RNA Folding: Local Versus Global Optimization

Max Ward (20748588)

This report is submitted as partial fulfilment
of the requirements for the Honours Programme of the
School of Computer Science and Software Engineering,
The University of Western Australia,
2014

Abstract

I am definitely going to need to write this at some point. This is a short report on how to use the `cshonours.cls` class to prepare dissertations using the latest LaTeX version, LaTeX2e. This class is based on the standard class `report.cls`.

Keywords: Honours, report, dissertation, UWA, RNA, bioinformatics
CR Categories: Not, really, sure

Acknowledgements

Going to need something here too. This class is designed to produce reports that look the same as those produced by the older `cshonours.sty` style for LaTeX2.09, which was modified by Nick Spadaccini from a style provided by Ken Wessen.

Contents

List of Tables

List of Figures

CHAPTER 1

Introduction

1.1 DNA and RNA

Deoxyribonucleic Acid (DNA) is the basic genetic building block upon which the classification of genetic material into genes and chromosomes is based. The role of DNA as the hereditary unit of genetics was determined in the 1940s [1]. Soon thereafter, Watson & Crick [35] published a highly acclaimed paper describing the fundamental chemical structure of DNA. In it, they outlined a double helix formation which has since become as iconic as it is canonical (see Figure 1.1). Each strand of the helix Watson & Crick discovered is essentially a chain of 'nucleotides'. Each nucleotide is made of a sugar-phosphate backbone attached to a single 'base'. The bases of corresponding strands form hydrogen bonds which hold the double helix together. The most astonishing and important of their findings was that these bases bond in a reciprocal fashion. They described four bases: Adenine (A), which always bonds to Thymine (T), and Guanine (G), which always bonds to Cytosine (C).

The reciprocal bonding relationships between bases is what allows replication to occur; a copy of the DNA can be made by simply allowing the correct bases to bond to one of the strands making up its helix. This gives a model for inheritance and cellular replication. However, there remains the question of how DNA can actually code for protein. Proteins are made up amino acids bonded in a specific sequence [1]. The DNA must therefore code for amino acids. This code, which can be thought of as the 'digital' representation for the 'analogue' protein used by our cells, needs to be carried to ribosomes which translate it into protein [1]. This is a task carried out by Ribonucleic Acid (RNA). RNA is very much like DNA in that it can bond reciprocally to another strand with matching bases. The main difference is that it is single stranded in structure, and has Uracil (U) in place of Thymine [1]. It is important to note that in RNA molecules G and U pairings are also possible. RNA bonds to DNA and, in a sense, reads it. This results in the production of a copy of the DNAs genetic payload. This

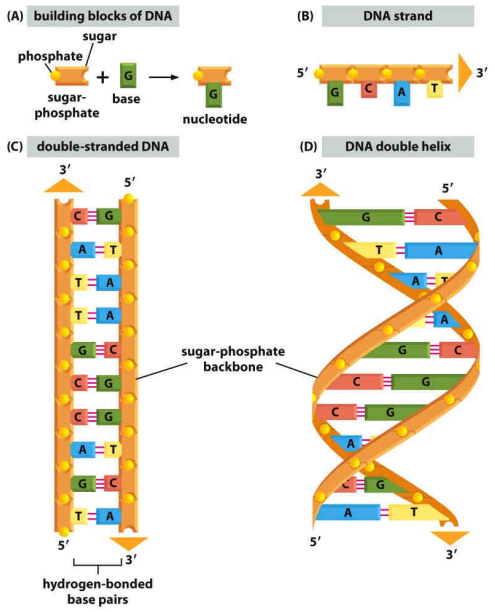

Figure 1.1: The structure and composition of DNA. Diagram taken from *Essential Cell Biology* [1].

'downloaded' information is then carried away to be translated into protein [1]. An example of this is depicted in Figure 1.2, in which we see a Messenger RNA molecule bonding to and thus making a copy of a section of DNA. As depicted in Figure 1.2, the 3' end of a DNA or RNA molecule is the end onto which new nucleotides are added. The 5' end is chemically stable, and nucleotides are not usually appended to it [1].

For many years the conventional wisdom was that DNA contained genes which coded for functional proteins used by the cell [1]. Though this is undoubtedly true, there was a problem: much of the human genome, and the genomes of other species, contains DNA which does not appear to code for anything [4]. Many theories have been put forward to explain this. It was argued that this 'junk' DNA is the perennial build-up of mutation, and that natural selection simply cannot act with strong enough selective force to cull this free-loading DNA [4]. Surprisingly, much of this non-coding DNA is actually transcribed into RNA, despite having no apparent function [16].

As it turns out, RNA is more than a simple messenger for encoded proteins. Recent research has found myriad important functions for RNA. For example, RNA can act as a catalyst for RNA splicing and peptide bond formation, and can also alter the regulation of genes [37]. It seems that much of our genome con-

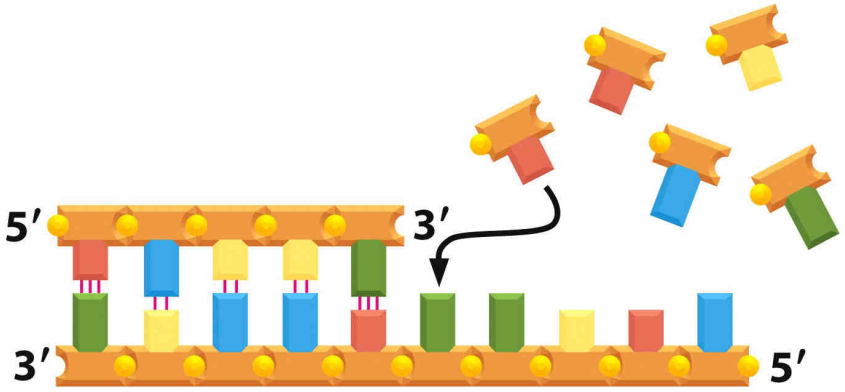

Figure 1.2: RNA transcription. Diagram taken from *Essential Cell Biology* [1].

tains templates for non-coding RNAs (ncRNAs). These RNAs perform essential cellular functions without actually being translated into protein at any point in their life-cycle [16]. Because of its inherently single stranded nature, RNA forms bonds with itself, folding into secondary and tertiary structures [6].

It is axiomatic that chemical structure is tantamount to biological function; RNA is no exception. For this reason there has and continues to be an intense interest in predicting the secondary structure and tertiary structure of RNA molecules. This is in part because it will elucidate the underlying principles of RNA structure formation and function [6], but also because it will allow the detection and classification of unknown RNAs, enable prediction of novel RNA function, and assist the design of new RNA based drugs [5]. In fact, RNA is an extremely versatile molecule, and as such is attractive from both an engineering and computational point of view. Small combinatorial computational problems have been solved by representing the solution set using RNAs. Furthermore, a theory of computation has been put forward using self assembling RNA molecules [5]. As if to comment on the upheaval of a protein-centric view of biology in recent years, researchers have found that RNA is capable of supporting all the processes required for life without the need of protein [5]. The secondary structure of RNA is also highly conserved during evolution, indicating its importance [11].

Secondary and tertiary structures can be treated hierarchically, as a result it is possible to predict the secondary structure of an RNA without understanding the tertiary structure. The tertiary structure in turn builds upon the secondary structure [32]. This paper will focus on secondary structure prediction.

It holds to reason that an algorithm for RNA secondary structure prediction

can never be realised if we do not understand how these structures form, or their general morphology. For this reason it is important to understand how true RNA secondary structures can be determined, and the limitations of these techniques. DNA and RNA molecules can be analysed using X-ray crystallographic methods. These types of approaches work because the wavelengths of some X-rays are the same as the dimensions of DNA and RNA inter-atomic bonds. The diffraction of X-ray light by these molecules can thus be observed and their structures can subsequently be inferred by analysis of the resulting data. Nuclear Magnetic Resonance (NMR) is another technique which can be applied to the analysis of DNA/RNA. It relies on the spin of atoms when in a magnetic field. These spin signals can be used to determine the atomic composition and topology of a molecule. This has the advantage of not requiring the molecule under observation to be crystallized before analysis. Arguably this gives a better in vivo view of RNAs/DNAs, which are fundamentally flexible structures. NMR also has some disadvantages; for instance, it is less accurate than X-ray crystallography, and cannot be used on extremely large molecules. The reason these techniques cannot be used for all RNA structural assays is that they are extremely expensive and time consuming. For more information refer to *Principles of Nucleic Acid Structure* by Stephen Neidle [20].

