AUTHORIZATION TO LEND AND REPRODUCE THE THESIS

As the sole author of this thesis, I authorize Brown University to lend it to other institutions or individuals for the purpose of scholarly research.

Date _____ _____

Tamaki Hiratsuka, Author

I further authorize Brown University to reproduce this thesis by photocopying or other means, in total or in part, at the request of other institutions or individuals for the purpose of scholarly research.

Date _____ _____

Tamaki Hiratsuka, Author

An Algorithm to Compute the Secondary Structure of tRNA Molecules

by
Tamaki Hiratsuka
ScM, Brown University, 2007

Thesis

Submitted in partial fulfillment of the requirements for the
Degree of Master of Science in the Department of Computer Science
at Brown University

PROVIDENCE, RHODE ISLAND
May 2007

This thesis by Tamaki Hiratsuka is accepted in its present form
by the Department of Computer Science as satisfying the
thesis requirements for the degree of Master of Science.

Date _____ _____

Franco Preparata, Advisor

Approved by the Graduate Council

Date _____ _____

Sheila Bonde, Dean of the Graduate School

Preface and Acknowledgments

I would like to thank my advisor Professor Franco P. Preparata for guiding me through this educational journey. His insightful guidance was only overshadowed by the generous donation of his time. His expert tutelage proved to be the shining light when all other solutions had failed. I am thankful for having been lucky enough to glean just a fraction of his seemingly endless knowledge.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

One of the most remarkable characteristics of transfer RNA (tRNA) molecules, despite the variations in their sequences, is that all tRNAs share the same cloverleaf secondary structure (figure 1.1). Some algorithms have been published to compute the secondary structure of RNA molecules. However, when applied to tRNA sequences, their success rate is low. In this paper, I propose a new algorithm to compute the secondary structure of tRNA.

Base pair maximization, free energy minimization, and covariation are some of the dynamic programming algorithms that have been published to compute the secondary structure of RNA molecules. As the name suggests, base pair maximization opts to maximize the number of base pairs in the resulting secondary structure and does not take into account the kinetics of folding. Therefore, this algorithm may not be very useful in predicting the realistic folding of RNA molecules. The other two algorithms have shown some promise, especially free energy minimization.

The theory behind the free energy minimization algorithm is that by folding on itself, an RNA sequence is able to achieve a more stable conformation than its linear configuration. Therefore, a given RNA sequence continues to fold until the most stable conformation has been obtained. The minimum free energy algorithm aims to

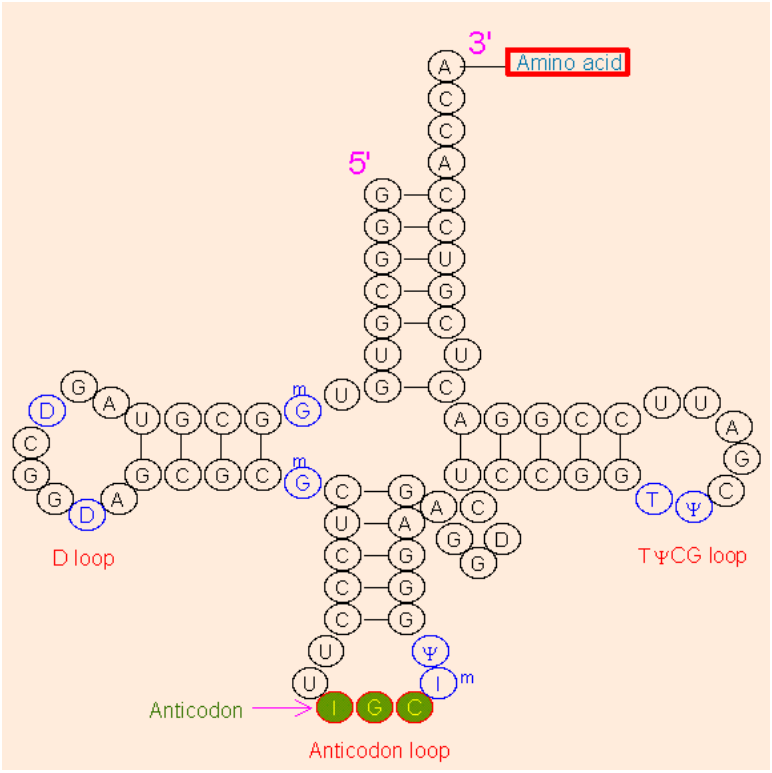

Figure 1.1: tRNA cloverleaf secondary structure: blue indicates modified nucleotides and "m" stands for "methylated". Bases labeled "D" are dihydrouridines and those labeled ψ are pseudouridines. Finally, "I" stands for inosine, a "special" base that can hydrogen bond with A, U, or C and enables the wobble hypothesis. [20] Anticodon is the trinucleotides complementary to a codon on mRNA. An extra arm is located between the anticodon stem and the TψC stem and is variable in length. [9]

model such folding kinetics.

Despite the plausibility of this theory, when applied to tRNA sequences, the minimum free energy algorithm performs rather poorly. One of the reasons for the low success rate may be because the algorithm computes a single optimal structure and does not take into account the folding mechanism. It is reasonable to assume that folding aims to achieve the most stable conformation. However, folding occurs over time and the final configuration may be composed of locally optimal substructures rather than the overall globally optimal structure. In my algorithm, I model a folding process as a function of time. Specifically, I propose that, initially, folding occurs locally. The local folding leads to the formation of the D, anticodon, and TψC arms.

Upon completion of the three arms, the ends of the sequence bind to form the acceptor arm.

Chapter 2

Background

2.1 RNA

RNA molecules can be divided into two categories: coding and non-coding. A coding RNA, such as messenger RNA (mRNA) carries genetic information transcribed from DNA. Using codons, triplets of nucleotides, mRNA encodes a sequence of amino acids. A non-coding RNA, on the other hand, functions without being translated into a protein.[6] One such RNA is a ribosomal RNA (rRNA). rRNA provides an environment in which proteins can be synthesized by translating the genetic codes embedded in mRNAs. Another type of non-coding RNA is transfer RNA (tRNA). tRNA is a relatively small molecule of seventy to ninety nucleotides in length. Its function is, using its anticodon, to "transfer" a specific amino acid coded by the mRNA being translated and "add" it to the growing chain of amino acids until a mature protein has been synthesized. Generally, a protein in an organism is comprised of up to twenty amino acids. Each of the twenty amino acids has at least one unique tRNA molecule assigned to it. Therefore, an organism has at least twenty, usually many more, different sequences of tRNA molecules and they all share the same cloverleaf secondary structure (figure 1.1).

A secondary structure of an RNA is composed of two substructures: helices and loops. A helix, also called a stem or a stack, is an intramolecular pairing of bases, and a loop is an unpaired region of the molecule. There are four types of loops: hairpin loop, bulge loop, internal loop, and multiloop. Therefore, a typical secondary structure is composed of up to five canonical structures. These elementary structures are illustrated in figure 2.1. In essence, a secondary structure is a two-dimensional definition of which nucleotides bind each other via hydrogen bonding. It is important to note that a molecule's structure is often critical in its correct function and that is why much attention is paid to folding algorithms.

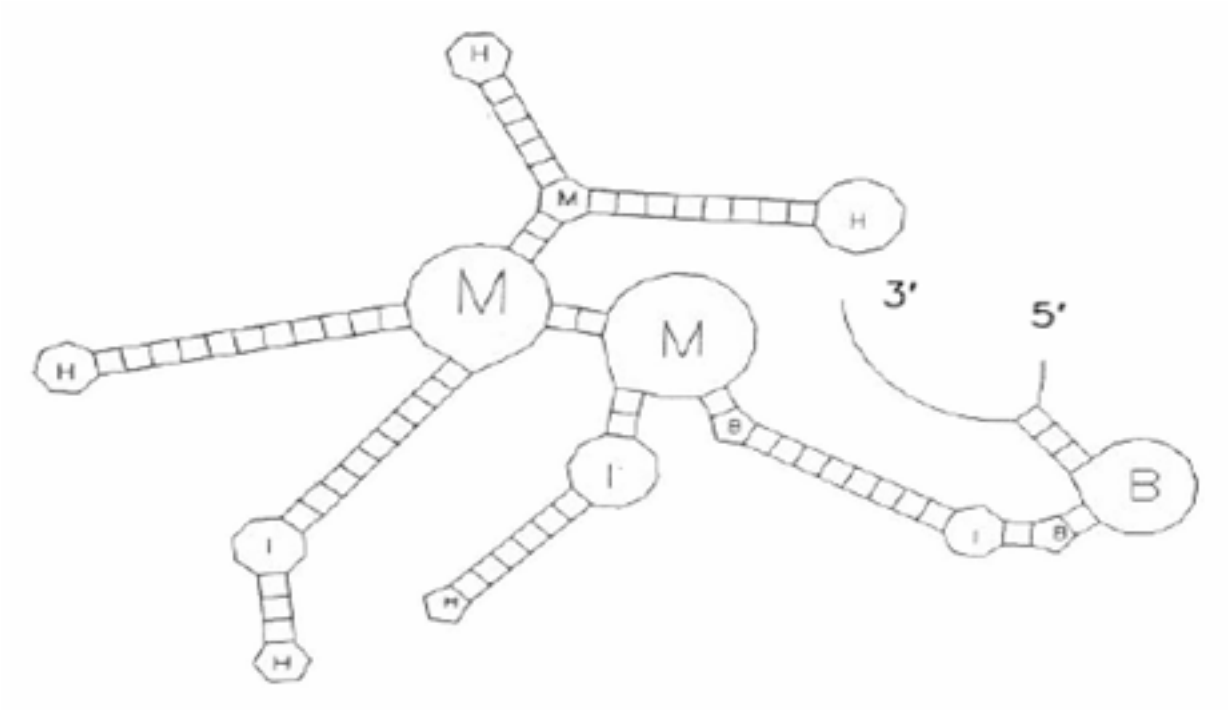

Figure 2.1: Canonical structures of an RNA secondary structure. B = bulge loop, H = hairpin loop, I = interior loop, and M = multiloop. Consecutive pairing of bases (ladder-like representation) is called a stack.[7]

Typical base pairs in RNA molecules are AU and CG. They bind via hydrogen bonding, illustrated as red dotted lines in figure 2.2.

Any given RNA molecule has four levels of structures. The first level is called a

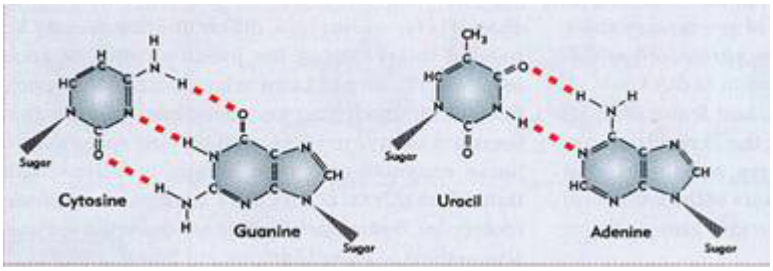

Figure 2.2: Watson-Crick base pairs of RNA molecules[1]

primary structure, which is a linear sequence of four bases: adenine, guanine, cytosine, and uracil. The next level is a secondary structure, which is a two-dimensional conformation that results from the molecule folding upon itself due to intramolecular attraction. A tertiary structure is formed when a molecule in its secondary structure folds upon itself to produce a three-dimensional configuration. Lastly, a quarternary structure is assembled when two or more molecules, in their tertiary structure conformations, interact with one another to create a three dimensional, multi-component complex. Even though molecules are only functional in their tertiary or quarternary structure, the secondary structure is equally critical, because the secondary structure is an intermediary step toward the functional conformation. In other words, the correct secondary structure is a prerequisite for the folding of a functional molecule. Secondary structure's relative ease of computation makes it an interesting subject for computer scientists. [8].

The cloverleaf secondary structure of tRNA (figure 1.1) is comprised of a multiloop, which is composed of four base-paired arms (acceptor, D, TψC, and anticodon arms) and three hairpin loops (D, TψC, and anticodon loops). [10] There is also an extra arm that may or may not be present, depending on the length of the tRNA sequence. The longer the sequence, the more likely that the extra arm is present. If it is present, it is located between the anticodon arm and the TψC arm, and it is usually an unpaired segment of bases that varies in length depending on the tRNA sequence.

The D arm is a three or four base-pair stack ending in the D loop that is usually up to eleven bases in size. The name D loop is derived from the frequent presence of dihydrouridine (D) in the loop. The anticodon arm is usually a five base-pair stem ending in the anticodon loop that is almost always seven bases in size. This loop contains the three base anticodon, which pairs with the codon on mRNA that codes for a specific amino acid. The 5' base of the anticodon is frequently modified into inosine (base with low binding specificity). The TψC arm is usually a five base-pair stem ending in the TψC loop that is up to ten bases in size. The name TψC loop comes from the presence of a TψC (ψ=pseudouridine) sequence in the loop. Finally, the acceptor arm is often a seven base-pair stem that is formed by the bases in the 3' end and the bases in the 5' end via hydrogen bonding.

tRNA is synthesized in a precursor form in both eukaryotic and prokaryotic cells. [13] For these tRNAs to become functional, they must be processed. The processing of tRNA precursors involves several distinctly different events [13] such as illustrated in figure 2.3. [4]

1. The 5' leader sequence must be removed by an enzyme called RNase P.
2. The 3' trailer sequence must be removed by a combination of enzymes: endonucleases and exonucleases.
3. A short sequence of CCA must be added to the 3' terminal.
4. Splicing of some introns occurs in some tRNAs.
5. Base modifications must occur at multiple positions.

