.517	0.527
14	*RF00560*	0.241	0.359	0.227	0.256
15	*RF01067*	0.801	0.849	0.794	0.821
16	*RF01701*	0.566	0.598	0.572	0.435
17	*RF01739*	0.747	0.703	0.699	0.631
18	*RF01852*	0.790	0.787	0.739	0.714
19	*RF01855*	0.315	0.425	0.332	0.408
20	*RF02680*	0.592	0.551	0.537	0.470
21	*RF02683*	0.675	0.635	0.688	0.551

Table 4: Average Sensitivity and PPV of Constrained Prediction and Zuker's algorithm for a subset of the benchmarked Rfam family datasets.

Constrained Prediction		Zuker Algorithm	
Sensitivity	PPV	Sensitivity	PPV
0.584	0.609	0.590	0.521

Table 5: Average performance of constrained prediction and Zuker algorithm across the complete Rfam dataset.

6.4 DISCUSSION

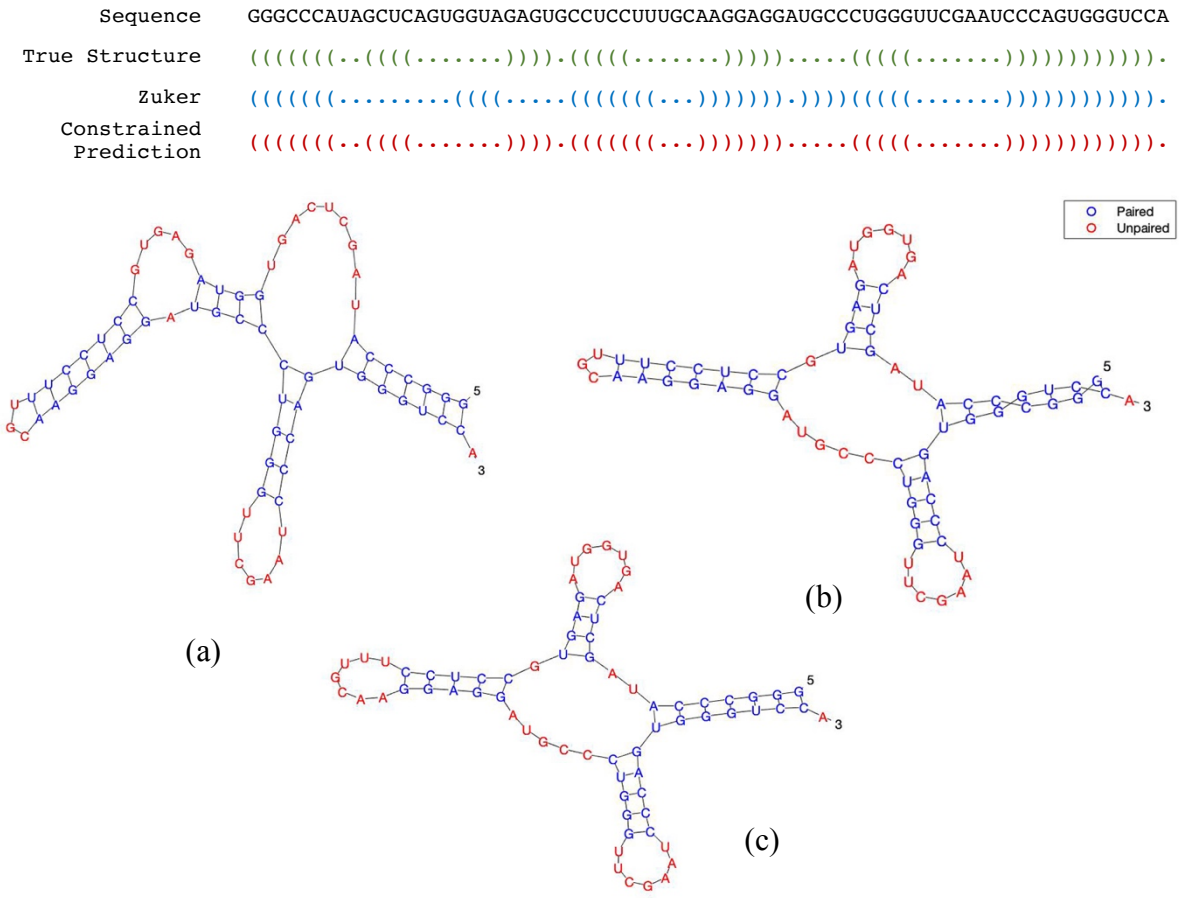


Figure 13: *Case study of a t-RNA structure prediction that benefits from modelling locally sub-optimal structures. Illustrated above are (a) Zuker prediction, (b) constrained prediction, and (c) true structure.*