RNA secondary structure prediction techniques can be broadly broken into two categories: those that use auxiliary information to assist in prediction, and those that predict structure ex nihilo—that is, with nothing but the 'proband' sequence we require a structure for. The former approach typically does consensus matching between some sequences for which a user already knows the secondary structures, and a sequence for which the structure is unknown [11]. In this paper I investigate the latter approach because it requires deeper knowledge about why and how RNAs fold. Also, it is the more general of the two.

1.2 Dynamic Programming Techniques

1.2.1 Fundamental Algorithms

The first such algorithms were based on relatively naive brute force. All possible secondary structures were enumerated, and the one with the most bonds was selected as the solution [22]. While being very simplistic these first approaches introduce an important assumption: RNA molecules will form energetically stable secondary structures. Maximising bonds is a crude but nonetheless accurate measure of energetic stability, as every bond increases the stability of a structure [22]. In the late 1970s, when the first large RNA molecules were being successfully

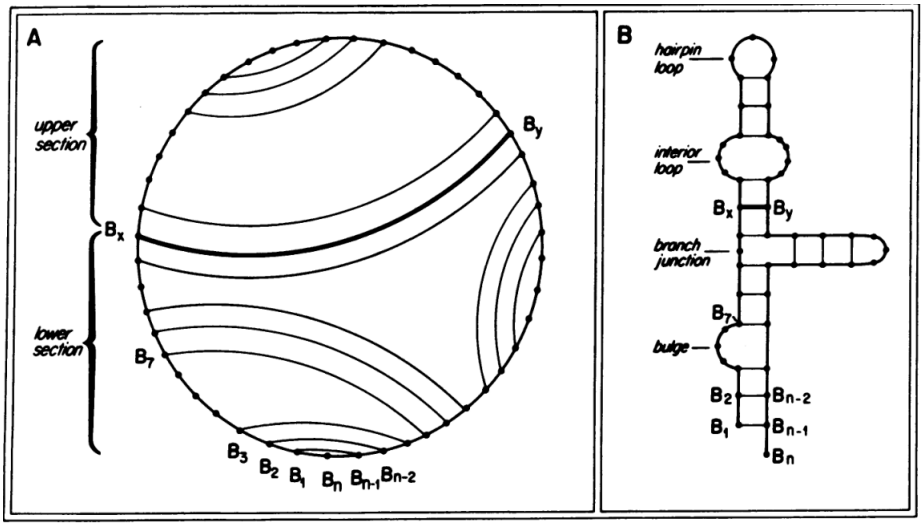

Figure 1.3: RNA secondary structure as described in the Nussinov algorithm. Taken from the original publication [21].

sequenced, Nussinov et al. [22] introduced an algorithm based on loop matching for bonding pairs. Their algorithm attempted to find a single structure with the maximal number of bonds using dynamic programming, with the restriction that all bonding pairs had to be entirely nested. It did this in $O(N^3)$ time and using $O(N^2)$ space. Thence Nussinov & Jacobson [21] introduced a refined version of the same algorithm and began testing it against experimentally verified RNA secondary structures. They had mixed success; transfer RNAs (tRNAs) were conspicuous in their difference from predicted structures.

Because of its dynamic programming nature, this algorithm performs recursive decompositions of the RNA, building larger structures out of repeated substructures. A natural representation of this is depicted in Figure 1.3. Part A of Figure 1.3 shows bonds as arcs across a circular graph. In it, we see the nested nature of the structures being explored by the Nussinov algorithm. Part B shows how these structures translate to actual RNAs in vivo. It also introduces the standard decompositions of secondary structures, namely the hairpin loop, the interior loop, and the branch junction or multi loop. Unlabelled in the diagram are stems; these are stacked base pairings, for example $B1$, $Bn-1$ and $B2$, $Bn-2$.

$$M(i,j) = \max\{A, B, C, D\} \tag{1.1}$$
$$A = M(i, j-1)$$

$$B = M(i+1, j)$$
$$C = M(i+1, j-1) + W(i, j)$$
$$D = \max\{M(i, k) + M(k+1, j)\} \; when \; i < k < j$$

The recurrence relation for the Nussinov algorithm is defind in Equation 1.1. The first two cases (A and B) find the score associated with not allowing i and j to bond. Case C conversely determines the score given that i and j are bonded. The final case D computes the score associated with a bifurcation. A bifurcation here means decomposition of the RNA into two separate structures. This recurrence relation implies a $O(N^3)$ worst case time complexity and a $O(N^2)$ space complexity, as an $O(N^2)$ state space (all combinations of i and j) is explored with a linear time recurrence relation. In the original algorithm a constant ($p = 3$) was introduced that indicated the minimum size of a hairpin loop, as real RNAs typically do not have hairpin loops of fewer bases. The recurrence relation presented here has also been modified for the sake of clarity (cases A and B can be merged into case D) but the logic of the algorithm is equivalent.

This algorithm can also be extended to accommodate a more advanced energy model. Instead of weighting each bond equally they can be weighted according to the proportion they are expected to contribute to the molecule's stability [21]. When considering the value of a bond, it might be given greater weight if it adds to the formation of a stem (a stabilizing structure), or given lower weight if it forms an internal loop or bulge, as these generally destabilize RNA molecules [21]. Unfortunately it is hard to find good values for such weights, and determining which substructure a bond contributes to requires backtracking in the modified algorithm presented by Nussinov & Jacobson.

The reader should note that the Nussinov algorithm is old technology, and is no longer used for the prediction of RNA secondary structures. I have presented it in detail here because it forms the basis for the most widely used algorithm today. This algorithm in turn forms the basis for my own algorithm.

Soon after the work of Nussinov & Jacobson, Zuker & Stiegler [39] described an altered version of the same algorithm which, instead of maximising base pairs, minimized the free energy of secondary structures. This is done by introducing a number of thermodynamic rules for canonical structures like hairpin loops, internal bulges, multiloops, unbonded base pairs, and stacked base pairs. The algorithm is similar to the Nussinov algorithm, but adds another mutually recursive dynamic programming recurrence to inject a complex and relatively comprehensive scoring system. The original thermodynamic scoring system used is borrowed from the work of Studnicka et al. [30] who presented a complex but theoretically

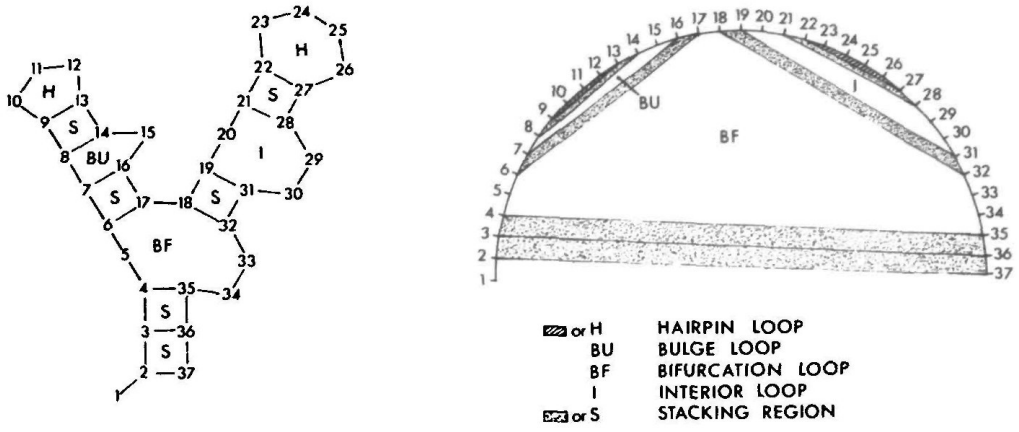

Figure 1.4: Diagram of faces used in the Zuker algorithm. Taken from original publication [39].

similar algorithm, albeit with much worse asymptotic and implementation complexities.