The post-transcriptional modifications are perhaps the most important in considering the folding of tRNA molecules by computational algorithms, because these bases may no longer be able to bind their original partners due to the modifications. As discussed in the Results section, these changes may be an important factor that could raise the

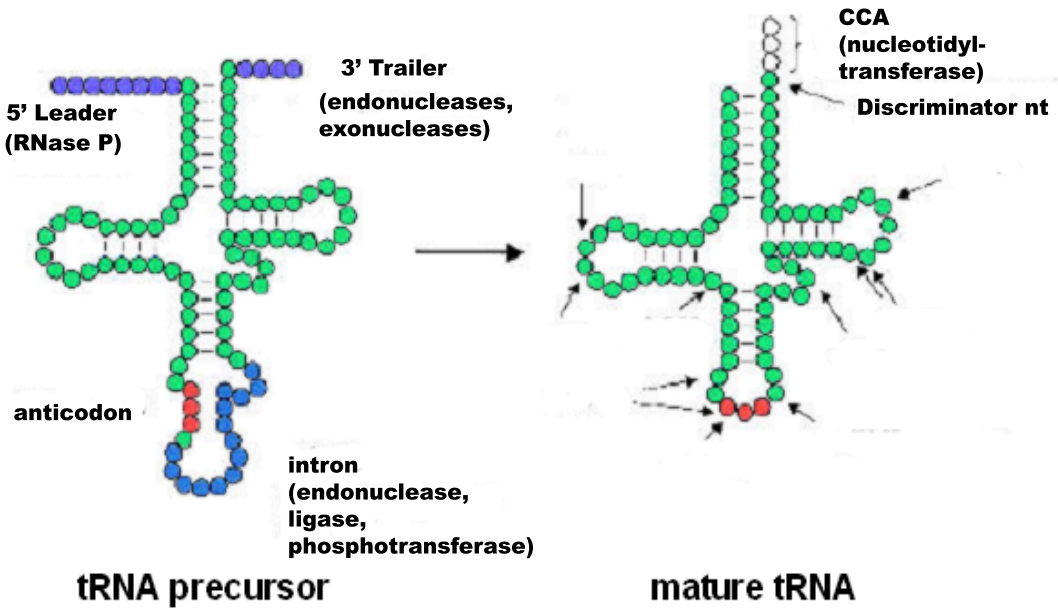

Figure 2.3: The maturation process of a tRNA: arrows pointing to the mature tRNA indicate bases that are possibly modified.[4]

accuracy of the algorithm. Typical post-transcriptional base modifications involve at least eight bases that are chemically modified. Dihydrouridine and pseudouridine mentioned above are the results of the modification process.[25] In addition, some bases are altered through the means of methylation.[9] Lastly, adenine can be modified into inosine, which is a "special" base that can bind A, U, and C.[3] This alteration is an important modification process in tRNA, because it helps to realize the wobble hypothesis. [20]

The wobble hypothesis was proposed by Francis Crick to generalize an observation of the presence of inosine at the 5' position of the anticodon. Crick suggested that while the interaction between the codon in the mRNA and the anticodon in the tRNA needed to be exact in two of the three nucleotide positions, this did not have to be so in the third position. He proposed that non-standard base-pairing might occur

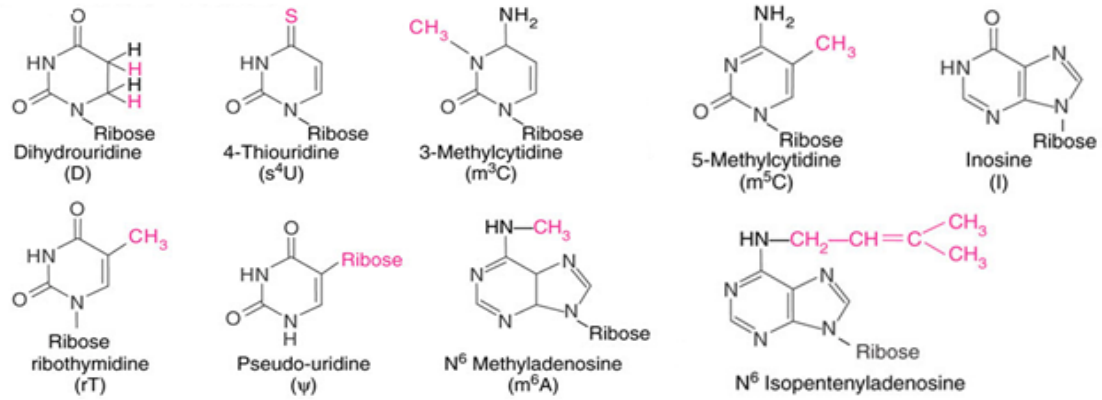

Figure 2.4: Examples of modified bases in tRNA. Those in pink represent the modified part of the base.[4]

between the nucleotide base in the 5' position of the anticodon and the 3' position of the codon.[2] This proposal is now known as the wobble hypothesis.

Currently, two functions of the post transcriptional base modifications have been identified. One is to stabilize the L-shaped tertiary structure. The stabilization is mainly achieved by the D-loop and the TψC loop. The other is to contribute to the wobble hypothesis and to the ribosome interaction, precise codon-anticodon pairing, and the correct recognition by the aminoacyl-tRNA synthetases. Aminoacyl-tRNA synthetases are enzymes that catalyze the binding of an amino acid to a tRNA. [22] In addition, figure 2.4 illustrates some of the modified bases. As can be observed, some of these bases are no longer able to form hydrogen bonds with their originally complementary bases due to steric hindrance or changes in the chemical properties. As previously mentioned, because the folding of a correct tertiary structure strongly depends on its correct secondary structure, the roles of the base modifications in the tertiary structure indicate that they most likely dictate the folding of a tRNA into the cloverleaf secondary structure and contribute to the stabilization of the conformation. After all, the secondary structure provides a geometric, kinetic, and thermodynamic

skeleton for tertiary structure formation.[15].

2.2 Previous Work

The Vienna package [17] [27] and RNAStructure 4.2 [14] [21] are some of the programs that compute the secondary structures of RNA molecules by implementing the free energy minimization algorithm.

Among many programs it offers, the Vienna package [17] [27] implements three dynamic programming algorithms that compute and predict the secondary structure(s) of an input RNA sequence. One such algorithm is the partition function algorithm (McCaskill 1990) that calculates base pair probabilities in the thermodynamic ensemble. Another implementation is of the suboptimal folding algorithm (Wuchty et.al 1999) which generates all suboptimal secondary structures within a given range of the optimal energy. The third algorithm is the minimum free energy algorithm (Zuker & Stiegler 1981) [27], which is implemented by the RNAfold program included in the Vienna package [17] [27].[18] It computes the lowest possible free energy that a given RNA sequence can achieve and yields a single optimal structure. It uses experimentally obtained energy parameters and an energy function for each canonical structure shown in figure 2.1. The package also implements tools for secondary structure comparison and an algorithm to design sequences for which a predefined structure is provided (inverse folding).[17] [12]

RNAStructure 4.2 [14] [21] is another program that is designed to compute and predict the secondary structure of RNA molecules. It includes a sequence editor, an integrated drawing tool, the OligoWalk program, OligoScreen, Dynalign, and a partition function. Most importantly, it implements the newer version of the Zuker algorithm (2004) than the Vienna package [17] [27] for free energy minimization, which is used to compute the secondary structure of an input RNA sequence. It uses the

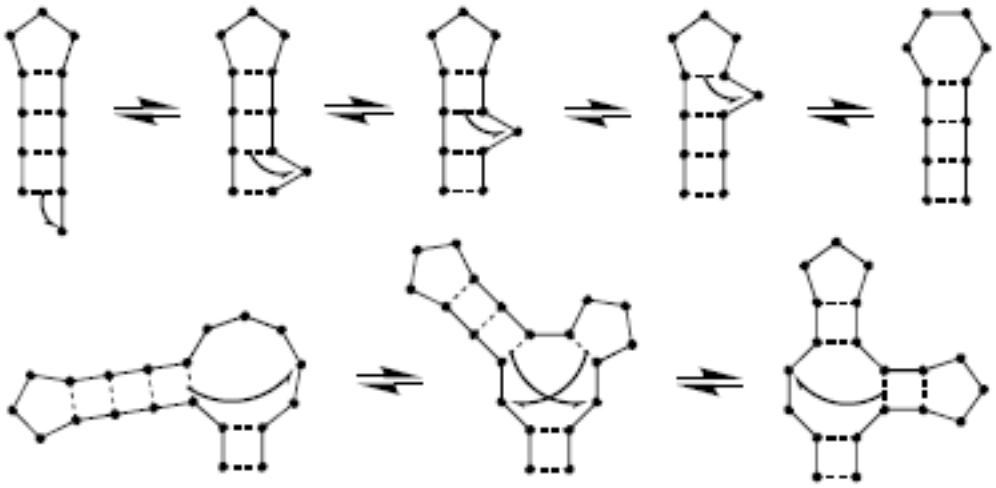

Figure 2.5: Defect diffusion and helix morphing. [15]

most current thermodynamic parameters available (at the time of implementation) from the Turner lab for each of the five canonical structures illustrated in figure 2.1. This is the algorithm I will focus on for the rest of my thesis. It also implements the modified recurrence functions that more accurately model the free energy of the folding process. [17] [21]

In addition to the free energy minimization approach, several other methods have been introduced to compute the secondary structure of RNA molecules.

One such method was published by C. Flamm, W. Fontana, I. L. Hofacker, and P. Schuster. They proposed the idea of move sets in their algorithm and implemented the "defect diffusion" concept published by Pörschke, 1974.[26] The defect diffusion concept basically states that intermediate formation of helices with incomplete base pairing occurs with a relatively high probability in RNA folding and that these mismatched regions are fixed by a fast chain sliding process illustrated in figure 2.5. [15]

Another algorithm is published by Wilfred Ndifon, who suggests modeling the kinetic folding of RNA molecules as a complex adaptive system (CAS). He defines

the kinetic folding of RNA as a time-series of structural rearrangements such as formation, dissociation, and shifting of individual base pairs. The characteristics of a CAS are the presence of a diverse mixture of components that participate in local interactions and an autonomous process that selects a subset of the components for thermal stability enhancement. In a CAS, the component-level dynamics produce global properties such as self-organization. The goal of the algorithm is to achieve a structure with the strongest thermal stability. [23]

Papanicolaou, Gouy, and Ninio took a different approach to improving the algorithm performance. They established a new energy model for tRNA and 5S RNA sequences. They tested the model with different tRNA and 5S RNA sequences and modified the energy parameters while adhering to certain rules that they established. The aim of the parameter modification was to increase the success rate of the algorithm. The finished model led to more than 80% correctly computed secondary structures. [24]. Similar increase in the success rate is observed for 5S RNA molecules. However, the authors note that the model is likely to be biased toward producing the correct secondary structures of tRNA and 5S RNA molecules and may not be applicable to other RNA strands nor reflect the real-world energy values of each canonical structure.

Chapter 3

Algorithm

My method for computing the secondary structure of tRNA combines the free energy minimization algorithm and the lowest free energy path algorithm: the FEM&LFEP algorithm. Free energy minimization is computed via dynamic programming that aims to minimize the overall free energy of the final structure. The lowest free energy path algorithm computes a combination of non-overlapping canonical structures that achieves the most stable conformation possible.

The idea behind free energy minimization is, as mentioned earlier, that the folding of an RNA molecule occurs in order to achieve a more stable conformation than its linear or previous configuration. Therefore, as the molecule folds on itself and forms hydrogen bonds between pairs of complementary bases, these pairings in turn stabilize the folded structure and decrease the free energy. The folding continues until the most stable structure has been achieved. If there is a tie, the algorithm picks one.

I propose that the folding process should be modeled as a function of time. Furthermore, I propose that, initially, folding occurs locally, based on two hypotheses. The first hypothesis is due to the physics at play: it is more likely for bases that are within some proximity to interact with one another than those that are far apart. The second hypothesis is due to the strength of the base interactions. The bases in a

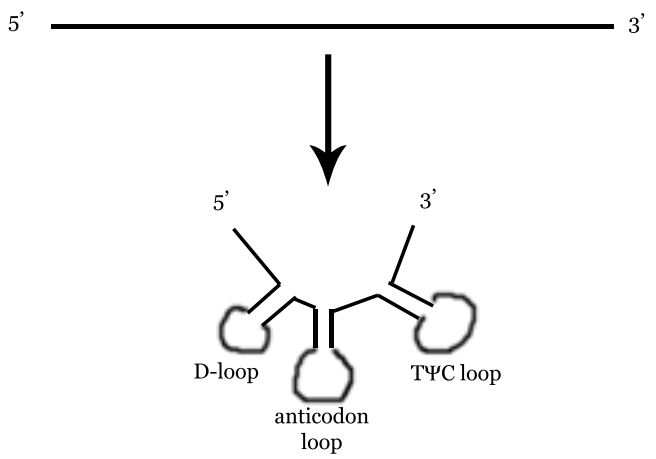

Figure 3.1: Hypothesis for preferred local interactions

tRNA molecule bind by the means of hydrogen bonding, which is not the strongest kind of attractive force. There are much stronger molecular interactions such as covalent bonding and ionic bonding. Because hydrogen bonding is the way in which bases bind, the attractive forces among the bases are relatively weak. This in turn allows only those bases that are within some neighborhood to "find" their partners. Based on these two hypotheses, I propose that, initially, local folding is favored. These local interactions result in the formations of the D, anticodon, and TψC arms.

Consequently, upon completion of the constructions of the three arms, the formation of the acceptor arm can take place. The reason behind it is built upon my first hypothesis: only bases that are within a certain proximity can interact with one another. The completion of the D, anticodon, and TψC arms "shortens" the length of the tRNA molecule as illustrated in figure 3.1 and brings the ends of the sequence within the hydrogen bonding proximity.

Thus, the folding process of a tRNA sequence can be divided into two stages:

1. formation of the D, anticodon, and TψC arms

2. formation of the acceptor arm

The subdivision of the folding process is what makes my FEM&LFEP algorithm different from the others that implemented the free energy minimization algorithm. My algorithm ensures that the canonical structures that form in the early stages of folding are not undone. I would like to note that this is not in total contradiction with Pörschke's defect diffusion concept, because my FEM&LFEP algorithm does not model such "hurried" folding of RNAs. Instead, for each possible pair of bases in the sequence, it computes all possible substructures and chooses the most stable configuration. This thorough computation eliminates the need to later "fix" the structure by shifting bases.

3.1 FEM: Dynamic Programming

Using dynamic programming, the FEM part of my algorithm computes the free energy of the intermediate structures and store the minimum *energy* value in a table E of size l-by-l, where l = (length of input tRNA sequence) - 1. Only the upper triangle of the energy table E indicated in grey in figure 3.2 is computed. The table is filled in diagonally in the direction of the arrow, also illustrated in figure 3.2, to model the folding process according to my proposal and as a function of time: each diagonal represents one time unit. The entry $E[0, l]$ will contain the free energy of the final conformation. The secondary structure is computed by back-tracing the E table from $E[0, l]$ and by implementing the LFEP part of my algorithm.

In order to compute the secondary structure of a given RNA sequence by dynamic programming, a recurrence must be defined and implemented. Moreover, the recurrence must compute and compare the free energies of all possible canonical structures for a given pair of bases i and j and store the minimum energy value in $E[i, j]$. While the exact implementations, energy functions, and energy parameters may be different, the Vienna package [17] [27] and RNAStructure 4.2 [14] [21] implement a recurrence

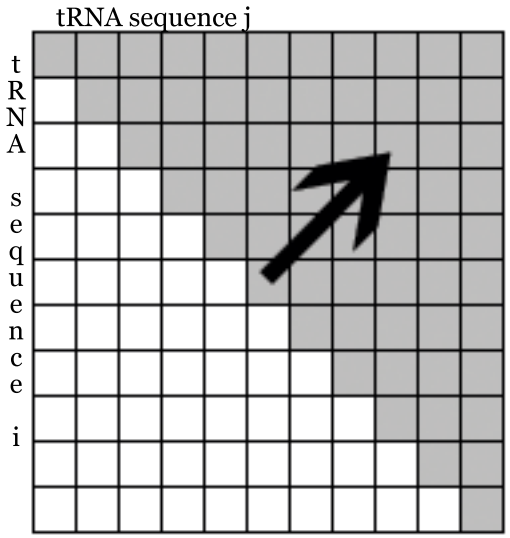

Figure 3.2: Energy table for tRNA secondary structure. The shaded region is computed in the direction of the arrow.

that performs just that.

My FEM&LFEP algorithm, on the other hand, implements a simplified recurrence. Specifically, the recurrence does not explicitly compute the free energies of bulge loops, multiloops, or internal loops of sizes bigger than 2-by-2. Instead, they are handled as variants of a dangle structure.