In close assessment, we can characterize our constrained structure prediction process as a combination of local and global optimality. The method looks for locally optimal stem structures that best stitch together to produce a globally least free energy structure. The search for the global optimal structure is carried out under the local optimum criteria set by the stem detection step. This can lead to more accurate predictions as compared to the Zuker's algorithm [2] which only models globally optimal structures. However, sometimes this may also lead the prediction to be highly inaccurate when it is largely influenced by the local structures. This can be illustrated by looking at the following two t-RNA sequences which were both parsed using Zuker's algorithm and the constrained prediction model.

Consider the t-RNA structure depicted in Figure 13, Zuker's algorithm predicted a two stemmed multi-loop structure whereas the true structure has three stems. However, constrained prediction is more accurate as it caters to all local optima accurately. This ensures that all three stems of the multi-loop structure are selected while stitching the final optimal structure. This modularity in the constrained algorithm design can also be a bane. Consider the structure predictions shown in Figure 14, here the algorithm makes an inaccurate local optimum prediction and this affects the global structure prediction which can't discredit this false stem prediction.

Despite these discrepancies, the experimental results we obtained depict that the overall performance of our constrained prediction model is closely comparable to that of conventional MFE Zuker's algorithm, indicating that the algorithm more often than not overcomes the ill-effects of local optima. This aligns with the theoretical expectations, as constrained prediction algorithm implements the same underlying principle as Zuker's algorithm.

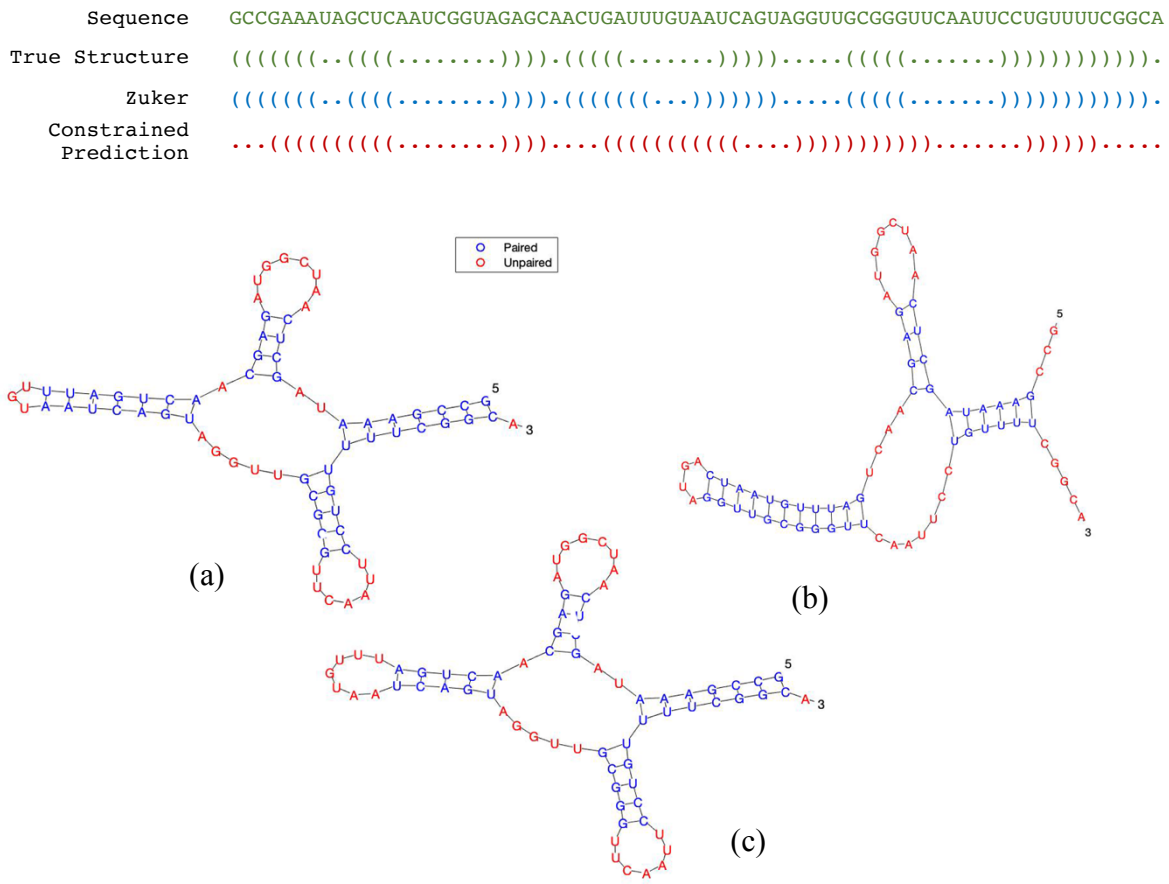


Figure 14: *Case study of a t-RNA structure prediction that is affected due to larger influence of locally sub-optimal structures. Illustrated above are (a) Zuker prediction, (b) constrained prediction, and (c) true structure.*