First I shall introduce some useful terminology which should clarify aspects of Zuker & Stieglers algorithm. The bases of an RNA molecule can be thought of as vertices in a planar graph. Edges between such vertices are then represented as chords on a semicircular diagram (Figures 1.3 and 1.4). These chords are not allowed to touch. A chord is admissible if it represents a chemically valid bond, and an admissible structure is a structure whose graph contains only admissible bonds. Thence, one can define a face of such a graph as any planar region bounded on all sides—either by chords, or the edge of the graph. The folding algorithm of Zuker & Stiegler considers such faces as the basic contributing factor to a molecule's stability, unlike the algorithm of Nussinov & Jacobson which considers only individual bonds.

Let $E(F)$ represent the energy of a face F; impossible structures are given an energy value of infinity, for example hairpin loops with less than three bases in the intervening loop region. In addition let $V(i,j)$ be defined as the minimum free energy given that bases i and j are bonded, and let $W(i,j)$ represent the minimum free energy of all structures contained within bases i and j inclusive. Note that for $W(i,j)$ there need not be a bond between bases i and j. Also, if i and j cannot bond then $V(i,j) = \infty$. Finally note that $FH(i,j)$ represents a hairpin loop structure from i to j, and that $FL(i,j,i',j')$ is defined as the face bounded by the bonds i,j and i',j'. Examples of these decompositions are shown

diagrammatically in the right half of Figure 1.4. The labelled regions show faces in a semicircular graph representing a strand of RNA. In the accompanying left half of the figure, the same RNA structure is shown as it would appear in a real RNA rather than in a purely diagrammatic depiction.

$$
\begin{aligned}
V(i,j) &= \min\{E1, E2, E3\} \\
E1 &= E(FH(i,j)) \\
E2 &= \min\{E(FL(i,j,i',j')) + V(i',j')\} \; where \; i < i' < j' < j \\
E3 &= \min\{W(i+1,i') + W(i'+1,j-1)\} \; where \; i+1 < i' < j-2
\end{aligned}
\tag{1.2}
$$

As shown by the definition provided in Equation 1.2, $V(i,j)$ is computed by minimizing three cases. The first case considers the bond between i and j closing off a hairpin loop (H in Figure 1.4). The second accounts for situations in which i and j are bonded. This can result in a bulge (BU in Figure 1.4), internal loop (I in Figure 1.4), or the continuation of a stacking region with the interior bond i', j' (S in Figure 1.4). The third and final case considers bifurcations ending with a bond between i and j (BF in Figure 1.4).

$$
\begin{aligned}
W(i,j) &= \min\{W(i+1,j), W(i,j-1), V(i,j), E4\} \\
E4 &= \min\{W(i,i') + W(i'+1,j)\} \; where \; i < i' < j-1
\end{aligned}
\tag{1.3}
$$

Equation 1.3 is the recurrence for $W(i,j)$ as described by Zuker & Stiegler. Again there are three cases. The first two cases $W(i+1,j)$ and $W(i,j-1)$ should be thought of as a single case which consider situations in which there is no bond between i and j. This is similar to cases A and B from the Nussinov algorithm (Equation 1.1). The third case considers taking the bond from i to j. The fourth and final case allows for bifurcations in which two bonding regions split the structure into two sections. The final minimum free energy of the best structure is defined by $W(1,n)$, where n is the length of the RNA molecule. It should be noted that the free energy for small molecules (fewer than 6 nucleotides in length) can easily be precomputed, and forms the base case of the given recurrence relations. Because of its efficiency ($O(N^3)$ time and $O(N^2)$ space), robustness, and extensibility, this method is, even today, still the most popular available. The most widely used packages for RNA secondary structure prediction all contain implementations of the Zuker algorithm [17, 24].

The Zuker algorithm suffers a major shortcoming, however. Because all bonding regions are assumed to be nested, it cannot handle the case of 'pseudoknots'. Pseudoknots are structures in which a bonding pair may have its first base inside

another bonding pair, and the other base outside said bonding pair. In short, it is not properly nested.

These structures are not common, but have been experimentally verified in numerous RNAs [31]. Additionally, pseudoknots also appear to perform useful biological functions. For example, pseudoknots have been shown to allow frame shifting during translation of proteins [19]. In laymans terms, pseudoknots can change the way RNA is read when being translated into protein. Such frame shifting is used extensively by viruses, particularly HIV [19]. Unfortunately, the problem of finding optimal structures with pseudoknots has been shown to be NP-Complete [18].

1.2.2 Pseudoknots

In my investigation I did not consider the prediction of pseudoknotted RNAs as they are uncommon and difficult to model. However, I deem it important to understand how such structures could be integrated into the algorithms presented in this paper. As such, I have included a brisk overview of pseudoknot prediction techniques.

Despite the problem being NP-Complete, in 1999 Rivas & Eddy [26] introduced an ingenious dynamic programming algorithm based on a new thermodynamic model encompassing pseudoknots. Their algorithm could predict a large set of pseudoknot classes using $O(N^6)$ time and $O(N^4)$ memory. They generalised the Zuker method by using a gap matrix to represent multiple regions being considered for bonding, rather than the single continuous region used in the Zuker method. Because of its extreme space and time requirements this algorithm is used only sparingly in practice; other thermodynamic based methods for pseudoknot prediction have been formulated using similar principals. Deogun et al. [8] described an algorithm which could handle a restricted class of pseudoknots (only those containing non-recursive pseudoknots) in $O(N^4)$ time and using $O(N^3)$ space. Shortly after which Reeder & Giegerich [23] presented an algorithm which could predict only simple recursive pseudoknots that met their 'canonization' criteria, and which required $O(N^4)$ time and $O(N^2)$ space. While seemingly restrictive, this did, in fact, predict a large array of pseudoknots accurately.

In recent years different approaches have been explored. In 2010 Sperschneider & Datta introduced DotKnot [28], which improved upon previous algorithms by using probability dot-plot guided heuristics, and an updated energy model, to predict pseudoknots. DotKnot was able to predict pseudoknots more accurately than existing methods with more frugal space and time requirements. This

technique was later refined so that it could predict H-type pseudoknots and intramolecular kissing hairpins [29].

1.3 Accuracy

It is important to test and compare the accuracy of various prediction methods. As such, well established nomenclature and techniques have been developed over the history of RNA structure prediction. These methods are simple but effective. Usually accuracy is determined by comparing predicted structures to known structures. True Positives (TP) is defined as the number of base pairs which appear in both the predicted structure and the actual structure. False Positives (FP) is the number of predicted base pairs not in the true structure [17]. Similarly, False Negatives (FN) is defined as the number of base pairings in the reference structure but not present in the predicted structure [17]. Sensitivity is also called the True Positive Rate (TPR), and can be defined using the previously introduced values. I have given a mathematical definition of the TPR in Equation 1.4.

$$\frac{TP}{TP + FN} \tag{1.4}$$

Precision, also known as Positive Predictive Value (PPV), can also be calculated using these values (see Equation 1.5).

$$\frac{TP}{TP + FP} \tag{1.5}$$

RNAfold is one of the leading RNA folding algorithms, and is made available as part of the Vienna RNA package [17]. At its heart, it is an implementation of the original dynamic programming algorithm first discovered by Zuker, albeit with a more refined energy model. It is an extremely efficient implementation of this algorithm, and is also one of the most accurate in terms of sensitivity and PPV as compared to other implementations of the same algorithm [17]. When Reeder & Giegerich [23] first described their algorithm (implemented in the package pknotsRG) for pseudoknot prediction they compared it to RNAfold, and the algorithm of Rivas & Eddy (implemented in the same package and hereafter referred to as pknotsRE) [26].

Their algorithm generally had higher sensitivity than both other methods, but it is worth noting that pknotsRE was extremely close despite being based on an outdated energy model. This is possibly explained by the fact that it is a more

general, and thus a more powerful algorithm. RNAfold lagged behind pknotsRE and pknotsRG in sensitivity, but executed orders of magnitude faster. Indeed, it has been shown to have excellent accuracy for smaller RNAs containing no pseudoknots while also exhibiting unrivalled computation speed [17].