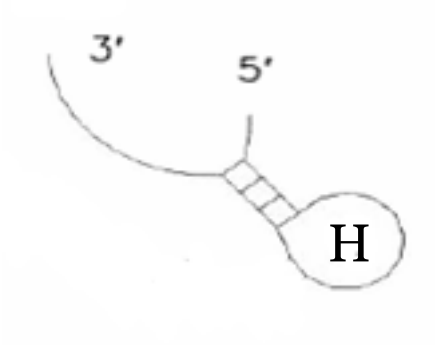

Figure 3.3: Dangle structure

A dangle structure consists of stacks and one or two unpaired sequences of bases that are not part of a hairpin loop. For example, the 3' and 5' ends in figure 3.3 are dangles: a single strand of bases growing from the stacked base pair substructure.

Expanding on this concept, I describe a bulge loop as a dangle on one side that is closed by a base pair. Similarly, a multiloop is characterized as two or more stack structures joined by dangles. Finally, an internal loop is generally defined as two dangles on both the 5' and 3' sides that are closed by a base pair. However, internal loops of sizes 1x1 and 2x2 are computed as according to Zuker's algorithm. [21] The new definitions of the elementary structures lead to the following recurrence.

$$
E[i,j] = \min \begin{cases} \text{hairpin(i,j)} & \\ E[i+r,j] + dangle(r) & \\ E[i,j-r] + dangle(r) & \\ E[i+2,j-2] + IntLoop(2) & \\ E[i+3,j-3] + IntLoop(4) & \\ E[i+1,j-1] + stack(i,j) & \\ \min \sum_{i+t<k\leq j-t-2} \sum_{k<l\leq j-t-1} & E[i,k] + E[l,j] + \\ & dangle(l-k-1) + premium \end{cases}
$$

Description of the terms:

1. t = smallest number of unpaired bases that is required for the molecule to turn. In the FEM&LFEP algorithm, $t = 3$.

2. $hairpin(i,j)$ computes the energy of a hairpin structure that forms with bases i and j of size $(j-i)$ bases given that i and j are complementary and $j-i > t$. The energy parameters for the hairpin loop are the same as those implemented by RNAStructure 4.2 [14] [21].

3. $E[i+r,j]+dangle(r)$ and $E[i,j-r]+dangle(r)$ compute a dangle structure. If bases $(i+r)$ and j are complementary and $i+r < j-t$, then $E[i+r,j]+dangle(r)$ computes a dangle structure in which bases $(i+r)$ and j are paired and unpaired bases of length r hang from the stack. Similarly, if bases i and $(j-r)$ are complementary and $i < j-r-t$, then $E[i,j-r]+dangle(r)$ computes a dangle structure in which bases i and $(j-r)$ are paired and unpaired bases of length r dangle from the structure. The free energy of the unpaired base sequence is

computed by an affine function $dangle(r) = a * r + penalty$. a and $penalty$ are constants that are determined experimentally. I ran the algorithm with about 100 different tRNA sequences and took the pair that gave the best results. In my implementation of the FEM&LFEP algorithm, $a = 33$ and $penalty = 277$.

4. $E[i+1, j-1] + stack(i, j)$ computes the energy of stacked structure. Bases $(i+1)$ and $(j-1)$ are paired and bases i and j are paired immediately on top of the $(i+1, j-1)$ pair. The stacking energy parameters are the same as those implemented by the Vienna package [17] [27].

5. $E[i+2, j-2] + IntLoop(2)$ computes the energy of a 1-by-1 internal loop structure such as (.((...)).) as implemented by RNAStructure 4.2 [14] [21].

6. $E[i+3, j-3] + IntLoop(4)$ computes the energy of a 2-by-2 internal loop structure such as ((..(((.....))).)) as implemented by RNAStructure 4.2 [14] [21].

7. $\min \sum_{i+TURN<k\leq j-TURN-2} \sum_{k<l\leq j-TURN-1} E[i,k] + E[l,j] + dangle(l-k-1) + premium$ computes the multiloop structure. Two hairpin-stack structures are joined by a sequence of $(l-k-1)$ unpaired bases. The ranges of k and l are designed to avoid pseudoknots, $j > i$, and the turn restriction is applied. A pseudoknot is defined as two pairs of bases (i, j) and (k, l) such that $i < k < j < l$.[19] The $dangle(l-k-1)$ is used to compute the free energy of the unpaired bases that join the two canonical structures to remain consistent with my definition of a multiloop. The $premium$ parameter is determined experimentally in the same fashion as a and $penalty$. It stabilizes the resulting multiloop structure. In the FEM&LFEP algorithm, $premium = -100$.

In order to implement my definitions of bulge loops, multiloops, and internal loops of sizes greater than 2-by-2, the stack energy function in the recurrence must be subdivided as follows.

$$E[i,j] = \begin{cases} E[i+1,j-1] + penalty & \text{if } (i,j) = \text{first base pair} \\ & \text{that is not a hairpin and} \\ & (i+1,j-1) \text{ is not complementary} \\ E[i+1,j-1] + stack(i,j) & \text{otherwise} \end{cases}$$

The *penalty* parameter is the same as that used in the *dangle*() function. The above function is designed to penalize the beginning of a stacking structure when it "closes" one or two dangles and the resulting structure is not a hairpin. These "closing" base pairs must be penalized, because some force is required to bring the two bases close enough to interact. In order to simplify the recurrence, the same *penalty* value is used for all such closing structures regardless of their sizes.

3.2 Lowest Free Energy Path (LFEP)

While the free energy minimization algorithm simply backtraces the E table to compute the optimal secondary structure, the FEM&LFEP algorithm performs two tasks: backtrace the energy table E to obtain the acceptor arm and compute the lowest free energy path. The lowest free energy path is the most stable combination of non-overlapping canonical structures. The LFEP algorithm conforms with my proposal, which states that the substructures that form during the early stages of folding are stable and are not likely to be undone.

The need for the LFEP algorithm arises due to the competition that occurs in the early folding phase. During the first stages of folding, many hairpin-stack structures are computed and stored in the E table. Some of these elementary structures compete with the D, anticodon, and TψC arms when E is backtraced, because they are individually as or more stable. The LFEP algorithm is based on the idea that by combining the sufficiently stable canonical structures, it may be possible to obtain a more stable overall structure than greedily selecting the most stable substructure at

any given point.

The lowest free energy path is computed as follows. During the early stages of folding, when local interactions occur, the algorithm collects segments that are comprised of the hairpin-arm structure. In order to perform this data collection, two decisions must be made: the minimum size of the stem in the hairpin-stem structure that is required and the definition of "early folding stages". To determine the two factors, several secondary structures of different tRNA sequences were examined and analyzed. The analysis showed that the D, anticodon, and TψC stems almost always consist of three or more base pairs. It should be noted that internal loop structures such as (.((...)).) and (..((...))..) were counted as stems of size three and ((.((...)).)) and ((..((...))..)) as stems of size four. The analysis also indicated that all three arms (D, anticodon, and TψC) almost always complete their formation within the computation of twenty time units or diagonals. Thus, in my implementation, three was used as the minimum stack size and "early folding stages" were defined as first twenty diagonals. Furthermore, the collected segments must lie between the regions of the 3' end bases and of the 5' end bases that form the acceptor arm. Hence, in my implementation, the algorithm ignores the first seven and the last eight bases of the input sequence. These numbers may be update after computing the lowest free energy path and before computing the acceptor arm. The update is described later in the section.

The LFEP algorithm treats collected segments as horizontal line segments in the plane. After all of the segments have been collected, they are sorted in non-decreasing order of the starting base number (the bases are always counted from the 5' end to the 3' end). Using these segments and the plane sweep concept, the LFEP algorithm computes the lowest possible free energy that can be achieved by concatenating non-overlapping segments.

The plane sweep algorithm is defined as follows. Let L be a vertical sweep line that

is initially placed to the left of all line segments or any other two-dimensional data sets such as points. Move L across to the right, keeping track of all segments or other data intersecting it [5], while computing a value of interest for each datum intersected by the sweep line and selectively storing the information of interest. Among many applications, the plane sweep has been used in Computational Geometry to compute the closest pair of points or segment intersections. In the LFEP algorithm, the plane sweep technique is applied to compute an optimal concatenation of non-overlapping segments.

Because the lowest free energy path is computed by the plane sweep method, the segments are sorted according to the starting positions, in non-decreasing order. The event points, also sorted in non-decreasing order, are the positions where either a new segment begins or an active segment ends. In figure 3.4, event points are those in parentheses. The gist of the algorithm is as follows.

Let S = {set of all segments sorted in non-decreasing order of their left extreme}

EP = {P_1, P_2, ..., P_{2n}, set of all event points

sorted in non-decreasing order; n is the number of segments}

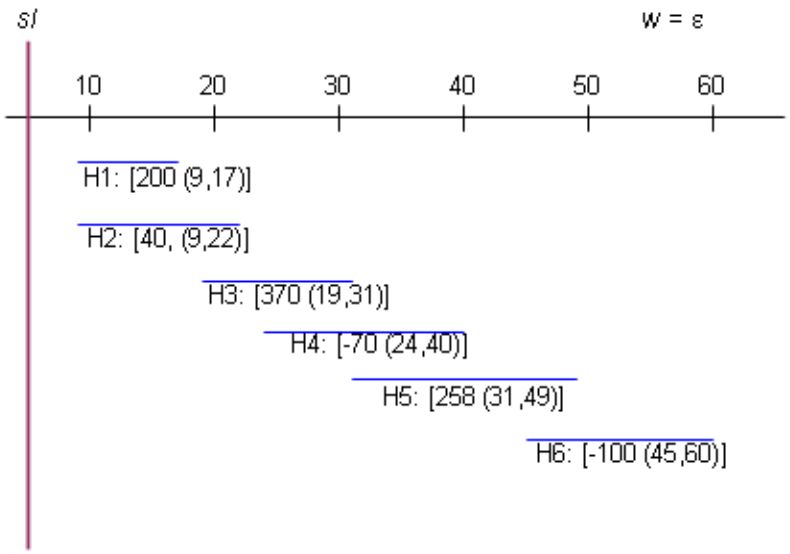

Figure 3.4: Lowest Free Energy Path Example. The blue lines are the energy segments of hairpin-stack substructures that may form during the initial stages of folding. Under each segment are (in the order listed): name: [total free energy (beginning index, ending index parentheses of the segment.

LowestFreeEnergyPath(S, EP)
    $w \leftarrow \epsilon$ (empty set of segments)
    $sl \leftarrow 1$
    while $sl \leq 2n$
        if $P_{sl} == b_s$
            if $b_s > e_w$
                then $s \leftarrow Concatenate(s, w)$
            then $A \Leftarrow s$
        if $P_{sl} == e_s$
            then remove s from A
            compare $En(s)$ and $En(w)$ and update
        $sl \leftarrow sl + 1$
    $DisplayResult(w)$

The detailed execution of the algorithm is illustrated in figures 3.6 through 3.15.

As the sweep line moves from left to right, one or two things can happen:

1. at least one active segment terminates

2. at least one new segment becomes active

An active segment is a segment that is intersected by the sweep line at an interior point, and a segment is terminated when the sweep line reaches its end point. During the sweeping, the algorithm keeps track of the following two items:

1. all active segments
2. w (a segment or a concatenation of non-overlapping segments with the lowest free energy)

The plane sweep algorithm is implemented as follows.

let sl = sweep line position

$EP[sl]$ = event point at sl

A = set of active segments

T = set of terminated segments

w = current winner, initially set to *empty* (ϵ)

b_s = beginning index of segment s

e_s = ending index of segment s

$En(s)$ = free energy of segment s

b_w = beginning index of w

e_w = ending index of w

$En(w)$ = free energy of w

The data structure used in the algorithm stores beginning and ending indices and free energy value of each segment that is being concatenated, as well as the overall beginning and ending indices and free energy value of the resulting concatenated segment.

The advancing step of the sweep corresponds to moving the sweep line from $EP[sl]$ to $EP[sl+1]$. If a segment s becomes active, determine whether s overlaps with w, that is whether $b_s \leq e_w$. If so, simply add s to A. Otherwise, concatenate s and w.

The concatenation involves an update of the free energy and is computed as follows:

$$En(s) = En(w) + (b_s - e_w - 1) * a + penalty + En(s)$$

$$b_s = b_w$$

$$e_s = e_s$$

The resulting segment is still referred to as s. While s stores complete information of both the original s and w segments in its data structure, it also stores additional information described above. At this point, A contains the updated version of A.

If a segment s terminates, determine whether w needs to be updated. If w does not yet exist, that is, if $w == \epsilon$, then s becomes w. If, on the other hand, w exists, the free energies of s and t must be computed and compared to determine whether to update w.

let p = ending index of acceptor arm on the 5' end

q = starting index of acceptor arm on the 3' end

When a segment $[En(s)\ (b_s,\ e_s)]$ terminates, the free energy of a structure such as those shown in figure 3.5(A) and figure 3.5(B) among other possible conformations is computed. It is important to reassert that s can be a single segment as in figure 3.5(A) or a concatenation of segments: one possible configuration is shown in figure 3.5(B). In any case, the structure is composed of at least two dangles, at least one stem, and at least one hairpin. Because the free energy of the structure between b_s and e_s is already computed as $En(s)$, all that is left to do is to add the dangle energies on both ends.

Specifically,

$$energy_s = penalty + (b_s - p - 1) * a + En(s) + penalty + (q - e_s - 1) * a$$

$$energy_w = penalty + (b_w - p - 1) * a + En(w) + penalty + (q - e_w - 1) * a$$

If $energy_s < energy_w$, the segment s becomes the new w. If, on the other hand, $energy_s) \geq energy_w$, s is discarded.

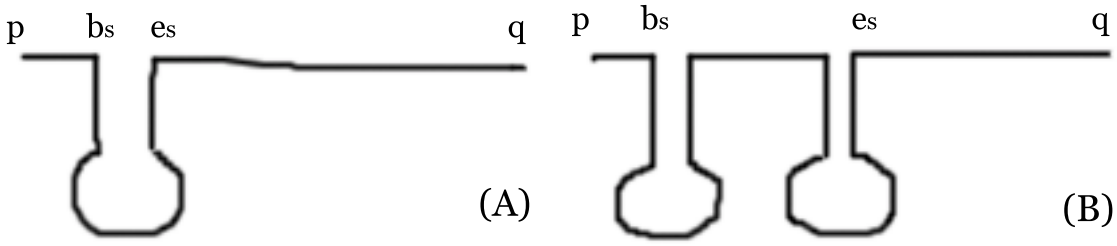

Figure 3.5: A: One possible overall structure; B: Another possible overall structure

When comparing a terminating segment s to w, the free energy of the structure between p and q for both s and w must be used in the comparison. Otherwise, the two structures are not compared under the same conditions. One of the reasons for this requirement is because the number of bases that comprises s may not be the same as that of w. The implementation of the LFEP algorithm employs the *dangle* function to compute the energies of the dangle structures on the 5' end and the 3' end as well as those of any dangle structures joining the canonical structures in the concatenation. However, the values of a and *penalty* differ from those used in the implementation of the FEM algorithm as they are determined experimentally by running the LFEP algorithm on a training set, which were $a = 33$ and $penalty = 277$. The training set contained the first fifteen sequences of each amino acid listed at [16]. $a = 55$ and $penalty = 227$ were determined to be the optimal set of parameters. It is interesting to note that the value of the *penalty* parameter is lower in the LFEP than the FEM algorithm by 50 while the value of the a parameter is higher in the LFEP than the FEM algorithm by 22.