However, Figure 15 plots the average CPU time of both methods as a function of the average sequence length of the family. This comparison clearly depicts the advantage of using constrained models over traditional methods. The difference in the order of polynomial time complexity between the algorithms translates appropriately into the large amount of time taken by Zuker's algorithm for very large RNA sequences. In comparison, our constrained model is significantly faster for long sequences. However, as expected, this difference is not quite staggering for sequences of smaller length.

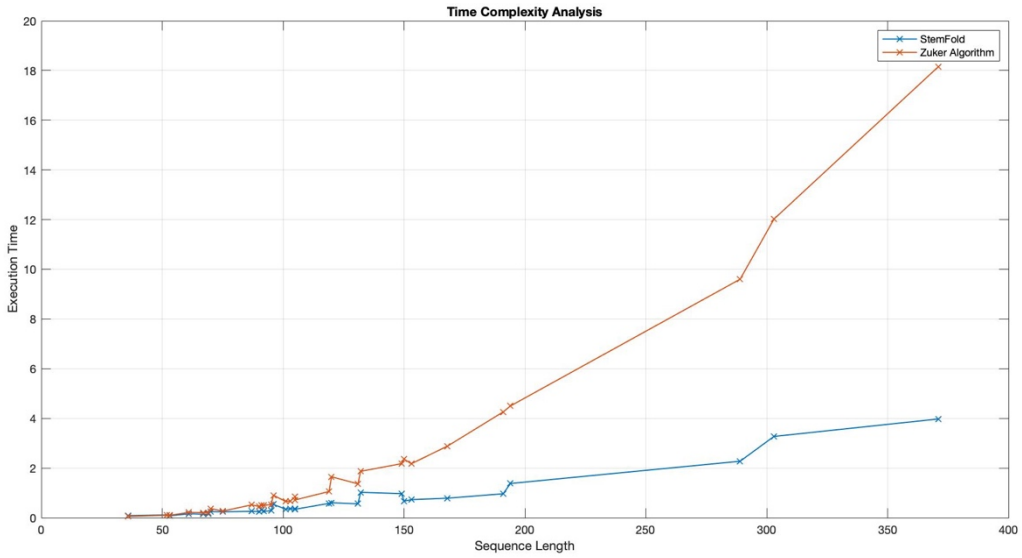


Figure 15: *Average CPU time comparison of constrained prediction method and Zuker algorithm.*

All experiments required to gather these performance metrics were performed using Matlab on a Macbook Pro computer with 2.4 GHz dual-core Intel Core i5 processor coupled with 8GB of internal memory.

CHAPTER 7

CONCLUSIONS

The material presented so far corroborates the efficiency of the constrained structure prediction algorithm and its superior computational complexity over conventional structure prediction algorithms. In this document, we presented a case study of the performance improvements when the constrained algorithm is developed using the minimum free energy model. However, our approach can also be extended to probabilistic models. Most probabilistic models operate on the set of all possible secondary structures to select its maximum likelihood estimate. Therefore, confining the search space could bore similar results and reduce the algorithm's time complexity.

Efficient structure prediction is an important aspect of RNA sequence analysis and our approach accounts for significant improvements in the prediction performance. The use of constrained models to analyze huge strands of RNA data could see significant reductions in processing time.

In general, even though the performance of our constrained prediction algorithm is comparable to conventional methods, the accuracy of the predictions is far from ideal as seen in the sensitivity measures tabulated in the previous chapter. Complex structure prediction algorithms have been developed over the past decade which significantly improve the prediction accuracy like the Hopfield Neural Network method developed by Steeg et al [23]. However, they have been slow to adapt either due to their complicated model or the lack of open-source implementations. These factors have been a major reason for the extensive use of thermodynamic and probabilistic models. Open source implementation of dynamic programming algorithms like `RNAfold` and `Mfold` [2] have been popular due to their robust implementation and frequent performance enhancements.