1.4 Alternative Techniques

1.4.1 Soft Computing

The use of soft computing techniques has also yielded some success in the prediction of RNA secondary structure. Koessler et al. [15] modelled RNA structures as a tree of internal structures, then used artificial neural networks to recognize which of these trees appeared most RNA like. These trees are generated by constructing basic secondary structures and combinatorially merging them together to form many trees, each of which is represented as a vector of simpler trees. This vector is used as the input to the neural network.

This kind of combinatorial blending of RNA stems is also a techniques shared by genetic algorithms. Indeed, this is precisely the starting point of Van Batenburg, Gultyaev, and Pleij [34], who used a simple genetic algorithm to predict secondary structure. Their algorithm started by computing an array of all possible stems; each genome was represented as a binary string where `1` indicates a stem is in the candidate structure, and `0` indicates that it was not. Their genetic algorithm proceeded by seeding the genomes with random bits, then in a series of generation steps performed typical binary mutation, crossover, and breeding, conserving and selectively breeding the fittest solutions. Fitness was defined in their algorithm as the summed total length (number of bonds) of all stems. In an improved version the summed stacking free energy reduction of all stems was used.

Unfortunately they discovered a problem with this approach: the population contained a relatively large portion of zero fitness individuals. This was because many combinations of stems are incompatible with each other, yielding impossible structures. Instead of giving these structures zero fitness, they altered their algorithm slightly to disallow crossover for stems that created an invalid structure. In addition to this, they also explored an important advantage of genetic algorithms for RNA secondary structure prediction: that of kinetic folding. Kinetic folding is the hypothesis that some RNAs, particularly large ones, have a rugged energy landscape, and because of the incremental process of transcription and folding (which happen simultaneously) become stuck in suboptimal areas

during folding [33, 34].

The algorithm of Van Batenburg, Gultyaev, and Pleij simulated this process by limiting the size of stems that could contribute to a genome, and increasing this size over time until the length of the RNA was reached. This single modification to their algorithm yielded the greatest improvement in predictive power. It should also be noted that it could also predict pseudoknots, as the algorithm did not force stems to be nested. Despite this, their approach was still less accurate than the dynamic programming approaches they compared it to. This was likely because their energy model was puerile in comparison rather than because the algorithm was flawed.

Indeed, Wiese, Deschenes, and Hendriks [36] introduced an improved genetic algorithm based on the same principles as that of Van Batenburg, Gultyaev, and Pleij, but instead using an advanced energy model for fitness. They then demonstrated that it outperformed the popular dynamic programming algorithm Mfold [38], which uses a similarly complex model. Wiese, Deschenes, and Hendriks noted that as the length of RNA molecules increased the correlation between lower free energy and accuracy decreased. The concluded that this was due to the incompleteness of our current thermodynamic model of RNA folding. The algorithm of Van Batenburg, Gultyaev, and Pleij avoided this problem by simulating kinetic folding. Such a heuristic was not present in the algorithm of Wiese, Deschenes, and Hendriks.

1.4.2 Context Free Grammars

RNA sequences and their secondary structures can be represented as Context Free Grammars (CFGs). Various production rules output different internal structures (such as hairpin loops, or internal bulges), with the terminals producing the bases A, U, G, and C. This is a fundamentally different approach to those discussed previously, however CFGs can, in fact, use the same thermodynamic energy model. Stochastic Context Free Grammars (SCFGs) can be used to encode the plausibility of production rules, and thus find the most plausible parse tree using a thermodynamic model [27]. In addition, these kinds of algorithms can be trained to incorporate statistical information such as phylogenetic similarity, or machine learned parameters [27].

These kinds of methods have trouble with pseudoknots as non-nested structures are not compatible with CFGs. A notable workaround was applied by Kato, Seki, and Kasami [14], who used multiple context free grammars to model pseudoknots. However, their approach increases the time and space requirements prodigiously.

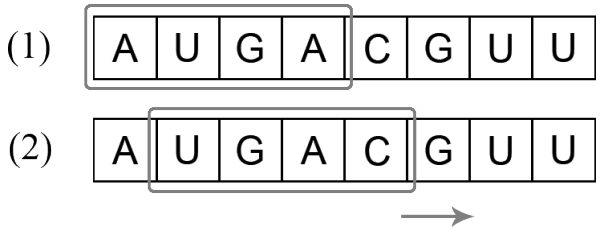

Figure 1.5: Depiction of how sliding windows can explore an RNA sequence.

The greatest strength of context free grammar based approaches is that they can diverge from the use of free energy minimisation entirely. This is advantageous as using a physics based model—such as free energy minimization—requires a large volume of experimentally verified parameters. For this reason, many parameters are often not included in such models because they cannot be quantified empirically. The energy value of multi-branch loops, for example, is not known, and is usually guessed in modern RNA prediction algorithms. Likewise, the interstructural interactions of hairpin loops, bulges, multi-branch loops, and internal loops has not been quantified experimentally and is thus not used as a free energy parameter. CONTRAfold [10] was one of the first SCFG based algorithms to achieve comparable performance to Zuker-like free energy minimization methods. It does away with the notion of free energy minimization altogether, and instead uses a set of trained parameters based on conditional log-linear models. CONTRAfold achieved an average prediction sensitivity higher than RNAfold [17], and also higher than that of Mfold [38], an RNA prediction package similar to RNAfold.

1.5 Locally Optimal Structure Prediction

DNA sequences, unlike typical RNA sequences, are very large indeed. Usually on the order of hundreds of megabytes of data. Such sequences contain DNA subsequences that code for RNAs. Functionally important RNAs typically have a recognizable secondary structure. When searching a large genome for functional RNAs, one can use a sliding window of fixed size to find locally optimal structures (see Figure 1.5). This can be done by running a typical cubic time algorithm like RNAfold or Mfold at every window location. Let L be defined as the chosen window size, and N represent the length of the genome. This leads to a total complexity of $O(NL^3)$. While not prohibitive, this becomes intractable for many genomes which are typically extremely large—millions or billions of bases.

In 2004, Hofacker, Priwitzer, and Stadler [12] provided an excellent insight, and were able to lower this bound to $O(NL^2)$, making it possible to scan large genomes for interesting RNA secondary structure motifs. This was achieved by using the dynamic programming table from the previous step to quickly fill the table for the next window in quadratic time; because consecutive windows overlap, preceding information can be meaningfully used in each forward computational step. As a result it requires only a single table of size $O(L^2)$, and as such its memory complexity is only $O(N + L^2)$.

Later, in 2009, Horesh et al. [13] managed to lower the expected time bound to $O(NL)$ under the assumption that one is folding RNAs that are typical of naturally occurring sequences. This average case time complexity was experimentally verified, and their algorithm was shown to outperform that of Hofacker, Priwitzer, and Stadler.

Clearly good algorithms are available for the folding of consecutive RNA windows. For even modest sized RNAs such algorithms are orders of magnitude faster than holistic secondary structure prediction algorithms. With the major caveat of not actually predicting a complete secondary structure, but only a set of locally optimal structures.

1.6 State of the Art: Global Optimization

Current state of the art algorithms fold RNA in a way that globally maximises their score according to some model. The Zuker algorithm, for example, finds the global minimum free energy configuration. SCFG based algorithms maximise the probability of the final parse tree. This bias is largely due to the 'thermodynamic hypothesis'. Anfinsen [3] presented this hypothesis as the underlying principle for the formation of biologically active proteins. He held that proteins fold into a minimum Gibbs free energy conformation in their typical biological environment; defined as the molecules physiological state: pH, temperature, ion concentration. Furthermore, through natural selection, molecules that are most likely to fold into the correct shape have evolved, and the atomic interactions of such molecules fully determine their final state. This insight has been invaluable for folding proteins and RNAs in silico. Despite this, it has recently become clear that methods for the prediction of RNA secondary structures have hit an upper limit in accuracy.