To summarize, the LFEP algorithm is implemented as follows. The variables are as previously defined.

```
LowestFreeEnergyPath(S, EP)
    /* Initialize variables */
    index ← 1
    sl ← 1
    A ← Ø
    s ← S[index]
    w ← ϵ
    while sl ≤ 2n
        /* Compute all active segments */
        while b_s == EP[sl]
            if e_w < b_s
                then s ← Concatenate(s, w)
            A ⇐ s
            index ← index + 1
            s ← S[index]
        /* Compute all newly terminated segments */
        T ← Ø
        for i ← size(A) to 1
            s_t ← A[i]
            if e_{s_t} ≤ P_sl
                then T ⇐ s_t
                remove s_t from A
        /* Compare w with terminated segments and update as necessary */
        w' ← w
        for i ← 1 to size(T)
            s1 ← T[i]
            E1 ← penalty + (b_{s1} − p − 1) * a + En(s1) + penalty + (q − e_{s1} − 1) * a
            E' ← penalty + (b_{w'} − p − 1) * a + En(w') + penalty + (q − e_{w'} − 1) * a
            if E1 < E'
                then w' ← s1
        w ← w'
        /* Move sweep line */
        while P_sl == P_{sl+1}
            sl ← sl + 1
    DisplayResult(w)

Concatenate(s, w)
    segments ⇐ s
    segments ⇐ w
    structure(Segs) ← segments
    En(Segs) ← En(s) + En(w) + penalty + (b_s − e_w − 1) * a
    b_Segs ← b_w
    e_Segs ← e_s
    return Segs
```

DisplayResult(w)
    $segments \leftarrow structure(w)$
    for $i \leftarrow 1$ to $size(segments)$
        $s \leftarrow segments_i$
        print($En(s)$, $b_{segments_i}$, $e_{segments_i}$)

After computing w, p and q are updated as:
    $p \leftarrow b_w - 1$
    $q \leftarrow e_w$
The update ensures more accurate computation of the acceptor arm.

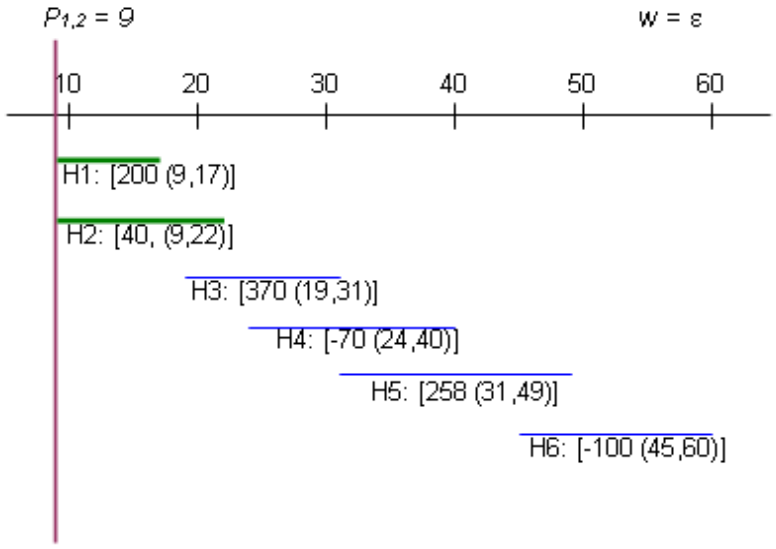

Figure 3.6: Sweep line at 9. Active segments: H1, H2; $w = \epsilon$
The active segments are displayed in green, w and newly terminated segments are displayed in red, joins are displayed as blue arrows, and discarded segments are displayed in black. When w is computed and is composed of only one segment s, it will be displayed in the following format: total free energy between p and q {segment name: [$En(s)$ (b_s, e_s)]}. If, on the other hand, w is a concatenation of segments, i.e., s_1 and s_2 where $b_{s_2} > e_{s_1}$, it will be displayed in the following format: total free energy between p and q {segment name of s_2: [$En(s_1)$ (b_{s_1}, e_{s_1}), $En(s_2)$ (b_{s_2}, e_{s_2})]}.

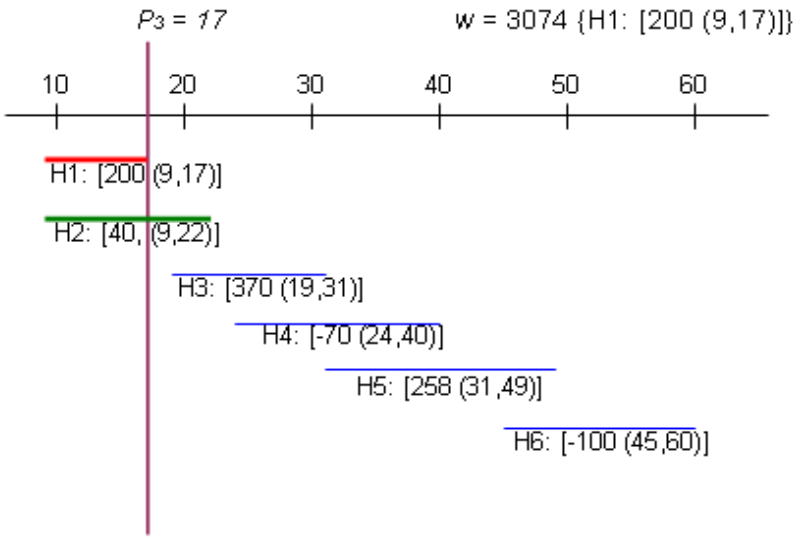

Figure 3.7: Sweep line at 17. Active segments: H2; Terminated segments: H1; w is computed as: $En(H1) = penalty + (9 - p - 1) * a + 200 + penalty + (q - 17 - 1) * a = 3074$. $w = 3074\{H1 : [200(9, 17)]\}$

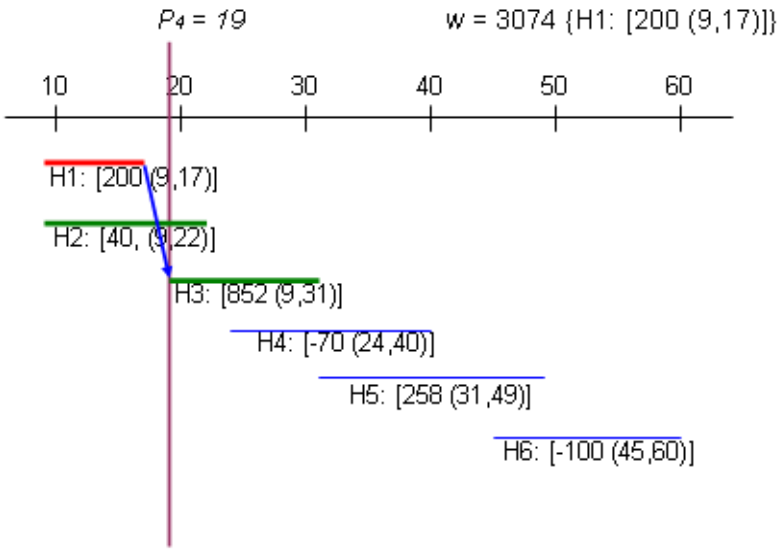

Figure 3.8: Sweep line at 19. Active segments: H2, H3; $w = 3074\{H1 : [200(9, 17)]\}$. Because $b_{H3} > e_w$, H3 is concatenated with w as follows: $En(H3) = En(H3) + En(w) + penalty + (b_{H3} - e_w - 1) * a$. The starting index is updated as follows: $b_{H3} = b_w$. H3 is a concatenation of segments. Therefore, its data structure is able to store the individual segment information, i.e., H3 and w as well as the overall information of the concatenation.

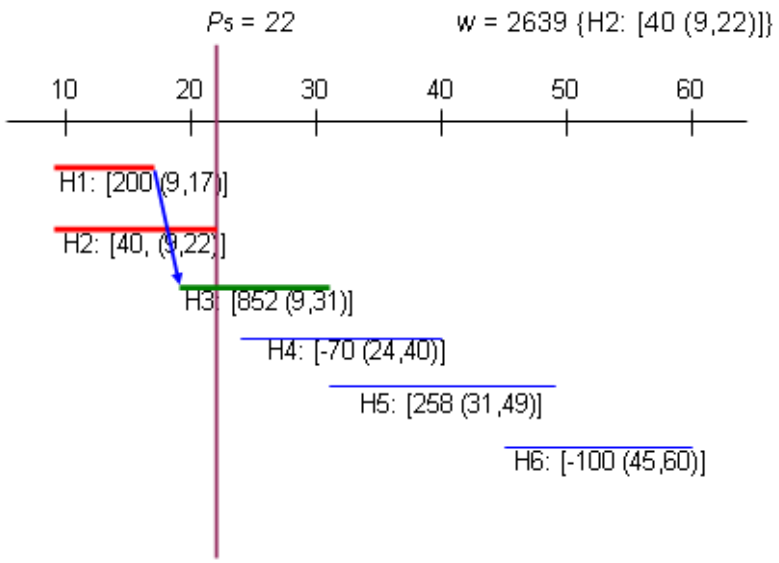

Figure 3.9: Sweep line at 22. Active segments: H3; Terminated segments: H2; Because $En(H2) < En(w)$, w is updated as $w = 2639\{H2 : [40(9, 22)]\}$

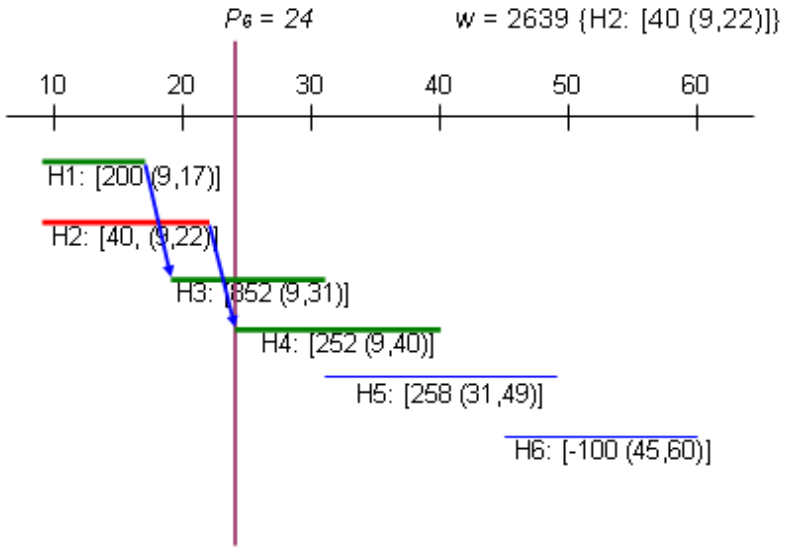

Figure 3.10: Sweep line at 24. Active segments: H3, H4; $w = 3074\{H2 : [200, (9, 17)]\}$ H4 is concatenated with w as described previously.

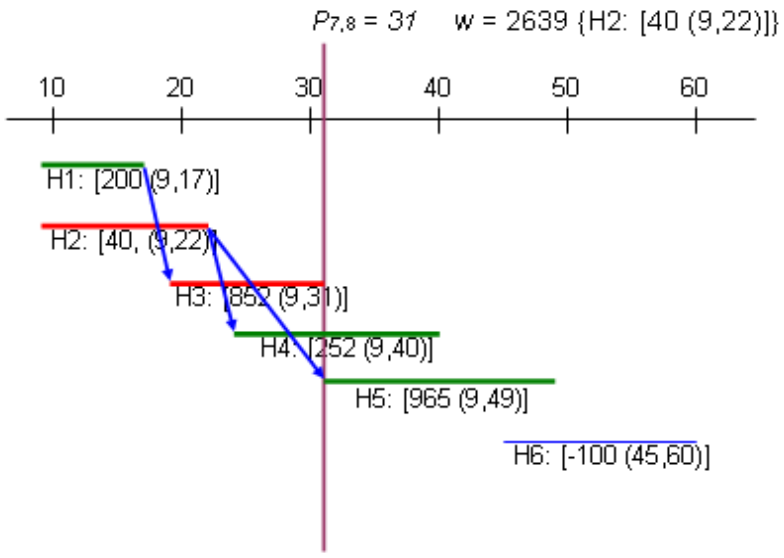

Figure 3.11: Sweep line at 31. Active segments: H4, H5; Terminated segments: H3. $w = 3074\{H2 : [200, (9, 17)]\}$ Because $En(H3) \geq En(w)$, H3 is discarded.