Therefore, an immediate strategy would be to improve the overall prediction accuracy by adapting the constrained prediction approach to work with advanced probabilistic models like Hidden Markov Models [1][8] (HMMs) and Covariance Models [1] (CM). We also plan to extend of the confined structure prediction model to estimate sequence alignments [5], which suffers from a larger computational overhead.

REFERENCES

[1] R. Durbin, S. Eddy, A. Krogh, and G. Mitchison, *Biological Sequence Analysis: Probabilistic Models of Proteins and Nucleic Acids*, Cambridge Univ. Press, Cambridge, UK, 1998.

[2] M. Zuker and P. Stiegler, *Optimal computer folding of large RNA sequences using thermodynamics and auxiliary information*, Nucleic Acids Res., 1981.

[3] R. D. Dowell and S. R. Eddy, *Evaluation of several lightweight stochastic context-free grammars for RNA secondary structure prediction*, BMC Bioinformatics, 2004.

[4] R. Nussinov and A. B. Jacobson, *Fast algorithm for predicting the secondary structure of single-stranded RNA*, Proc Natl Acad Sci USA, 1980.

[5] B. J. Yoon, *Sequence alignment by passing messages,* BMC Genomics, 2014.

[6] D. Lin and R. Goebel, *Context-Free Grammar Parsing by Message Passing,*1993.

[7] K. J. Won, T. Hamelryck, A. Prgel-Bennett and A. Krogh, *An evolutionary method for learning HMM structure: prediction of protein secondary structure*. BMC Bioinform., 2007.

[8] B. J. Yoon and P. P. Vaidyanathan, *RNA secondary structure prediction using context-sensitive hidden Markov models,* Proc. International Workshop on Biomedical Circuits and Systems (BioCAS), 2004.

[9] B. J. Yoon and P. P. Vaidyanathan, *Optimal alignment algorithm for context-sensitive hidden Markov models*, Proc. 30th IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP), 2005.

[10] Francis Crick, *Central dogma of molecular biology*, Nature, 1970.

[11] I. Tinoco and C. Bustamante, "*How RNA folds"*, Journal of Molecular Biology, 1999.

[12] Z. Sükösd and B. Knudsen, *Multithreaded comparative RNA secondary structure prediction using stochastic context-free grammars,* BMC Bioinformatics, 2011.

[13] E. Bindewald and BA. Shapiro, *RNA secondary structure prediction from sequence alignments using a network of k-nearest neighbor classifiers,* RNA, 2006.

[14] J. Berg, L. Stryer and J. Tymoczko, *DNA, RNA, and the Flow of Genetic Information*, 2007.

[15] D. M. Layton and R. Bundschuh, *A statistical analysis of RNA folding algorithms through thermodynamic parameter perturbation*, Nucleic Acid Research, 2005.

[16] Y. Ding, CE. Lawrence, *A bayesian statistical algorithm for RNA secondary structure prediction*, Computational Chemistry, 1999.

[17] Q. Liu, X. Ye, Y. Zhang, *A Hopfield Neural Network Based Algorithm for RNA Secondary Structure Prediction*, Computer and Computational Sciences, 2006.

[18] M. Aldwairi and R. Duwairi, *A Classification System for Predicting RNA Hairpin Loops*, International Joint Conference on Bioinformatics, Systems Biology and Intelligent Computing, 2009.

[19] B. Knudsen and J. Hein, *Pfold: RNA secondary structure prediction using stochastic context-free grammars*, Nucleic Acids Research, 2003.

[20] I. Kalvari, J. Argasinska, N. Quinones-Olvera, E.P. Nawrocki, E. Rivas, S.R. Eddy, A. Bateman, R.D. Finn, and A.I. Petrov, *Rfam 13.0: shifting to a genome-centric resource for non-coding RNA families*, Nucleic Acids Research, 2017.

[21] C. H. Chang, C.-B. Yang, *Accuracy Improvement for RNA Secondary Structure Prediction with SVM*, 2008.

[22] H. Peng, F. Long and C. Ding, *Feature Selection Based on Mutual Information: Criteria of Max-Dependency, Max-Relevance, and Min-Redundancy*, IEEE Transactions on Pattern Analysis and Machine Intelligence, 2005.

[23] Evan W. Steeg, *Neural Networks, Adaptive Optimization, and RNA Secondary Structure Prediction*, 1993.

[24] R. Lorenz and Stephan H. Bernhart, *ViennaRNA Package 2.0*, Algorithms for Molecular Biology, 2011.