In her discussion of modern RNA prediction, Rivas [25] unified pseudoknot-free RNA folding algorithms. Her core observation is that all such prediction algorithms contain the same four key components: an architecture, or the production rules of a grammar; a scoring scheme, or how scores are assigned to these

production rules; and the parametrization of the scoring scheme, or the specific values assigned to it. These features are referred to by Rivas, and by myself in the following discussion, as the 'model'. The fourth and final feature is the folding algorithm used to find the best structure given the model. Here Rivas notes that the two dominant folding algorithms are interchangeable. The Cocke-Younger-Kasami (CYK) algorithm used to parse SCFGs, and the MFE algorithm based on the work of Zuker, are isomorphic for the purpose of parsing RNA grammars. Rivas additionally notes that all scoring schemes and parametrizations appear to hit an accuracy upper limit, and that complex, machine learned models are only slightly more accurate than thermodynamic models. In fact, relatively basic grammars with hundreds of parameters seem to perform almost equivalently to those with tens of thousands. While Rivas managed to unify many aspects of RNA prediction she did not recognise that all such algorithms are based on those same assumptions underpinning the thermodynamic hypothesis: they seek to globally maximise a scoring function for the final RNA secondary structure.

I propose that in large RNA molecules, local interactions are stronger than global interactions. As a result RNA molecules will misfold into a global structure made up of locally optimal structures. This is my core hypothesis. If this assumption is correct it follows that there exists a set of 'windows' that when folded using any reasonable model will be more accurate than the corresponding global optimum using the same model. There is already some evidence for this hypothesis. Dawson et al. [7] used variable Kuhn lengths to accurately predict RNAs containing less than 100 nucleotides. Kuhn length is the size of a segment in a polymer chain. It is a simplifying assumption that allows one to treat the entire chain as a sequence of Kuhn segments. In addition, Dawson et al. showed that the energy landscape of these RNAs, when the correct Kuhn length was applied, was funnel shaped. Without such simplifying assumptions the energy landscape of RNAs is notoriously rugged, with in vivo secondary structures often becoming 'trapped' in suboptimal states; this is most apparent in large RNAs [9].

If my hypothesis is supported, I aim to leverage the insights provided to improve the accuracy and speed of RNA secondary structure prediction.

CHAPTER 2

Methodology

2.1 Materials

2.1.1 Environment

All algorithms were implemented and tested using Ubuntu 13.10 running on an Intel i5-3210m processor with four gigabytes of RAM. The GNU C compiler (GCC) version 4.8.2 was used to compile all C and C++ code.

2.1.2 Software

The Vienna RNA package was used as a base for all algorithms presented in this paper. This package contains many useful programs for working with RNA. The RNAfold and RNALfold modules were used in this investigation.

RNAfold is a modern implementation of Zuker's folding algorithm. It predicts the minimum free energy secondary structure of an RNA given the primary sequence. It can also calculate the Boltzmann partition function, producing a matrix of base pair probabilities. Additionally the RNAfold tool can compute the energy of any arbitrary secondary structure, given a corresponding RNA primary sequence, under it's thermodynamic energy model.

The RNALfold module implements a sliding window RNA folding algorithm. It is designed to find all locally optimal secondary structures for an RNA of size n, using a window of fixed size L. RNALfold implements Hofacker, Priwitzer, and Stadler's algorithm [12], and thus uses $O(nL^2)$ time and $O(n + L^2)$ space. I note here that this is not the most optimal algorithm available. As I have discussed in section 1.5, Horesh et al. [13] presented an algorithm that also folds consecutive windows using a similar model. However, they achieved a typical time complexity of $O(nL)$. This algorithm was not used because the implementation give by the authors is based on an older energy model taken from the Mfold

package. RNALfold is based on the same energy model as RNAfold, which has been recently updated [17]. For the sake of accurate comparison to RNAfold, and to ensure a state-of-the-art energy model, RNALfold was used.

Version 2.1.6 of the Vienna RNA package was used. The package was built from the C source code after minor modifications were made to the RNALfold module. The Vienna RNA makefile was used to compile the package. The makefile compiles numerous standalone console applications for Vienna RNA's modules. It also creates a static library called RNAlib. This library was linked at compile time, and used to call RNAfold and RNALfold from the algorithms I shall subsequently describe.

The `Lfold.c` and `Lfold.h` files (which comprise RNALfold) were modified so that, when the algorithm was executed, RNALfold returned a linked list of local secondary structures and their free energy. Before modification it would instead print them to the standard output stream. The modified versions of these files are available in the files associated with this report.

Statistical tests, regression analyses, and plots were generated using GNU Regression, Econometrics and Time-series Library (as called 'gretl') version 1.9.14.

2.1.3 Test Set

The RNA secondary structures used to test algorithms presented in this paper were taken from the RNA STRAND database [2]. The RNA STRAND database is a free-to-use, curated collection of RNA secondary structures taken from various publicly available databases and publications. A subset of RNA structural data was extracted from the database. This subset contained only RNA structures that were marked having been verified using X-Ray Crystallography, or Nuclear Magnetic Resonance imaging. It also comprises only whole RNAs; none of the RNAs used were fragments or subsequences of larger RNA molecules. Finally, no duplicates were allowed in the test set. Hereafter, I shall refer to this collection of RNA secondary structures as the 'testing set'.

2.2 RNA Intervals

Running an algorithm which generates consecutive windows of size L over a RNA primary sequence of size n will produce $n - L$ windows of size L. Each of these windows may contain one or more discrete structural element. For example, a window may contain two disjoint stems. After generating these windows, a post-processing step was executed in which any such disjoint secondary structural

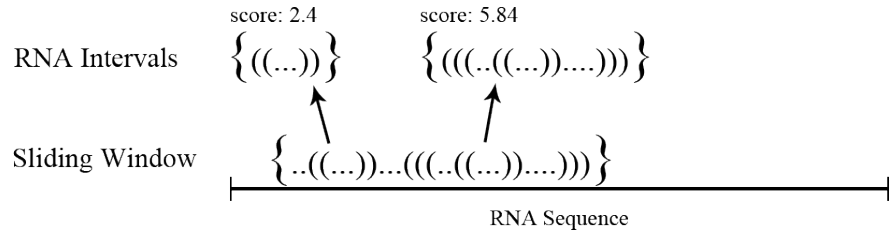

Figure 2.1: The relationship between a sliding window and it's RNA intervals. RNA secondary structures are represented using dot-bracket notation.

elements were split up. While RNALfold does this without modification, it might be necessary to encode this manually using a different implementation.

Once all disjoint structural elements were split up, they were assigned a score. This score was proportional to their free energy reduction. I call this pair of stem and score an 'RNA interval'. Every window generated using a sliding window algorithm was processed into a collection of RNA intervals. A window may contain none or many RNA intervals depending on the density of stems.

2.3 Merging RNA Intervals

Given an RNA sequence, it is easy to generate a set of RNA intervals by running any sliding window algorithm, and storing the RNA intervals generated for each window. Many of these intervals will overlap, or contain fragments of one-another. To construct a plausible and complete secondary structure for the original RNA out of these intervals, a procedure must select a subset of the RNA intervals generated this way. I devised several methods for doing this. The goal of an RNA interval merging algorithm is to produce a valid secondary structure out of a set of RNA intervals which is as accurate (compared to the actual secondary structure) as possible.

Let W be the set of all RNA intervals to choose from, and let the set S represent the set of selected intervals. Also, an interval is compatible with another if they do not overlap and neither interval contains the other (see Figure 2.2. Taking only mutually compatible intervals will result in a valid RNA structure. The way in which these intervals are selected determines the accuracy of the resulting structure.

If any RNA merging algorithm can find RNA secondary structures which are consistently more accurate than those generated by conventional prediction

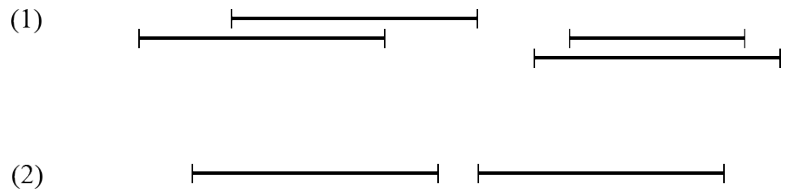

Figure 2.2: Case 1 shows RNA intervals that are not compatible. Case 2 shows an example of compatible intervals.

algorithms, my hypothesis would be supported. This is because such an algorithm builds final structures out of a collection of locally optimal substructures.

2.3.1 Top-Down Selection

In this method, W is sorted by RNA interval size in descending order. The algorithm then examines each element of W in order. If an element of W is compatible with all elements of S, it is added to S.