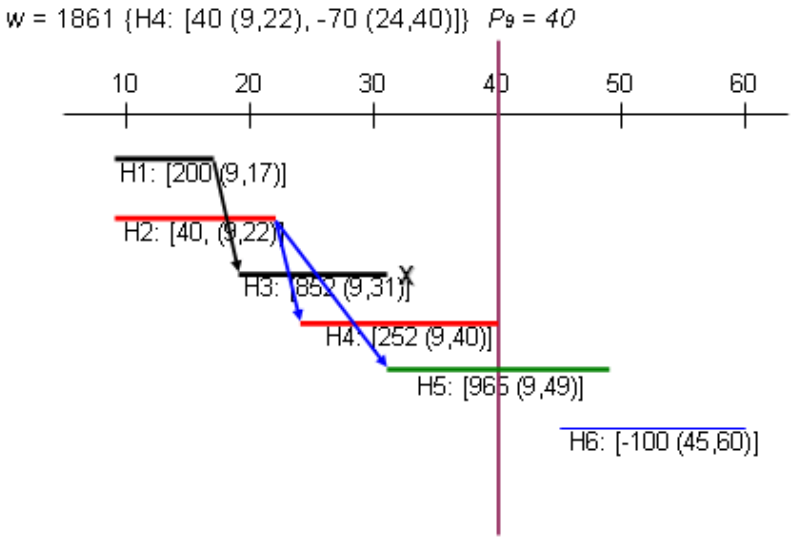

Figure 3.12: Sweep line at 40. Active segments: H5; Terminated segments: H4; Because $En(H4) < En(w)$, w is updated as $w = 1861\{H4 : [40(9, 22), -70(24, 40)]\}$

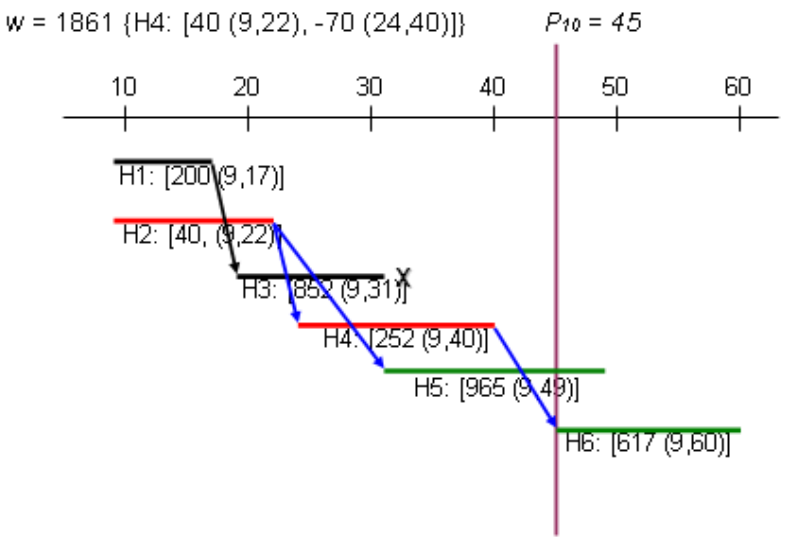

Figure 3.13: Sweep line at 45. Active segments: H5, H6;
$w = 1861\{H4 : [40(9, 22), -70(24, 40)]\}$

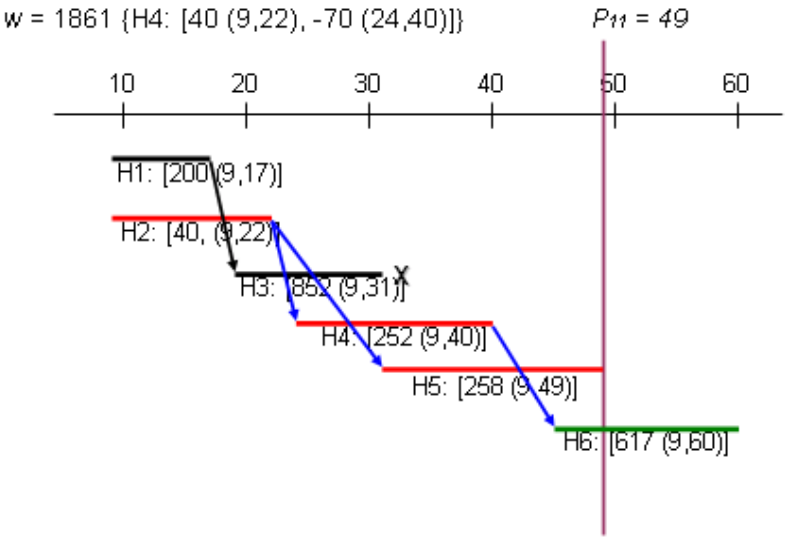

Figure 3.14: Sweep line at 49. Active segments: H6; Terminated segments: H5; Because $En(H5) \geq En(w)$, H5 is discarded. $w = 1861\{H4 : [40(9, 22), -70(24, 40)]\}$

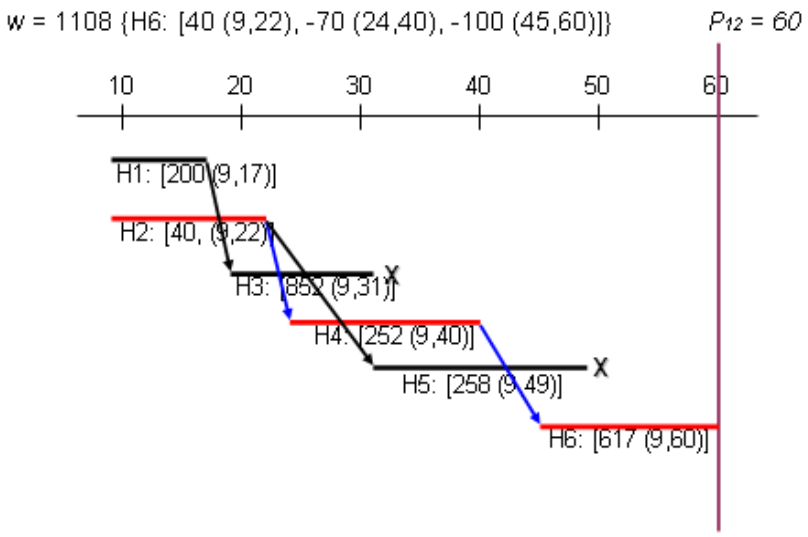

Figure 3.15: Sweep line at 60. Active segments: *empty*; Terminated segments: H6; Because $En(H6) < En(w)$, w is updated as
$w = 1108\{H6 : [40(9, 22), -70(24, 40), -100(45, 60)]\}$

Chapter 4

Results

I ran the FEM&LFEP algorithm on a test set that was composed of fifteen different organisms: bovine, chicken, chimpanzee, drosophila, ecoli, house mouse, fat dormouse, mouse, platypus, hamadryas baboon, orangutan, pygmy chimpanzee, rat, rhinoceros, and wild boar. Twenty or more different tRNA sequences have been published for each of these organisms by [16] and [11]. I compared the results from FEM&LFEP to those from RNAStructure 4.2 [14] [21]. They are summarized in the tables that follow. Note that I chose not to compare the results to those from the Vienna package [17] [27], because RNAStructure 4.2 [14] [21] implements the updated version of the algorithm that is implemented by the Vienna Package [17] [27] and the performance of RNAStructure 4.2 [14] [21] was significantly better than the Vienna Package [17] [27] when I compared the two programs using several different sequences.

Following two tables contain the results for the tRNA sequences of bovine obtained from [11]. These numbers were calculated by first computing the secondary structures by both FEM&LFEP and RNAStructure 4.2 [14] [21] and comparing the results to the published structures. Specifically, for each sequence, Table 4.1 computes the percentage of correct base pairs computed by each algorithm in relation to the published structures, and Table 4.2 calculates the percentage of base pairs computed by each algorithm in relation to the total number of base pairs computed by each algorithm.

BOVINE		
	FEM&LFEP	RNAStructure 4.2 [14] [21]
aa	$\frac{\#correctBPs}{\#BPsintRNA}$	$\frac{\#correctBPs}{\#BPsintRNA}$
ala	90.91%	90.91%
arg	100.00%	85.00%
asn	55.00%	25.00%
asp	95.00%	35.00%
cys	94.74%	100.00%
glu	100.00%	95.45%
gln	76.19%	47.62%
gly	95.45%	86.36%
his	90.00%	95.00%
ile	94.74%	94.74%
leu1	90.48%	35.00%
leu2	90.00%	100.00%
lys	100.00%	100.00%
phe	89.47%	100.00%
pro	100.00%	95.24%
ser1	71.43%	80.95%
ser2	90.48%	90.48%
thr	44.44%	22.22%
trp	94.74%	78.95%
tyr	90.00%	95.00%
val	57.89%	73.68%
$\geq 90\%$	71.43%	52.38%
$\geq 85\%$	76.19%	66.67%

Table 4.1: Comparison of FEM&LFEP and RNAStructure 4.2 [14] [21]. The results were obtained for the tRNA sequences of bovine.[11] The table computes the percentage of correct base pairs computed by each algorithm in relation to the correct tRNA secondary structures.

BOVINE		
	FEM&LFEP	RNAStructure 4.2 [14] [21]
aa	$\frac{\#correctBPs}{\#BPscomputed}$	$\frac{\#correctBPs}{\#BPscomputed}$
ala	90.91%	95.24%
arg	100.00%	85.00%
asn	52.38%	20.00%
asp	90.48%	35.00%
cys	90.00%	90.48%
glu	100.00%	100.00%
gln	66.67%	43.48%
gly	95.45%	90.48%
his	90.00%	90.48%
ile	90.00%	90.00%
leu1	86.36%	38.89%
leu2	81.82%	86.96%
lys	95.24%	100.00%
phe	100.00%	100.00%
pro	91.30%	90.91%
ser1	62.50%	77.27%
ser2	100.00%	82.61%
thr	50.00%	23.53%
trp	94.74%	78.95%
tyr	100.00%	100.00%
val	55.00%	70.00%
$\geq 90\%$	66.67%	47.62%
$\geq 85\%$	71.43%	57.14%

Table 4.2: Comparison of FEM&LFEP and RNAStructure 4.2 [14] [21]. The results were obtained from using the tRNA sequences of bovine.[11] The table computes the percentage of correct base pairs computed by each algorithm in relation to all base pairs computed by the algorithm.

As shown in the last two rows of the Table 4.1 and Table 4.2, FEM&LFEP outperforms RNAStructure 4.2 [14] [21] in computing the cloverleaf secondary structure of bovine's tRNA sequences. To provide a more concrete idea on what these numbers mean, following is some more information about the secondary structure of tRNAs. A published tRNA secondary structure contains usually 21 base pairs. In order for a computed secondary structure to be $\geq 90\%$ correct, the computed structure must contain at least 19 correct base pairs. Therefore, while majority of the $\geq 90\%$ structures are not quite completely correct, they most likely only differ by at most 2 base pairs.

Results for the rest of the test set are included in the following pages. The results show that FEM&LFEP produces better results than RNAStructure 4.2 [14] [21] for all of the organisms in the test set except house mouse and wild boar, for which the two algorithms perform comparably.

CHICKEN		
	FEM&LFEP	RNAStructure 4.2 [14] [21]
aa	$\frac{\#correctBPs}{\#BPsintRNA}$	$\frac{\#correctBPs}{\#BPsintRNA}$
ala1	95.00%	100.00%
ala2	55.00%	60.00%
ala3	71.43%	57.14%
arg	95.24%	95.24%
asn	100.00%	90.48%
asp1	84.21%	94.74%
asp2	100.00%	76.19%
cys	95.45%	72.73%
glu	100.00%	100.00%
gln	100.00%	54.55%
gly	66.67%	83.33%
his	90.00%	95.00%
ile	63.16%	94.74%
leu1	85.71%	33.33%
leu2	63.16%	78.95%
lys1	47.37%	57.89%
lys2	71.43%	76.19%
met	84.21%	100.00%
phe1	84.21%	94.74%
phe2	66.67%	57.14%
pro1	95.24%	95.24%
pro2	95.24%	80.95%
pro3	95.24%	80.95%
ser1	71.43%	71.43%
ser2	22.22%	94.44%
thr	100.00%	95.00%
trp1	95.00%	60.00%
trp2	95.24%	95.24%
tyr	63.16%	57.89%
val	89.47%	89.47%
$\geq 90\%$	46.67%	43.33%
$\geq 85\%$	53.33%	46.67%

Table 4.3: Comparison of FEM&LFEP and RNAStructure 4.2 [14] [21]. The results were obtained for the tRNA sequences of chicken.[11] The table computes the percentage of correct base pairs computed by each algorithm in relation to the correct tRNA secondary structures.

CHICKEN		
	FEM&LFEP	RNAStructure 4.2 [14] [21]
aa	$\frac{\#correctBPs}{\#BPscomputed}$	$\frac{\#correctBPs}{\#BPscomputed}$
ala1	90.48%	100.00%
ala2	55.00%	57.14%
ala3	71.43%	46.15%
arg	100.00%	100.00%
asn	91.30%	86.36%
asp1	80.00%	90.00%
asp2	100.00%	76.19%
cys	87.50%	66.67%
glu	91.30%	91.30%
gln	100.00%	48.00%
gly	63.16%	100.00%
his	100.00%	100.00%
ile	85.71%	94.74%
leu1	90.00%	33.33%
leu2	85.71%	65.22%
lys1	50.00%	45.83%
lys2	71.43%	72.73%
met	100.00%	100.00%
phe1	80.00%	100.00%
phe2	70.00%	50.00%
pro1	86.96%	90.91%
pro2	90.91%	70.83%
pro3	90.91%	70.83%
ser1	62.50%	62.50%
ser2	30.77%	89.47%
thr	100.00%	100.00%
trp1	95.00%	57.14%
trp2	95.24%	100.00%
tyr	52.17%	55.00%
val	100.00%	89.47%
$\geq 90\%$	50.00%	40.00%
$\geq 85\%$	63.33%	50.00%

Table 4.4: Comparison of FEM&LFEP and RNAStructure 4.2 [14] [21]. The results were obtained for the tRNA sequences of chicken.[11] The table computes the percentage of correct base pairs computed by each algorithm in relation to all base pairs computed by the algorithm.

CHIMPANZEE		
	FEM&LFEP	RNAStructure 4.2 [14] [21]
aa	$\frac{\#correctBPs}{\#BPsintRNA}$	$\frac{\#correctBPs}{\#BPsintRNA}$
ala	90.48%	90.48%
arg	100.00%	25.00%
asn	100.00%	47.62%
asp	95.00%	95.00%
cys	95.00%	100.00%
glu1	75.00%	50.00%
glu2	86.36%	27.27%
gln	71.43%	47.62%
gly	71.43%	90.48%
his	70.00%	95.00%
ile	100.00%	78.95%
leu1	95.00%	55.00%
leu2	100.00%	55.00%
lys	76.19%	66.67%
met	66.67%	83.33%
phe	63.16%	78.95%
pro	100.00%	57.14%
ser1	71.43%	76.19%
ser2	47.62%	66.67%
thr	84.21%	89.47%
trp	94.74%	94.74%
tyr	90.00%	95.00%
val	57.89%	78.95%
$\geq 90\%$	47.83%	30.43%
$\geq 85\%$	52.17%	34.78%

Table 4.5: Comparison of FEM&LFEP and RNAStructure 4.2 [14] [21]. The results were obtained for the tRNA sequences of chimpanzee.[11] The table computes the percentage of correct base pairs computed by each algorithm in relation to the correct tRNA secondary structures.

CHIMPANZEE		
	FEM&LFEP	RNAStructure 4.2 [14] [21]
aa	$\frac{\#correctBPs}{\#BPscomputed}$	$\frac{\#correctBPs}{\#BPscomputed}$
ala	90.48%	95.00%
arg	100.00%	26.32%
asn	100.00%	45.45%
asp	90.48%	90.48%
cys	90.48%	90.91%
glu1	65.22%	43.48%
glu2	100.00%	25.00%
gln	65.22%	43.48%
gly	83.33%	95.00%
his	70.00%	90.48%
ile	90.48%	75.00%
leu1	100.00%	50.00%
leu2	100.00%	61.11%
lys	76.19%	60.87%
met	80.00%	83.33%
phe	70.59%	75.00%
pro	100.00%	66.67%
ser1	62.50%	69.57%
ser2	41.67%	63.64%
thr	100.00%	100.00%
trp	94.74%	100.00%
tyr	94.74%	95.00%
val	78.57%	78.95%
$\geq 90\%$	56.52%	34.78%
$\geq 85\%$	56.52%	34.78%

Table 4.6: Comparison of FEM&LFEP and RNAStructure 4.2 [14] [21]. The results were obtained for the tRNA sequences of chimpanzee.[11] The table computes the percentage of correct base pairs computed by each algorithm in relation to all base pairs computed by the algorithm.

DROSOPHILA		
	FEM&LFEP	RNAStructure 4.2 [14] [21]
aa	$\frac{\#correctBPs}{\#BPsintRNA}$	$\frac{\#correctBPs}{\#BPsintRNA}$
ala	95.24%	57.14%
asp	90.48%	66.67%
glu1	100.00%	95.24%
glu2	100.00%	95.24%
glu3	52.38%	33.33%
phe	76.19%	57.14%
gly1	54.55%	68.18%
gly2	52.38%	66.67%
his	71.43%	95.24%
ile	75.00%	80.00%
lys1	71.43%	76.19%
lys2	76.19%	66.67%
leu1	90.00%	85.00%
leu2	95.24%	57.14%
met	70.00%	35.00%
asn	95.24%	57.14%
pro	95.24%	76.19%
arg1	60.00%	55.00%
arg2	95.24%	95.24%
arg3	95.24%	95.24%
ser1	90.48%	85.71%
ser2	61.90%	85.71%
thr	95.00%	100.00%
val1	55.00%	25.00%
val2	95.00%	85.00%
tyr	95.24%	95.24%
$\geq 90\%$	53.85%	26.92%
$\geq 85\%$	53.85%	42.31%

Table 4.7: Comparison of FEM&LFEP and RNAStructure 4.2 [14] [21]. The results were obtained for the tRNA sequences of drosophila. [11] The table computes the percentage of correct base pairs computed by each algorithm compared to the correct structure.

DROSOPHILA		
	FEM&LFEP	RNAStructure 4.2 [14] [21]
aa	$\frac{\#correctBPs}{\#BPscomputed}$	$\frac{\#correctBPs}{\#BPscomputed}$
ala	100.00%	50.00%
asp	100.00%	60.87%
glu1	95.45%	95.24%
glu2	95.45%	95.24%
glu3	44.00%	31.82%
phe	80.00%	50.00%
gly1	54.55%	62.50%
gly2	52.38%	60.87%
his	93.75%	100.00%
ile	71.43%	72.73%
lys1	71.43%	72.73%
lys2	69.57%	60.87%
leu1	100.00%	68.00%
leu2	100.00%	41.38%
met	60.87%	31.82%
asn	95.24%	48.00%
pro	90.91%	66.67%
arg1	57.14%	52.38%
arg2	100.00%	100.00%
arg3	100.00%	100.00%
ser1	100.00%	69.23%
ser2	81.25%	69.23%
thr	100.00%	100.00%
val1	50.00%	21.74%
val2	82.61%	85.00%
tyr	100.00%	100.00%
$\geq 90\%$	53.85%	26.92%
$\geq 85\%$	53.85%	30.77%

Table 4.8: Comparison of FEM&LFEP and RNAStructure 4.2 [14] [21]. The results were obtained for the tRNA sequences of drosophila.[11] The table computes the percentage of correct base pairs computed by each algorithm in relation to all base pairs computed by the algorithm.

ECOLI					
	FEM&LFEP	RNAStructure 4.2 [14] [21]		FEM&LFEP	RNAStructure 4.2 [14] [21]
aa	$\frac{\#correctBPs}{\#BPsintRNA}$	$\frac{\#correctBPs}{\#BPsintRNA}$	aa	$\frac{\#correctBPs}{\#BPsintRNA}$	$\frac{\#correctBPs}{\#BPsintRNA}$
ala1	66.67%	95.24%	pro3	45.00%	95.00%
ala2	66.67%	95.24%	gln1	95.00%	100.00%
cys	100.00%	75.00%	gln2	95.00%	100.00%
asp	23.81%	71.43%	arg1	45.00%	100.00%
glu	100.00%	90.91%	arg2	100.00%	95.24%
phe	95.24%	95.24%	arg4	90.48%	95.24%
gly1	95.00%	75.00%	arg5	95.24%	95.24%
gly2	95.24%	71.43%	ser1	65.00%	60.00%
gly3	95.24%	95.24%	ser2	95.00%	100.00%
his	95.24%	42.86%	ser4	45.00%	85.00%
ile1	85.71%	33.33%	ser5	61.90%	57.14%
ile2	100.00%	72.73%	thr1	71.43%	100.00%
lys	100.00%	77.27%	thr2	95.24%	95.24%
leu1	85.00%	0.00%	thr4	70.00%	75.00%
leu2	90.00%	100.00%	thr5	95.24%	100.00%
leu3	95.00%	95.00%	val1	76.19%	28.57%
leu4	66.67%	95.24%	val2	71.43%	76.19%
leu5	95.00%	95.00%	val3	66.67%	76.19%
met	75.00%	50.00%	trp	90.48%	100.00%
asn	95.24%	52.38%	tyr1	95.00%	100.00%
pro1	95.24%	95.24%	tyr2	95.00%	100.00%
pro2	100.00%	95.24%			
$\geq 90\%$	51.16%	41.86%			
$\geq 85\%$	62.79%	48.84%			

Table 4.9: Comparison of FEM&LFEP and RNAStructure 4.2 [14] [21]. The results were obtained for the tRNA sequences of Ecoli.[11] The table computes the percentage of correct base pairs computed by each algorithm compared to the correct structure.

ECOLI					
	FEM&LFEP	RNAStructure 4.2 [14] [21]		FEM&LFEP	RNAStructure 4.2 [14] [21]
aa	$\frac{\#correctBPs}{\#BPscomputed}$	$\frac{\#correctBPs}{\#BPscomputed}$	aa	$\frac{\#correctBPs}{\#BPscomputed}$	$\frac{\#correctBPs}{\#BPscomputed}$
ala1	82.35%	100.00%	pro3	52.94%	90.48%
ala2	73.68%	90.91%	gln1	95.00%	100.00%
cys	83.33%	62.50%	gln2	95.00%	100.00%
asp	27.78%	65.22%	arg1	64.29%	100.00%
glu	100.00%	100.00%	arg2	87.50%	100.00%
phe	90.91%	90.91%	arg4	100.00%	100.00%
gly1	100.00%	88.24%	arg5	100.00%	100.00%
gly2	90.91%	62.50%	ser1	76.47%	42.86%
gly3	90.91%	95.24%	ser2	86.36%	80.00%
his	95.24%	42.86%	ser4	40.91%	58.62%
ile1	85.71%	25.93%	ser5	61.90%	44.44%
ile2	100.00%	64.00%	thr1	88.24%	91.30%
lys	100.00%	73.91%	thr2	90.91%	90.91%
leu1	89.47%	0.00%	thr4	63.64%	68.18%
leu2	100.00%	71.43%	thr5	90.91%	91.30%
leu3	90.48%	82.61%	val1	69.57%	26.09%
leu4	100.00%	80.00%	val2	65.22%	80.00%
leu5	90.48%	76.00%	val3	66.67%	72.73%
met	78.95%	47.62%	trp	90.48%	100.00%
asn	100.00%	52.38%	tyr1	95.00%	86.96%
pro1	83.33%	90.91%	tyr2	95.00%	86.96%
pro2	84.00%	90.91%			
$\geq 90\%$	51.16%	41.86%			
$\geq 85\%$	62.79%	48.84%			

Table 4.10: Comparison of FEM&LFEP and RNAStructure 4.2 [14] [21]. The results were obtained for the tRNA sequences of Ecoli.[11] The table computes the percentage of correct base pairs computed by each algorithm in relation to all base pairs computed by the algorithm.

HOUSE MOUSE		
	FEM&LFEP	RNAStructure 4.2 [14] [21]
aa	$\frac{\#correctBPs}{\#BPsintRNA}$	$\frac{\#correctBPs}{\#BPsintRNA}$
ala	95.00%	95.00%
arg	85.71%	66.67%
asn	95.00%	95.00%
asp	95.24%	95.24%
cys	95.24%	100.00%
glu	95.24%	76.19%
gln	100.00%	95.24%
gly	94.74%	94.74%
his	45.00%	95.00%
ile	90.00%	95.00%
leu1	95.00%	95.00%
leu2	80.00%	75.00%
lys	73.68%	100.00%
met	83.33%	66.67%
phe	90.00%	100.00%
pro	85.71%	76.19%
ser	76.19%	76.19%
thr	94.74%	94.74%
trp	83.33%	77.78%
tyr	90.00%	95.00%
val	95.00%	25.00%
$\geq 90\%$	61.90%	61.90%
$\geq 85\%$	71.43%	61.90%

Table 4.11: Comparison of FEM&LFEP and RNAStructure 4.2 [14] [21]. The results were obtained for the tRNA sequences of house mouse.[16] The table computes the percentage of correct base pairs computed by each algorithm compared to the correct structure.

HOUSE MOUSE		
	FEM&LFEP	RNAStructure 4.2 [14] [21]
aa	$\frac{\#correctBPs}{\#BPscomputed}$	$\frac{\#correctBPs}{\#BPscomputed}$
ala	90.48%	90.48%
arg	94.74%	77.78%
asn	95.00%	95.00%
asp	86.96%	86.96%
cys	83.33%	87.50%
glu	90.91%	66.67%
gln	91.30%	86.96%
gly	100.00%	100.00%
his	50.00%	100.00%
ile	90.00%	90.48%
leu1	90.48%	90.48%
leu2	84.21%	68.18%
lys	77.78%	100.00%
met	100.00%	54.55%
phe	100.00%	100.00%
pro	90.00%	84.21%
ser	64.00%	66.67%
thr	94.74%	100.00%
trp	83.33%	87.50%
tyr	100.00%	100.00%
val	100.00%	27.78%
$\geq 90\%$	66.67%	47.62%
$\geq 85\%$	71.43%	66.67%

Table 4.12: Comparison of FEM&LFEP and RNAStructure 4.2 [14] [21]. The results were obtained for the tRNA sequences of house mouse.[16] The table computes the percentage of correct base pairs computed by each algorithm in relation to all base pairs computed by the algorithm.

FAT DORMOUSE		
	FEM&LFEP	RNAStructure 4.2 [14] [21]
aa	$\frac{\#correctBPs}{\#BPsintRNA}$	$\frac{\#correctBPs}{\#BPsintRNA}$
ala	95.24%	90.48%
arg	100.00%	80.95%
asn	60.00%	50.00%
asp	95.24%	95.24%
cys	95.00%	100.00%
glu	100.00%	33.33%
gln	73.68%	78.95%
gly	71.43%	66.67%
his	80.95%	95.24%
ile	95.00%	95.00%
leu1	90.00%	100.00%
leu2	80.00%	55.00%
lys	78.95%	73.68%
met	83.33%	61.11%
phe	94.44%	0.00%
pro	95.24%	95.24%
ser	68.18%	54.55%
thr	80.00%	85.00%
trp	55.00%	55.00%
tyr	100.00%	95.00%
val	100.00%	76.19%
$\geq 90\%$	52.38%	38.10%
$\geq 85\%$	52.38%	42.86%

Table 4.13: Comparison of FEM&LFEP and RNAStructure 4.2 [14] [21]. The results were obtained for the tRNA sequences of fat dormouse.[16] The table computes the percentage of correct base pairs computed by each algorithm compared to the correct structure.

FAT DORMOUSE		
	FEM&LFEP	RNAStructure 4.2 [14] [21]
aa	$\frac{\#correctBPs}{\#BPscomputed}$	$\frac{\#correctBPs}{\#BPscomputed}$
ala	90.91%	82.61%
arg	100.00%	85.00%
asn	60.00%	55.56%
asp	90.91%	90.91%
cys	79.17%	83.33%
glu	95.45%	30.43%
gln	66.67%	75.00%
gly	65.22%	58.33%
his	85.00%	90.91%
ile	82.61%	82.61%
leu1	81.82%	90.91%
leu2	80.00%	68.75%
lys	62.50%	63.64%
met	100.00%	52.38%
phe	94.44%	0.00%
pro	90.91%	90.91%
ser	62.50%	60.00%
thr	76.19%	73.91%
trp	55.00%	55.00%
tyr	95.24%	95.00%
val	95.45%	80.00%
$\geq 90\%$	42.86%	23.81%
$\geq 85\%$	47.62%	28.57%

Table 4.14: Comparison of FEM&LFEP and RNAStructure 4.2 [14] [21]. The results were obtained for the tRNA sequences of fat dormouse.[16] The table computes the percentage of correct base pairs computed by each algorithm in relation to all base pairs computed by the algorithm.

MOUSE		
	FEM&LFEP	RNAStructure 4.2 [14] [21]
aa	$\frac{\#correctBPs}{\#BPsintRNA}$	$\frac{\#correctBPs}{\#BPsintRNA}$
ala1	95.00%	95.00%
ala2	76.19%	76.19%
arg	81.82%	68.18%
asn	95.24%	95.24%
asp1	95.24%	95.24%
asp2	100.00%	76.19%
cys1	95.24%	100.00%
cys2	42.86%	76.19%
cys3	0.00%	0.00%
glu1	95.45%	77.27%
glu2	100.00%	19.05%
gln	100.00%	57.14%
gly1	94.74%	94.74%
gly2	85.00%	85.00%
his1	45.00%	95.00%
his2	57.14%	52.38%
ile1	90.00%	95.00%
ile2	95.00%	95.00%
leu1	95.00%	95.00%
leu2	76.19%	52.38%
leu3	85.71%	57.14%
lys1	73.68%	100.00%
lys2	76.19%	100.00%
lys3	71.43%	76.19%
phe	89.47%	100.00%
pro1	81.82%	77.27%
pro2	95.24%	80.95%
pro3	95.24%	80.95%
ser	76.19%	76.19%
thr	89.47%	89.47%
trp	83.33%	77.78%
tyr	90.00%	95.00%
val	94.74%	21.05%
$\geq 90\%$	48.48%	39.39%
$\geq 85\%$	60.61%	45.45%

Table 4.15: Comparison of FEM&LFEP and RNAStructure 4.2 [14] [21]. The results were obtained for the tRNA sequences of mouse.[11] The table computes the percentage of correct base pairs computed by each algorithm compared to the correct structure.

MOUSE		
	FEM&LFEP	RNAStructure 4.2 [14] [21]
aa	$\frac{\#correctBPs}{\#BPscomputed}$	$\frac{\#correctBPs}{\#BPscomputed}$
ala1	90.48%	90.48%
ala2	72.73%	76.19%
arg	94.74%	83.33%
asn	100.00%	100.00%
asp1	86.96%	86.96%
asp2	100.00%	76.19%
cys1	83.33%	87.50%
cys2	56.25%	80.00%
cys3	0.00%	0.00%
glu1	95.45%	70.83%
glu2	100.00%	19.05%
gln	100.00%	52.17%
gly1	100.00%	100.00%
gly2	85.00%	80.95%
his1	50.00%	100.00%
his2	54.55%	52.38%
ile1	90.00%	90.48%
ile2	90.48%	100.00%
leu1	90.48%	90.48%
leu2	84.21%	50.00%
leu3	100.00%	44.44%
lys1	77.78%	100.00%
lys2	66.67%	100.00%
lys3	71.43%	72.73%
phe	94.44%	100.00%
pro1	94.74%	89.47%
pro2	90.91%	70.83%
pro3	90.91%	70.83%
ser	64.00%	66.67%
thr	94.44%	94.44%
trp	83.33%	87.50%
tyr	100.00%	100.00%
val	94.74%	22.22%
$\geq 90\%$	57.58%	36.36%
$\geq 85\%$	63.64%	48.48%

Table 4.16: Comparison of FEM&LFEP and RNAStructure 4.2 [14] [21]. The results were obtained for the tRNA sequences of mouse.[11] The table computes the percentage of correct base pairs computed by each algorithm in relation to all base pairs computed by the algorithm.