Algorithm 1 Top-Down Selection

```
1: Let W be the set of RNA intervals
2: Let S = {}
3: SORT(W)
4: for all W as e do
5:     if COMPATIBLE(e, S) then
6:         S = S ∪ e
7:     end if
8: end for
9: return S
```

The algorithm (see Algorithm 1) examines every element of W exactly once, and for each element checks for compatibility with S. The naive way to check for compatibility is to examine every element in S. This would give the algorithm a worst case time complexity of $O(n^2)$, where n is the number of intervals, as all the intervals in W may be mutually compatible. This is a result of the algorithm first checking S of size 0, then 1, up to $n-1$, totalling $(n-1)((n-1)+1)/2$ steps. Additionally it requires $O(n)$ space to store S.

It is possible to lower the time complexity to $O(n \log n)$ by storing the elements of S in an augmented interval tree. Querying this tree finds any elements of S that intersect the interval in question. This query takes $O(\log n)$ time and is

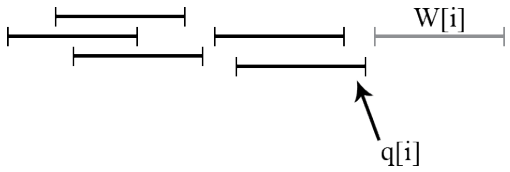

Figure 2.3: The interval stored in $q[i]$ for the RNA interval $W[i]$.

executed n times—once for every element of W. This is optimal, since the `Sort` operation requires $O(n \log n)$ time. Hence, the final worst case time complexity is $O(n \log n)$ and the final worst case space complexity is $O(n)$.

2.3.2 Bottom-Up Selection

This algorithm is identical to Top-Down Selection, except that W is sorted by interval size in ascending order.

2.3.3 Score Selection

This algorithm is identical to Top-Down Selection, except that W is sorted by score in descending order.

2.3.4 Weighted Activity Selection

The weighted activity selection problem is a generalization of the activity selection problem. Given a set of intervals, each of which is assigned a weight, the problem is to find a subset such that none of these intervals touch one-another, and the sum of their weights is maximised. This maps naturally to the problem domain of this investigation. I used this algorithm to find the subset of compatible windows which have the minimum sum free energy, or put another way, maximum score.

The initial sort (see line 1 in Algorithm 2) requires $O(n \log n)$ time. In the following loop (line 4) the algorithm finds the right most element in W such that it is compatible with element i (see Figure 2.3). This can be done using a binary search. Hence the total worst case run time of the loop is also $O(n \log n)$. The final dynamic programming array fill (line 9) requires only $O(n)$ time. As a result the final time complexity of this algorithm is $O(n \log n)$. Additionally, $O(n)$ space is required for the arrays q and dp.

Algorithm 2 Weighted Activity Selection

```
1: Sort W by right end points
2: Let n = |W|
3: Let q = array[1..n]
4: for i = 1 → n do
5:     q[i] = highest index < i which is compatible with i
6: end for
7: Let dp = array[0..n]
8: dp[0] = 0
9: for i = 1 → n do
10:    dp[i] = max(weight[i] + dp[q[i]], dp[i − 1])
11: end for
12: return dp[n]
```

2.3.5 Comparison of RNA Interval Selection Algorithms

The reader should note that all presented selection algorithms have the same $O(n \log n)$ worst case time complexity, and the same auxiliary space complexity of $O(n)$. None of these algorithms have large constant factors. Therefore, my final choice of algorithm was based solely on the accuracy of the computed secondary structure.

In order to test and compare these selection algorithms, all the windows of size five to 500 were precomputed for every RNA in the testing set. This is to say that the sliding window algorithm was run for a window size of five, then size, up to and including size 500 for each RNA. The minimum size of five was chosen because meaningful stems do not form with a size fewer than five nucleotides. The maximum of 500 was due to space and time constraints. Computing these windows took several days, and used 2.5 gigabytes of memory to store.

Each selection algorithm was run using these data as input. For every RNA, the single most accurate window size was recorded. Accuracy was judged as the F-Score, which is the harmonic mean of sensitivity and precision (defined in Section 1.3).

2.4 Prediction Using Windows

In this approach I computed sliding windows using a 'splat' of sizes. Weighted activity selection was then done on the resulting set of computed RNA intervals. I explored several methods for finding good splats. I present here to most successful

technique, which I call '*ab*-splat' prediction. Initially the algorithm started at a small window size, and exponentially increased the scope of the window until a threshold was exceeded. This is explained using pseudocode in Algorithm 3.

Algorithm 3 *ab*-splat

```
1: Let RNA be the primary RNA sequence being folded
2: Let i = a
3: Let S = {}
4: while i ≤ Treshold(RNA) do
5:     S = S ∪ ComputeSlidingWindows(RNA, i)
6:     i = i × b
7: end while
8: return WeightedActivitySelection(S)
```

The `Threshold` function was defined as $\sqrt{RNALength} \times 9.5$. This formula was found using empirical experimentation.

2.4.1 Time Complexity

The time complexity of this algorithm is non-trivial to deduce. The runtime of `ComputeSlidingWindows` is $O(nL^2)$ where L is the window size, and n is the length of the RNA. It is called once for every window size. In addition `WeightedActivitySelection` uses $O(n \log n)$ time, where n is the number of RNA intervals in the set S. I shall now show that *ab*-splat has a worst case time complexity of $O(n^2)$.

$$na^2 + n(ab)^2 + n(ab^2)^2 + n(ab^3)^2 + \cdots + n^2 \tag{2.1}$$

Equation 2.1 shows the number of steps required to run the sliding window algorithm for every window size in *ab*-splat. The second term represents L^2 for exponentially increasing *ab* pairs. Clearly the worse case is where $L = \sqrt{n}$ and hence $L^2 = n$ leading to a largest computational cost of n^2. Factoring out n^2, we get Equation 2.2

$$n^2\left(\frac{na^2}{n^2} + \frac{n(ab)^2}{n^2} + \frac{n(ab^2)^2}{n^2} + \frac{n(ab^3)^2}{n^2} + \cdots + 1\right) \tag{2.2}$$

In the limit to infinity all terms except `1` tend toward zero. Hence we are left with a final time complexity of $O(n^2)$ for computing all required sliding

windows. Now I shall show that this dominates the cost of running weighted activity selection.

Every window size produces $O(n)$ windows of size L. Every such window can contain at most $O(L)$ RNA intervals. Equation 2.3 shows the total number of RNA intervals produced. By factoring out $n^{\frac{3}{2}}$ in a similar manner to Equation 2.2 described above, we see that the number of RNA intervals is $O(n^{\frac{3}{2}})$.

$$na + n(ab) + n(ab^2) + n(ab^3) + \cdots + n^{\frac{3}{2}} \tag{2.3}$$

Weighted activity selection uses $O(n \log n)$ time. As we have $O(n^{\frac{3}{2}})$ windows in the worst case, this leads to a worst case time complexity of $O(n^{\frac{3}{2}} \log n^{\frac{3}{2}})$, which is dominated by $O(n^2)$. Computing the largest sliding window requires only $O(n)$ space, since it uses $O(L^2)$ space in all cases. Weighted activity selection will additionally require only $O(n^{\frac{3}{2}})$ space. As all of these RNA intervals must be stored in S, *ab*-splat will thus use $O(n^{\frac{3}{2}})$ memory in the worst case. In contrast, the Zuker algorithm requires $O(n^3)$ time and $O(n^2)$ space. The computational bottleneck is usually time, however.

2.4.2 Test Procedure

A brute force method was used to test a combinatorial set of values for a and b such that $10 \leq a \leq 30$ and $1.5 \leq b \leq 4.0$. In this method, all integer values for a were attempted, and values for b were generated with a step size of `0.1`. This method was exhaustive, but slow. To speed up computation, precomputed window were used. This meant that the `Threshold` function had to be altered slightly, as only windows up to size 500 were precomputed: $min(\sqrt{RNALength} \times 9.5,\ 500)$.

To test this algorithm, the testing set was randomly partitioned into two sets containing an equal number of RNAs. One of these sets I called the training set, the other I called the validation set. The aforementioned brute force search was done on the training set, and the F-Scores for *ab* pairs was recorded; *ab* pairs with highest F-Scores were deemed the best. These scores were then validated by running the same procedure on the validation set, then attempting to correlate the scores of *ab* pairs between training and validation sets. This indicated if high scores in the training set were related to high scores in the validation set. The correlation also allowed me to determine if results found in the training set were valid.

The best *ab* pair was used to configure the *ab*-splat algorithm, which was then run on the full testing set, and it's F-Scores recorded. These F-Scores were

then compared directly to F-Scores produced by RNAfold, which was also run on the full testing set. In addition to this, the time to execute each algorithm was recorded and compared. Precomputed windows were not used so that the runtime and accuracy could be fairly compared to RNAfold. Because of this, the standard `Threshold` function was used in this scenario.

CHAPTER 3

Results

3.1 RNA Interval Selection

The average score between all RNA interval selection algorithms was compared (see Table 3.1). The highest scoring algorithm appeared to be Top-Down Selection, closely followed by Weighted Activity Selection, then by Bottom-Up Selection. To verify this performance gap, a Wilcoxon Signed-Rank Test was done to compare the recorded F-Scores for corresponding RNA. This test was chosen because the data recorded did not appear to be parametric.

While this test reflected the small difference in averages between Weighted Activity Selection and Top-Down Selection ($z = 1.33066$, see Table 3.2), it was not statistically significant ($p > 0.001$). To further test Top-Down Selection and Weighted Activity Selection, another Wilcoxon Signed-Rank Test was done on only RNAs of length ≥ 300 bases. This revealed a moderate, and statistically significant, difference, indicating that Weighted Activity Selection generally had higher F-Scores for larger RNAs ($p < 0.001, z = 4.55591$). Finally, a Wilcoxon Signed-Rank Test was done between Weighted Activity Selection and Score Selection to check for circular dominance. In accordance with the mean F-score values, Weighted Activity Selection appeared to have higher F-Scores than Score Selection ($p < 0.001, z = 12.0957$).

A Wilcoxon Signed Ranks Test was also used to compare the best recorded F-Scores for all RNA interval selection algorithms, and those recorded for RNAfold (see Table 3.2). All tests were statistically significant ($p < 0.001$). All algorithms but Bottom-Up Selection were shown to outperform RNAfold; Weighted Activity Selection in particular ($z = 13.2082$). Figure 3.1 depicts this performance difference clearly.

Algorithm	Mean	Median
BUS	0.38122	0.35065
SS	0.68373	0.73098
WAS	0.70395	0.75709
TDS	0.71684	0.80000
RNAfold	0.57483	0.60870

Table 3.1: Summary statics of recorded F-Scores for Weighted Activity Selection (WAS), Top-Down Selection (TDS), Bottom-Up Selection (BUS), Score Selection (SS), and RNAfold.

Test Subjects	z-value	Two-tailed p-value
WAS & TDS	-1.33066	0.183301
WAS & TDS (RNA length ≥ 300)	4.55591	2.60801×10^{-6}
WAS & SS	5.94681	2.73418×10^{-9}
BUS & RNAfold	-9.13933	6.28392×10^{-20}
SS & RNAfold	12.0957	0
TDS & RNAfold	13.119	0
WAS & RNAfold	13.2082	0

Table 3.2: Results of Wilcoxon Signed-Rank Testing for F-Scores. Weighted Activity Selection (WAS), Top-Down Selection (TDS), Bottom-Up Selection (BUS), Score Selection (SS), and RNAfold are included.

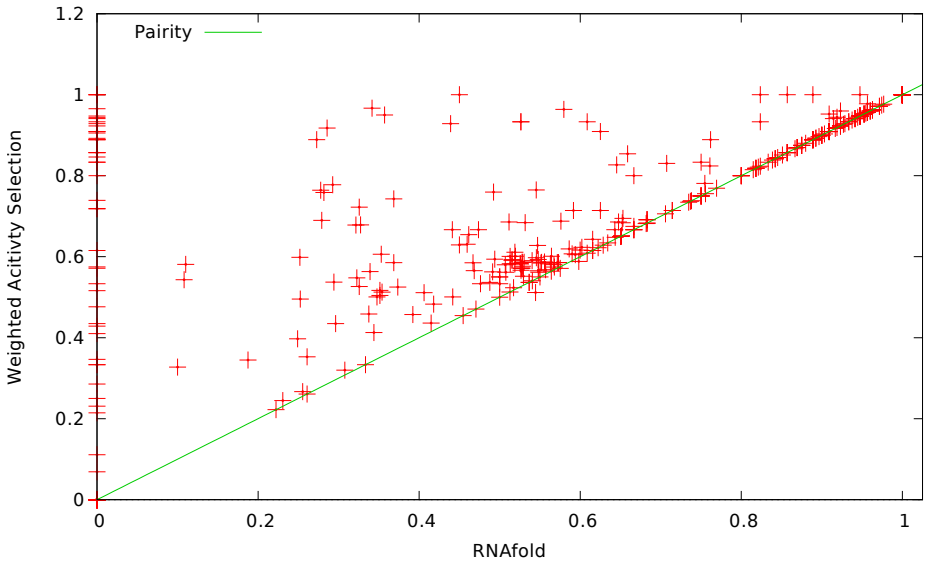

Figure 3.1: F-Scores for matching RNA test cases achieved by Weighted Activity Selection and RNAfold plotted against one-another. The green line represents parity. Any points above the line are cases in which Weighted Activity Selection performed better, points below the line indicate that RNAfold was more accurate. Note that a maximum F-Score is `1.0`.

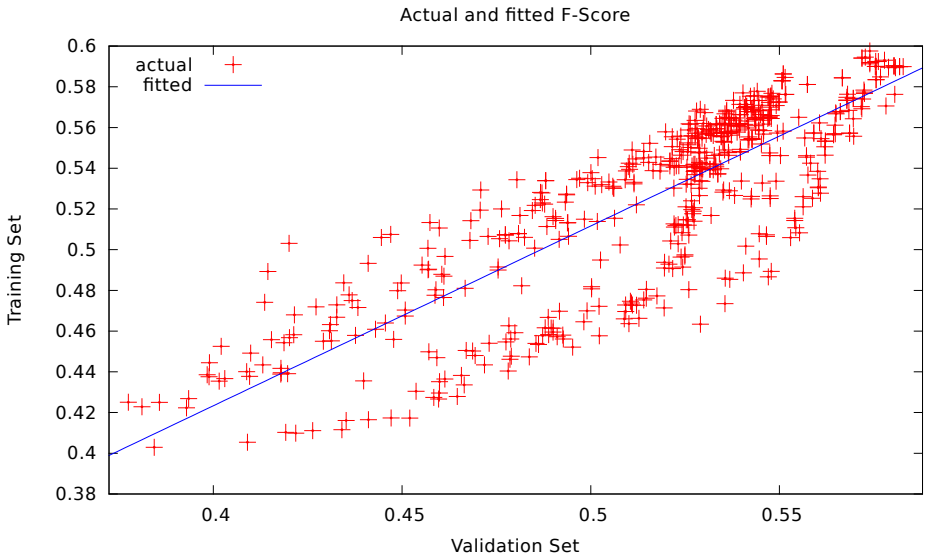

Figure 3.2: Correlation of F-Scores for *ab*-splat in training set versus validation set. R-squared value = 0.702901, p-value < 0.0001. Standard Error of Regression = 0.025032.

3.1.1 Using RNA Intervals For Prediction

An ordinary least squares linear regression was done to determine the correlation between F-scores recorded for *ab*-splat in the training and validation sets. Figure 3.2 depicts the resulting model. A strong correlation was found ($R^2 = 0.703$, $p < 0.001$) for scores in the training set versus scores in the validation set.

The best *ab* value pair found in the training set was $a = 24$, $b = 1.8$. The *ab*-splat algorithm was configured using these values. As the data did not appear to be normally distributed, a Wilcoxon Signed-Rank test was done to compare F-scores for *ab*-splat (with the best *ab* pair) and RNAfold. This test revealed a small difference in the F-scores in the population ($z = 0.1067$), which reflected the small difference in averages (mean RNAfold score = 0.57483, *ab*-splat = 0.58588). However, this discrepancy was not statistically significant ($p = 0.915021$).