PLATYPUS		
	FEM&LFEP	RNAStructure 4.2 [14] [21]
aa	$\frac{\#correctBPs}{\#BPsintRNA}$	$\frac{\#correctBPs}{\#BPsintRNA}$
ala	95.24%	90.48%
arg	100.00%	55.00%
asn	70.00%	60.00%
asp	73.68%	57.89%
cys	94.74%	36.84%
glu	100.00%	71.43%
gln	42.86%	52.38%
gly	50.00%	65.00%
his	95.24%	95.24%
ile	100.00%	95.24%
leu1	90.48%	71.43%
leu2	89.47%	52.63%
lys	100.00%	100.00%
met	85.00%	95.00%
phe	90.00%	100.00%
pro	95.24%	57.14%
ser	95.24%	100.00%
thr	44.44%	22.22%
trp	95.00%	95.00%
tyr	100.00%	94.74%
val	100.00%	85.71%
$\geq 90\%$	66.67%	42.86%
$\geq 85\%$	76.19%	47.62%

Table 4.17: Comparison of FEM&LFEP and RNAStructure 4.2 [14] [21]. The results were obtained for the tRNA sequences of platypus.[16] The table computes the percentage of correct base pairs computed by each algorithm compared to the correct structure.

PLATYPUS		
	FEM&LFEP	RNAStructure 4.2 [14] [21]
aa	$\frac{\#correctBPs}{\#BPscomputed}$	$\frac{\#correctBPs}{\#BPscomputed}$
ala	90.91%	86.36%
arg	100.00%	45.83%
asn	77.78%	54.55%
asp	82.35%	57.89%
cys	90.00%	33.33%
glu	95.45%	68.18%
gln	56.25%	47.83%
gly	50.00%	59.09%
his	83.33%	90.91%
ile	91.30%	95.24%
leu1	82.61%	75.00%
leu2	80.95%	55.56%
lys	95.24%	100.00%
met	100.00%	100.00%
phe	100.00%	100.00%
pro	95.24%	54.55%
ser	90.91%	91.30%
thr	50.00%	22.22%
trp	95.00%	100.00%
tyr	100.00%	100.00%
val	100.00%	94.74%
$\geq 90\%$	61.90%	42.86%
$\geq 85\%$	61.90%	47.62%

Table 4.18: Comparison of FEM&LFEP and RNAStructure 4.2 [14] [21]. The results were obtained for the tRNA sequences of platypus.[16] The table computes the percentage of correct base pairs computed by each algorithm in relation to all base pairs computed by the algorithm.

HAMADRYAS BABOON		
	FEM&LFEP	RNAStructure 4.2 [14] [21]
aa	$\frac{\#correctBPs}{\#BPsintRNA}$	$\frac{\#correctBPs}{\#BPsintRNA}$
ala	95.24%	76.19%
arg	94.74%	57.89%
asn	85.00%	80.00%
asp	95.24%	95.24%
cys	75.00%	80.00%
glu	100.00%	52.38%
gln	95.00%	100.00%
gly	100.00%	95.24%
his	95.24%	23.81%
ile	95.00%	95.00%
leu1	90.00%	100.00%
leu2	100.00%	55.00%
lys	65.00%	95.00%
met	83.33%	94.44%
phe	68.42%	36.84%
pro	100.00%	57.14%
ser	71.43%	71.43%
thr	88.89%	77.78%
trp	95.00%	95.00%
tyr	100.00%	100.00%
val	77.78%	83.33%
$\geq 90\%$	61.90%	42.86%
$\geq 85\%$	71.43%	42.86%

Table 4.19: Comparison of FEM&LFEP and RNAStructure 4.2 [14] [21]. The results were obtained for the tRNA sequences of hamadryas baboon.[16] The table computes the percentage of correct base pairs computed by each algorithm compared to the correct structure.

HAMADRYAS BABOON		
	FEM&LFEP	RNAStructure 4.2 [14] [21]
aa	$\frac{\#correctBPs}{\#BPscomputed}$	$\frac{\#correctBPs}{\#BPscomputed}$
ala	95.24%	84.21%
arg	100.00%	84.62%
asn	73.91%	66.67%
asp	90.91%	90.91%
cys	65.22%	69.57%
glu	95.45%	47.83%
gln	95.00%	100.00%
gly	100.00%	100.00%
his	95.24%	25.00%
ile	82.61%	90.48%
leu1	90.00%	90.91%
leu2	100.00%	64.71%
lys	72.22%	100.00%
met	100.00%	100.00%
phe	81.25%	38.89%
pro	95.45%	52.17%
ser	65.22%	60.00%
thr	100.00%	82.35%
trp	95.00%	100.00%
tyr	95.24%	100.00%
val	77.78%	83.33%
$\geq 90\%$	66.67%	42.86%
$\geq 85\%$	66.67%	42.86%

Table 4.20: Comparison of FEM&LFEP and RNAStructure 4.2 [14] [21]. The results were obtained for the tRNA sequences of hamadryas baboon.[16] The table computes the percentage of correct base pairs computed by each algorithm in relation to all base pairs computed by the algorithm.

ORANGUTAN		
	FEM&LFEP	RNAStructure 4.2 [14] [21]
aa	$\frac{\#correctBPs}{\#BPsintRNA}$	$\frac{\#correctBPs}{\#BPsintRNA}$
ala	95.24%	23.81%
arg	100.00%	60.00%
asn	100.00%	85.00%
asp	95.24%	95.24%
cys	95.00%	100.00%
glu	57.89%	57.89%
gln	88.89%	100.00%
gly	71.43%	57.14%
his	95.00%	95.00%
ile	94.74%	94.74%
leu1	95.00%	95.00%
leu2	60.00%	55.00%
lys	71.43%	71.43%
met	60.00%	75.00%
phe	65.00%	75.00%
pro	90.48%	57.14%
ser	90.48%	100.00%
thr	82.35%	47.06%
trp	94.74%	78.95%
tyr	95.00%	95.00%
val	88.89%	94.44%
$\geq 90\%$	57.14%	42.86%
$\geq 85\%$	66.67%	47.62%

Table 4.21: Comparison of FEM&LFEP and RNAStructure 4.2 [14] [21]. The results were obtained for the tRNA sequences of orangutan.[16] The table computes the percentage of correct base pairs computed by each algorithm compared to the correct structure.

ORANGUTAN		
	FEM&LFEP	RNAStructure 4.2 [14] [21]
aa	$\frac{\#correctBPs}{\#BPscomputed}$	$\frac{\#correctBPs}{\#BPscomputed}$
ala	90.91%	25.00%
arg	100.00%	85.71%
asn	95.24%	89.47%
asp	86.96%	90.91%
cys	79.17%	83.33%
glu	57.89%	61.11%
gln	88.89%	81.82%
gly	83.33%	57.14%
his	86.36%	90.48%
ile	90.00%	95.00%
leu1	82.61%	86.36%
leu2	54.55%	55.00%
lys	75.00%	71.43%
met	66.67%	83.33%
phe	81.25%	100.00%
pro	90.48%	52.17%
ser	95.00%	95.45%
thr	82.35%	61.54%
trp	94.74%	88.24%
tyr	95.00%	95.00%
val	100.00%	100.00%
$\geq 90\%$	42.86%	33.33%
$\geq 85\%$	57.14%	52.38%

Table 4.22: Comparison of FEM&LFEP and RNAStructure 4.2 [14] [21]. The results were obtained for the tRNA sequences of orangutan.[16] The table computes the percentage of correct base pairs computed by each algorithm in relation to all base pairs computed by the algorithm.

PYGMY CHIMPANZEE		
	FEM&LFEP	RNAStructure 4.2 [14] [21]
aa	$\frac{\#correctBPs}{\#BPsintRNA}$	$\frac{\#correctBPs}{\#BPsintRNA}$
ala	90.91%	90.91%
arg	100.00%	25.00%
asn	65.00%	50.00%
asp	95.00%	95.00%
cys	95.00%	100.00%
glu1	90.00%	100.00%
glu2	86.36%	31.82%
gln	89.47%	100.00%
gly	71.43%	90.48%
his	70.00%	95.00%
ile	100.00%	78.95%
leu1	63.16%	57.89%
leu2	95.00%	55.00%
lys1	71.43%	33.33%
lys2	55.00%	55.00%
met	66.67%	83.33%
phe1	73.68%	78.95%
phe2	73.68%	78.95%
pro1	95.24%	57.14%
pro2	90.48%	57.14%
ser1	71.43%	76.19%
ser2	47.62%	66.67%
thr	63.16%	52.63%
trp	95.00%	60.00%
tyr	90.00%	95.00%
val	57.89%	78.95%
$\geq 90\%$	42.31%	30.77%
$\geq 85\%$	50.00%	30.77%

Table 4.23: Comparison of FEM&LFEP and RNAStructure 4.2 [14] [21]. The results were obtained for the tRNA sequences of pygmy chimpanzee.[11] The table computes the percentage of correct base pairs computed by each algorithm compared to the correct structure.

PYGMY CHIMPANZEE		
	FEM&LFEP	RNAStructure 4.2 [14] [21]
aa	$\frac{\#correctBPs}{\#BPscomputed}$	$\frac{\#correctBPs}{\#BPscomputed}$
ala	90.91%	95.24%
arg	100.00%	26.32%
asn	59.09%	45.45%
asp	90.48%	90.48%
cys	86.36%	90.91%
glu1	90.00%	90.91%
glu2	100.00%	29.17%
gln	89.47%	86.36%
gly	83.33%	95.00%
his	66.67%	90.48%
ile	90.48%	75.00%
leu1	66.67%	68.75%
leu2	100.00%	50.00%
lys1	75.00%	31.82%
lys2	52.38%	47.83%
met	66.67%	83.33%
phe1	73.68%	75.00%
phe2	73.68%	75.00%
pro1	95.24%	54.55%
pro2	90.48%	50.00%
ser1	62.50%	69.57%
ser2	41.67%	63.64%
thr	100.00%	71.43%
trp	100.00%	60.00%
tyr	94.74%	95.00%
val	78.57%	78.95%
$\geq 90\%$	46.15%	26.92%
$\geq 85\%$	53.85%	30.77%

Table 4.24: Comparison of FEM&LFEP and RNAStructure 4.2 [14] [21]. The results were obtained for the tRNA sequences of pygmy chimpanzee.[16] The table computes the percentage of correct base pairs computed by each algorithm in relation to all base pairs computed by the algorithm.

RAT					
	FEM&LFEP	RNAStructure 4.2 [14] [21]		FEM&LFEP	RNAStructure 4.2 [14] [21]
aa	$\frac{\#correctBPs}{\#BPsintRNA}$	$\frac{\#correctBPs}{\#BPsintRNA}$	aa	$\frac{\#correctBPs}{\#BPsintRNA}$	$\frac{\#correctBPs}{\#BPsintRNA}$
ala	95.24%	95.24%	gly1	95.24%	90.48%
arg	95.45%	54.55%	gly2	85.71%	85.71%
asn1	75.00%	90.00%	gly3	85.71%	85.71%
asn2	80.00%	65.00%	his	90.48%	23.81%
asn3	65.00%	65.00%	ile	100.00%	95.00%
asp1	95.24%	95.24%	leu1	95.00%	95.00%
asp2	100.00%	76.19%	leu2	76.19%	52.38%
cys1	95.24%	100.00%	leu3	100.00%	80.95%
cys2	95.24%	100.00%	leu4	100.00%	52.38%
glu1	100.00%	95.24%	leu5	85.71%	57.14%
glu2	100.00%	33.33%	lys1	41.18%	23.53%
glu3	100.00%	100.00%	lys2	72.22%	50.00%
glu4	70.00%	25.00%	lys3	71.43%	76.19%
glu5	100.00%	95.24%	phe1	89.47%	100.00%
glu6	100.00%	95.24%	phe2	66.67%	57.14%
glu7	100.00%	95.24%	pro1	63.64%	59.09%
glu8	100.00%	95.24%	pro2	63.64%	77.27%
glu9	100.00%	95.24%	pro3	95.24%	80.95%
glu10	100.00%	95.24%	pro4	90.48%	80.95%
gln1	65.00%	55.00%	ser1	76.19%	76.19%
gln2	100.00%	57.14%	ser2	76.19%	100.00%
gln3	100.00%	95.24%	thr	94.74%	89.47%
gln4	100.00%	95.24%	trp1	95.00%	90.00%
gln5	100.00%	52.38%	trp2	100.00%	60.00%
gln6	100.00%	95.24%	tyr	100.00%	95.00%
gln7	100.00%	95.24%	val	89.47%	89.47%
gln8	100.00%	95.24%			
$\geq 90\%$	64.15%	47.17%			
$\geq 85\%$	73.58%	54.72%			

Table 4.25: Comparison of FEM&LFEP and RNAStructure 4.2 [14] [21]. The results were obtained for the tRNA sequences of rat.[11] The table computes the percentage of correct base pairs computed by each algorithm compared to the correct structure.

RAT					
	FEM&LFEP	RNAStructure 4.2 [14] [21]		FEM&LFEP	RNAStructure 4.2 [14] [21]
aa	$\frac{\#correctBPs}{\#BPscomputed}$	$\frac{\#correctBPs}{\#BPscomputed}$	aa	$\frac{\#correctBPs}{\#BPscomputed}$	$\frac{\#correctBPs}{\#BPscomputed}$
ala	90.91%	90.91%	gly1	100.00%	100.00%
arg	91.30%	60.00%	gly2	85.71%	85.71%
asn1	65.22%	85.71%	gly3	85.71%	85.71%
asn2	76.19%	54.17%	his	100.00%	26.32%
asn3	68.42%	59.09%	ile	90.91%	90.48%
asp1	90.91%	90.91%	leu1	90.48%	90.48%
asp2	100.00%	76.19%	leu2	84.21%	55.00%
cys1	86.96%	87.50%	leu3	100.00%	73.91%
cys2	86.96%	87.50%	leu4	100.00%	50.00%
glu1	95.45%	95.24%	leu5	100.00%	57.14%
glu2	100.00%	33.33%	lys1	63.64%	30.77%
glu3	100.00%	100.00%	lys2	76.47%	50.00%
glu4	100.00%	26.32%	lys3	71.43%	72.73%
glu5	95.45%	100.00%	phe1	94.44%	100.00%
glu6	95.45%	100.00%	phe2	70.00%	48.00%
glu7	95.45%	100.00%	pro1	73.68%	68.42%
glu8	95.45%	100.00%	pro2	73.68%	89.47%
glu9	95.45%	100.00%	pro3	90.91%	70.83%
glu10	95.45%	100.00%	pro4	90.48%	70.83%
gln1	72.22%	47.83%	ser1	72.73%	69.57%
gln2	91.30%	46.15%	ser2	69.57%	91.30%
gln3	95.45%	100.00%	thr	94.74%	94.44%
gln4	95.45%	100.00%	trp1	100.00%	100.00%
gln5	95.45%	45.83%	trp2	100.00%	80.00%
gln6	95.45%	100.00%	tyr	100.00%	100.00%
gln7	95.45%	100.00%	val	89.47%	89.47%
gln8	95.45%	100.00%			
$\geq 90\%$	66.04%	43.40%			
$\geq 85\%$	75.47%	56.60%			

Table 4.26: Comparison of FEM&LFEP and RNAStructure 4.2 [14] [21]. The results were obtained for the tRNA sequences of rat.[11] The table computes the percentage of correct base pairs computed by each algorithm in relation to all base pairs computed by the algorithm.

RHINOCEROS		
	FEM&LFEP	RNAStructure 4.2 [14] [21]
aa	$\frac{\#correctBPs}{\#BPsintRNA}$	$\frac{\#correctBPs}{\#BPsintRNA}$
ala	65.00%	55.00%
arg	100.00%	95.24%
asn	70.00%	60.00%
asp	50.00%	50.00%
cys	95.00%	55.00%
glu	100.00%	80.95%
gln	65.00%	25.00%
gly	100.00%	90.48%
his	95.00%	95.00%
ile	95.00%	95.00%
leu1	90.00%	100.00%
leu2	95.00%	35.00%
lys	100.00%	100.00%
met	77.78%	94.44%
phe	88.89%	100.00%
pro	95.24%	57.14%
ser	47.62%	76.19%
thr	88.89%	77.78%
trp	95.00%	95.00%
tyr	100.00%	95.00%
val	73.68%	94.74%
$\geq 90\%$	57.14%	52.38%
$\geq 85\%$	66.67%	52.38%

Table 4.27: Comparison of FEM&LFEP and RNAStructure 4.2 [14] [21]. The results were obtained for the tRNA sequences of rhinoceros.[16] The table computes the percentage of correct base pairs computed by each algorithm compared to the correct structure.

RHINOCEROS		
	FEM&LFEP	RNAStructure 4.2 [14] [21]
aa	$\frac{\#correctBPs}{\#BPscomputed}$	$\frac{\#correctBPs}{\#BPscomputed}$
ala	61.90%	45.83%
arg	100.00%	100.00%
asn	77.78%	52.17%
asp	47.37%	42.86%
cys	90.48%	55.00%
glu	95.45%	80.95%
gln	65.00%	27.78%
gly	100.00%	95.00%
his	90.48%	90.48%
ile	90.48%	90.48%
leu1	81.82%	90.91%
leu2	95.00%	41.18%
lys	95.24%	100.00%
met	82.35%	100.00%
phe	88.89%	90.00%
pro	95.24%	63.16%
ser	47.62%	69.57%
thr	94.12%	73.68%
trp	95.00%	100.00%
tyr	95.24%	100.00%
val	73.68%	90.00%
$\geq 90\%$	57.14%	52.38%
$\geq 85\%$	61.90%	52.38%

Table 4.28: Comparison of FEM&LFEP and RNAStructure 4.2 [14] [21]. The results were obtained for the tRNA sequences of rhinoceros.[16] The table computes the percentage of correct base pairs computed by each algorithm in relation to all base pairs computed by the algorithm.

WILD BOAR		
	FEM&LFEP	RNAStructure 4.2 [14] [21]
aa	$\frac{\#correctBPs}{\#BPsintRNA}$	$\frac{\#correctBPs}{\#BPsintRNA}$
ala	75.00%	55.00%
arg	100.00%	95.24%
asn	100.00%	95.24%
asp	95.24%	95.24%
cys	95.24%	100.00%
glu	95.24%	76.19%
gln	95.00%	100.00%
gly	95.00%	90.00%
his	95.00%	60.00%
ile	95.00%	95.00%
leu1	90.00%	100.00%
leu2	100.00%	55.00%
lys	90.00%	100.00%
met	66.67%	94.44%
phe	81.25%	56.25%
pro	95.00%	90.00%
ser	71.43%	76.19%
thr	90.00%	95.00%
trp	95.00%	90.00%
tyr	65.00%	95.00%
val	77.78%	94.44%
$\geq 90\%$	71.43%	71.43%
$\geq 85\%$	71.43%	71.43%

Table 4.29: Comparison of FEM&LFEP and RNAStructure 4.2 [14] [21]. The results were obtained for the tRNA sequences of wild boar.[16] The table computes the percentage of correct base pairs computed by each algorithm compared to the correct structure.

WILD BOAR		
	FEM&LFEP	RNAStructure 4.2 [14] [21]
aa	$\frac{\#correctBPs}{\#BPscomputed}$	$\frac{\#correctBPs}{\#BPscomputed}$
ala	68.18%	52.38%
arg	100.00%	100.00%
asn	100.00%	100.00%
asp	86.96%	90.91%
cys	90.91%	91.30%
glu	95.24%	76.19%
gln	90.48%	90.91%
gly	95.00%	95.00%
his	90.48%	57.14%
ile	90.48%	90.48%
leu1	90.00%	90.91%
leu2	95.24%	57.89%
lys	94.74%	100.00%
met	80.00%	94.44%
phe	86.67%	75.00%
pro	95.00%	94.74%
ser	68.18%	72.73%
thr	100.00%	95.00%
trp	95.00%	90.00%
tyr	92.86%	100.00%
val	77.78%	85.00%
$\geq 90\%$	71.43%	66.67%
$\geq 85\%$	80.95%	71.43%

Table 4.30: Comparison of FEM&LFEP and RNAStructure 4.2 [14] [21]. The results were obtained for the tRNA sequences of wild boar.[16] The table computes the percentage of correct base pairs computed by each algorithm in relation to all base pairs computed by the algorithm.

Chapter 5

Future Work

5.1 The FEM&LFEP Algorithm

Despite the improved performance, the FEM&LFEP algorithm may be customized for tRNA sequences to produce the cloverleaf secondary structure. Before I discuss the bias in my algorithm, I would like to point out that the idea of preferred initial local folding and the stability of earlier substructures seems valid as evidenced by the algorithm's performance, and it is worth exploring further.

The customization of the FEM&LFEP algorithm for tRNA sequences occurs in the implementation of the LFEP part of the algorithm. Specifically, the influence is evidenced in the data collection step in two ways. First, the algorithm purposely ignores fixed numbers of bases at the ends of the sequence. This is due to the a priori knowledge that the ends bind to form the acceptor arm. Second, the restriction of twenty time units that is placed for data collection also requires a priori knowledge about the sizes of the D, anticodon, and TψC loops.

5.2 General Improvements

One problem that is common to the FEM&LFEP algorithm and RNAStructure 4.2 [14] [21] is computing the correct hairpin structure. Specifically, both algorithms compute the smallest possible hairpin loops rather than those with the correct size. For example, in the anticodon loop whose sequence is $UUACUUUGAUAGUAA$, both algorithms compute (((((((...))))))), a stem composed of seven consecutive base pairs and a hairpin loop of size three, as the secondary structure while the correct configuration is (((((.....))))), an arm with five consecutive base pairs and a hairpin loop of size seven. Because in both energy models, stacking generally has a stabilizing effect on the structure, both algorithms favor the first structure over the later. In many cases, more stacking gives more stable substructures. Thus, whenever a subsequence such as $UUACUUUGAUAGUAA$ is folded by either of the implementations, the resulting structure contains two more base pairs than it should.

The problem may be handled by obtaining better energy parameters. The Vienna package [17] [27] and RNAStructure 4.2 [14] [21] provide proof that the updated energy model and recurrence functions can remarkably improve the accuracy of the algorithm. Implementing the FEM&LFEP algorithm with the most current energy model will likely produce more accurate results. It is likely that most of the energy parameters implemented by RNAStructure 4.2 [14] [21] are already sufficiently accurate. Therefore, the thermodynamic and chemical data that should be obtained are those for the modified bases. The new energy model used by RNAStructure 4.2 [14] [21] incorporates some of the effects of the modified bases. By obtaining more information, the algorithm likely will be improved. Specifically, the new data will describe the effect of each of the modified bases such as whether each modified base remains complementary to its original base partner and if so, the free energy of their base pairing. Or, if not, by how much the modification increases the molecule's free energy. These data will lead to an even more accurate energy model.

The improved energy model leads to a need for obtaining the tRNA sequences that reflect all post-transcriptional base modifications. Full utilization of data can be realized by executing the algorithm on these sequences.

Lastly, another improvement that can be made to the FEM&LFEP algorithm is the modeling of the bulge loop, internal loop, and multiloop structures. As I noted earlier, I simplified the recurrence functions of these substructures to show that the folding process is not just pursuing the minimum free energy structure but also a function of time. More precisely modeling the free energy functions of these substructures will likely produce more accurate results, as they did for Zuker's free energy minimization algorithm. The difference between the performances by the Vienna package [17] [27] and RNAStructure 4.2 [14] [21] is significant.

Chapter 6

Conclusions

Using a test set containing tRNA sequences of fifteen different organisms, the FEM&LFEP algorithm outperformed RNAStructure 4.2 [14] [21] for thirteen organisms while it produced comparable results as RNAStructure 4.2 [14] [21] for the remaining two organisms. The importance of modeling the folding process as a function of time in conjunction with the free energy minimization concept and the need for such subdivision of the computation method seem to be supported by the consistent performance of the FEM&LFEP algorithm.

However, free energy minimization and lowest free energy path do not seem to be the sole determinant of the folding of RNA molecules into their secondary structures. The computation problem with the hairpin structure mentioned in the previous chapter is one of the factors that led to the idea of a multi-component folding mechanism. Perhaps, this is one of the places where the post-transcriptional base modifications come into play. As shown in figure 1.1 and explained in the maturation process, certain bases at certain positions in a tRNA sequence are frequently modified. Using the same example as the previous chapter, among the modified bases are those at the positions of AU in the anticodon loop whose sequence is $UUACUUUGAUAGUAA$. As previously discussed, these base modifications may create steric hindrance that

prevents the bases from forming hydrogen bonds, or they may alter the chemical properties of the bases in such a way that it requires more energy to form hydrogen bonds than it originally did, therefore, these particular bases are no longer as thermodynamically attracted as they were before the modifications. Or the modified bases may simply become unable to interact with other bases. If any of the three theories is actually what happens, then the anticodon loop would be computed correctly.

In addition, the post transcriptional base modifications have been shown to participate in the correct folding of tRNAs into the tertiary structure. For example, the modifications in the D loop and the TψC loop have been identified as the stabilizers of the tertiary structure. [22] Because the correct folding of the tertiary structure strongly depends on the correct folding of the secondary structure, these base modifications most likely participate in the secondary structure folding process as well.

Also, the multi-component folding mechanism hypothesis may be supported by the presence of chaperone proteins. Chaperone proteins are a class of proteins that assists other molecules to perform their roles. In terms of structures and tRNAs, chaperone proteins help to maintain and stabilize tRNA's tertiary structure.[22] Again, because the correct folding of the tertiary structure requires the correct secondary structure, these chaperone proteins may also play a role in the secondary structure folding and stabilization.

Despite the amount of suggested future work, the FEM&LFEP algorithm has shown some promise by consistently outperforming RNAStructure 4.2 [14] [21] in computing the secondary structure of tRNA molecules. Even though there seems to be more studies that can be done to help improve the algorithms for computing secondary structures of biological molecules, the FEM&LFEP algorithm's attempt to subdivide the folding process into two phases by implementing the LFEP and FEM improves the outcome of the computation.

Bibliography

[1] http://www.cmb.usc.edu/cbmp/2001/tRNA/trna\%20bases.htm.

[2] Biochemistry 3107 - fall 2002 trna the adaptor hypothesis and the wobble hypothesis. http://www.mun.ca/biochem/courses/3107/Lectures/Topics/tRNA.html.

[3] Inosine is a post-transcriptionally modified nucleotide in trna. http://oregonstate.edu/instructin/bb331/lecture12/AMP-IMP.html.

[4] Lecture 12 - rna processing: trna and rrna. http://www.wisc.edu/molpharm/Courses/pharm620/Lecture_12.web.ppt.

[5] Lecture 4: Line segment intersection. http://www.cs.wustl.edu/\~pless/546/lectures/l4.html.

[6] Non-coding rna. http://en.wikipedia.org/wiki/RRNA.

[7] Rna secondary structure prediction. http://www.cse.lehigh.edu/~lopresti/Courses/2003-04/CSE397-497/lecture\%_18.ppt.

[8] Secondary structure. http://en.wikipedia.org/wiki/Secondary_structure.

[9] Structure of trna. http://www.web-books.com/MoBio/Free/Ch3C2.htm.

[10] trna. http://www.biochem.uwo.ca/meds/medna/tRNA.html.

[11] trna compilation 2000. http://www.staff.uni-bayreuth.de/~btc914/search/.

[12] Vienna rna package. http://bioweb.pasteur.fr/docs/ViennaRNA/RNAlib_toc.html\#TOC1.

[13] Becker, Reece, and Poenie. *THE WORLD OF THE CELL*. The Benjamin/Cummings Publishing Company, 1996.

[14] Matthews DH, Disney MD, Childs JL, Schroeder SJ, Zuker M, and Turner DH. Rnastructure, version 4.1. http://128.151.176.70/RNAstructure.html.

[15] Christoph Flamm, Walter Fontana, Ivo L. Hofacker, and Peter Schuster. Rna folding at elementary step resolution. *RNA*, 6:325–338, 2000.

[16] Catherine Florentz and Joern Pürtz. Compilation of mitochondrial mammalian trna genes. `http://www.mamit-trna.u-strasbg.fr/index.html`.

[17] Ivo Hofacker. Vienna rna package rna secondary structure prediction and comparison. `http://www.tbi.univie.ac.at/ivo/RNA`.

[18] Ivo L Hofacker, Walter Fontana, Sebastian Bonhoeffer, and Peter F Stadler. Rnafold. `http://www.tbi.univie.ac.at/\~ivo/RNAfold.html`.

[19] Samuel Leong. Problem definition. `http://zoo.cs.yale.edu/classes/cs490/99-00b/ieong.samuel.ssi2/node2.htm%l`.

[20] Hartmut "Hudel" Luecke. The wobble theory. `http://bass.bio.uci.edu/hudel/bs99a/lecture21/lecture2\_4.html`.

[21] David H. Matthews, Matthew D. Disney, Jessica L. Childs, Susan J. Schroeder, Michael Zuker, and Douglas H. Turner. Incorporating chemical modification constraints into a dynamic programming algorithm for prediction of rna secondary structure. *PNAS*, 101(19):7287–7292, 200r.

[22] Kentaro Nakanishi and Osamu Nureki. Recent progress of structural biology of trna processing and modification. *Molecules and Cells*, 19(2):157–166, 2005.

[23] Wilfred Ndifon. A complex adaptive systems approach to the kinetic folding of rna. *Biosystems*, 82(3):203–292, 2005.

[24] Catherine Papanicolaou, Manolo Gouy, and Jacques Ninio. An energy model that predicts the correct folding of both the trna and the 5s rna molecules. *Nucleic Acid Research*, 12(1):133–148, 1984.

[25] Richard A. Paselk. Translation trna. `http://www.humboldt.edu/rap1/C432.S02/C432Notes/C432n10Apr.html`.

[26] D. Pörshke. Model calculations on the kinetics of oligonucleotide double helix coil transisions. evidence for a fast chain sliding reaction. *Biophys Chem*, 2(2):83–96, 1974.

[27] Michael Zuker and Patrick Stiegler. Optimal computer folding of large rna sequences using thermodynamics and auxiliary information. *Nucleic Acids Research*, 9(1):133–148, 1981.