Bibliography

[1] Alberts, B., Bray, D., Hopkin, K., Johnson, A., Lewis, J., Raff, M., Roberts, K., and Walter, P. *Essential Cell Biology*, third ed. Garland Science: New York, 2009.

[2] Andronescu, M., Bereg, V., Hoos, H. H., and Condon, A. Rna strand: the rna secondary structure and statistical analysis database. *BMC bioinformatics 9*, 1 (2008), 340.

[3] Anfinsen, C. Principles that govern the protein folding chains. *Science 181* (1973), 233–230.

[4] Beaton, M. J., and Cavalier-Smithf, T. Eukaryotic non-coding dna is functional: evidence from the differential scaling of cryptomonad genomes. *Proceedings of the Royal Society of London. Series B: Biological Sciences 266*, 1433 (1999), 2053–2059.

[5] Condon, A. Problems on rna secondary structure prediction and design. In *Automata, Languages and Programming*. Springer, 2003, pp. 22–32.

[6] Conn, G. L., and Draper, D. E. Rna structure. *Current opinion in structural biology 8*, 3 (1998), 278–285.

[7] Dawson, W., Takai, T., Ito, N., Shimizu, K., and Kawai, G. A new entropy model for rna: part iii, is the folding free energy landscape of rna funnel shaped? *Journal of Nucleic Acids Investigation 4*, 1 (2013).

[8] Deogun, J. S., Donis, R., Komina, O., and Ma, F. Rna secondary structure prediction with simple pseudoknots. In *Proceedings of the second conference on Asia-Pacific bioinformatics-Volume 29* (2004), Australian Computer Society, Inc., pp. 239–246.

[9] Ditzler, M. A., Rueda, D., Mo, J., Håkansson, K., and Walter, N. G. A rugged free energy landscape separates multiple functional rna folds throughout denaturation. *Nucleic acids research 36*, 22 (2008), 7088–7099.

[10] Do, C. B., Woods, D. A., and Batzoglou, S. Contrafold: Rna secondary structure prediction without physics-based models. *Bioinformatics 22*, 14 (2006), e90–e98.

[11] Hofacker, I. L. Rna consensus structure prediction with rnaalifold. In *Comparative Genomics*. Springer, 2008, pp. 527–543.

[12] Hofacker, I. L., Priwitzer, B., and Stadler, P. F. Prediction of locally stable rna secondary structures for genome-wide surveys. *Bioinformatics 20*, 2 (2004), 186–190.

[13] Horesh, Y., Wexler, Y., Lebenthal, I., Ziv-Ukelson, M., and Unger, R. Rnaslider: a faster engine for consecutive windows folding and its application to the analysis of genomic folding asymmetry. *BMC bioinformatics 10*, 1 (2009), 76.

[14] Kato, Y., Seki, H., and Kasami, T. Stochastic multiple context-free grammar for rna pseudoknot modeling. In *Proceedings of the Eighth International Workshop on Tree Adjoining Grammar and Related Formalisms* (2006), Association for Computational Linguistics, pp. 57–64.

[15] Koessler, D. R., Knisley, D. J., Knisley, J., and Haynes, T. A predictive model for secondary rna structure using graph theory and a neural network. *BMC bioinformatics 11*, Suppl 6 (2010), S21.

[16] Leung, Y. Y., Ryvkin, P., Ungar, L. H., Gregory, B. D., and Wang, L.-S. Coral: predicting non-coding rnas from small rna-sequencing data. *Nucleic acids research 41*, 14 (2013), e137–e137.

[17] Lorenz, R., Bernhart, S. H., Zu Siederdissen, C. H., Tafer, H., Flamm, C., Stadler, P. F., Hofacker, I. L., et al. Viennarna package 2.0. *Algorithms for Molecular Biology 6*, 1 (2011), 26.

[18] Lyngsø, R. B., and Pedersen, C. N. Rna pseudoknot prediction in energy-based models. *Journal of computational biology 7*, 3-4 (2000), 409–427.

[19] Namy, O., Moran, S. J., Stuart, D. I., Gilbert, R. J., and Brierley, I. A mechanical explanation of rna pseudoknot function in programmed ribosomal frameshifting. *Nature 441*, 7090 (2006), 244–247.

[20] Neidle, S. *Principles of nucleic acid structure*. Academic Press, 2010.

[21] Nussinov, R., and Jacobson, A. B. Fast algorithm for predicting the secondary structure of single-stranded rna. *Proceedings of the National Academy of Sciences 77*, 11 (1980), 6309–6313.

[22] Nussinov, R., Pieczenik, G., Griggs, J. R., and Kleitman, D. J. Algorithms for loop matchings. *SIAM Journal on Applied mathematics 35*, 1 (1978), 68–82.

[23] Reeder, J., and Giegerich, R. Design, implementation and evaluation of a practical pseudoknot folding algorithm based on thermodynamics. *BMC bioinformatics 5*, 1 (2004), 104.

[24] Reuter, J. S., and Mathews, D. H. Rnastructure: software for rna secondary structure prediction and analysis. *BMC bioinformatics 11*, 1 (2010), 129.

[25] Rivas, E. The four ingredients of single-sequence rna secondary structure prediction. a unifying perspective. *RNA biology 10*, 7 (2013), 1185.

[26] Rivas, E., and Eddy, S. R. A dynamic programming algorithm for rna structure prediction including pseudoknots. *Journal of molecular biology 285*, 5 (1999), 2053–2068.

[27] Rivas, E., Lang, R., and Eddy, S. R. A range of complex probabilistic models for rna secondary structure prediction that includes the nearest-neighbor model and more. *RNA 18*, 2 (2012), 193–212.

[28] Sperschneider, J., and Datta, A. Dotknot: pseudoknot prediction using the probability dot plot under a refined energy model. *Nucleic acids research 38*, 7 (2010), e103–e103.

[29] Sperschneider, J., Datta, A., and Wise, M. J. Heuristic rna pseudoknot prediction including intramolecular kissing hairpins. *RNA 17*, 1 (2011), 27–38.

[30] Studnicka, G. M., Rahn, G. M., Cummings, I. W., and Salser, W. A. Computer method for predicting the secondary structure of single-stranded rna. *Nucleic acids research 5*, 9 (1978), 3365–3388.

[31] Taufer, M., Licon, A., Araiza, R., Mireles, D., Van Batenburg, F., Gultyaev, A. P., and Leung, M.-Y. Pseudobase++: an extension of pseudobase for easy searching, formatting and visualization of pseudoknots. *Nucleic acids research 37*, suppl 1 (2009), D127–D135.

[32] Tinoco Jr, I., and Bustamante, C. How rna folds. *Journal of molecular biology 293*, 2 (1999), 271–281.

[33] Treiber, D. K., and Williamson, J. R. Beyond kinetic traps in rna folding. *Current opinion in structural biology 11*, 3 (2001), 309–314.

[34] Van Batenburg, F., Gultyaev, A. P., and Pleij, C. W. An apl-programmed genetic algorithm for the prediction of rna secondary structure. *Journal of Theoretical Biology 174*, 3 (1995), 269–280.

[35] Watson, J. D., Crick, F. H., et al. Molecular structure of nucleic acids. *Nature 171*, 4356 (1953), 737–738.

[36] Wiese, K. C., Deschenes, A. A., and Hendriks, A. G. Rnapredictan evolutionary algorithm for rna secondary structure prediction. *Computational Biology and Bioinformatics, IEEE/ACM Transactions on 5*, 1 (2008), 25–41.

[37] Xu, Z., Almudevar, A., and Mathews, D. H. Statistical evaluation of improvement in rna secondary structure prediction. *Nucleic acids research 40*, 4 (2012), e26–e26.

[38] Zuker, M. Mfold web server for nucleic acid folding and hybridization prediction. *Nucleic acids research 31*, 13 (2003), 3406–3415.

[39] Zuker, M., and Stiegler, P. Optimal computer folding of large rna sequences using thermodynamics and auxiliary information. *Nucleic acids research 9*, 1 (1981), 133–148.