RNA Folding and Interaction Prediction: A Survey

Syed Ali Ahmed

Graduate Center, City University of New York
New York, NY

November 19, 2015

Abstract

The problem of computationally predicting the structure of a single folding RNA has been studied extensively since the 70’s, while the prediction of the structure of two interacting RNAs has received considerable attention in the past ten years. Despite this, there is considerable room for improvement in the accuracy of these algorithms. In addition, recently there has been a focus on predicting the structure of multiple (more than two) RNA strands. In this paper, we survey various folding and interaction algorithms from the literature. We start with simpler models and then show how accuracy can be increased by making the models more realistic by including observations from physics and biology, while maintaining computational tractability. The application of a wide range of algorithmic topics will be explored, including dynamic programming, stochastic grammars, and approximation. We conclude by describing how these approaches can be applied to the very recent problem of multiple RNA interaction.
## Contents

1 Introduction .......................................................... 3
   1.1 The Structure of RNAs ............................................ 4
   1.2 Secondary Structure .............................................. 5
   1.3 Free Energy and the Nearest Neighbor Model ................. 7
   1.4 Canonical Ensemble and Boltzmann Distribution .............. 9

2 Secondary Structure Prediction: RNA Folding .................. 10
   2.1 Nussinov Base Pair Maximization Algorithm .................. 11
   2.2 Zuker Free Energy Minimization Algorithm ................... 12
   2.3 McCaskill's Partition Function ................................. 14
   2.4 Stochastic Context Free Grammars ............................ 17

3 RNA Interaction Prediction ........................................ 19
   3.1 Ignoring Intramolecular Structures ............................ 20
   3.2 A Model to Maximize Arcs and Bonds .......................... 21
   3.3 Concatenation Approaches ..................................... 23
   3.4 Other MFE approaches with Intramolecular Bonds ............ 24

4 Partition Function based Interaction Algorithms ............... 26
   4.1 Computing Accessibility ........................................ 26
   4.2 Joint Partition Function ........................................ 28
   4.3 From Joint Partition Function to Structure ................... 29
   4.4 Joint Probability of Unpaired Regions ........................ 29
   4.5 Performance .................................................... 31

5 Conclusion and Future Work ..................................... 32
1 Introduction

RNAs (ribonucleic acids), along with DNAs (deoxyribonucleic acids) are important biopolymers that reside inside cells. Nucleic acids are chains of nucleotides, each of which is based on one of four bases: adenine, thymine/uracil, guanine, and cytosine. DNA is more popularly known since it is the carrier of genetic information, but this notion seriously undermines the role of the RNA. The Central Dogma of Biology stipulates that “DNA makes RNA makes protein” [15]. This idea can be elaborated as follows:

Segments of DNAs, which are double stranded chains, are unwound to synthesize RNA via a process known as transcription. The enzyme RNA polymerase carries out this operation, and the result of this transcription are messenger RNA (mRNA) molecules. After an RNA transcript has been created, it is translated by a ribosome to create a protein - a chain of amino acids. Each group of three nucleotides in the RNA codes for a specific amino acid.

While this is true, the role of RNA is much greater than simply being an intermediary in the cycle of life. The majority of RNA transcripts do not code for amino acids, and recent developments have shown that those noncoding regions have several important functions outside of protein biosynthesis. They interact with proteins and other molecules to facilitate or regulate other processes. An example of such an interaction is RNA interference (RNAi), where a small interfering RNA (known as siRNA) can be used to silence a given gene by targeting its messenger RNA: The siRNA binds to the (possibly folded) messenger RNA and triggers a cascade of events that would eventually destroy it (Post Transcriptional Gene Silencing by RNAi) [34].

Some types of RNAs assemble complex molecular machinery such as the spliceosome. In a spliceosome, small nuclear RNA (snRNA) and protein complexes get together to remove introns from pre-mRNA transcripts - a process known as splicing. The spliceosome is a location of intense activity, where RNAs interact with other RNAs as well as proteins [8, 43, 37].

In general, to understand and predict the function of RNAs, their behavior with respect to other molecules has to be understood. This behavior, or function, is closely related to how an RNA folds onto itself or interacts with others. Folding is a process in which the nucleotides on an RNA form bonds with other nucleotides on the same RNA. Since an RNA strand is quite unstable by itself, the folding seems to be a rather necessary process (it is an energetically favorable process). Once folded, some regions (continuous subsequences) on the RNA (particularly loops) play functionally important roles. When a set of RNA nucleotides bond with nucleotides on an other RNA, the process is known as interaction. An interaction complex may contain both types of bonds, and this complex is fundamental to understanding the function of RNAs which work by interacting with other long RNAs. The secondary structure of a folded RNA or RNA interaction complex depicts these bonds in two dimensions.
1 INTRODUCTION

Figure 1: Nucleotides and canonical base pairs. On the left, an adenine (A) molecule binds to a thymine (T) molecule via two hydrogen bonds. On the right, a guanine (G) molecule binds to a cytosine (C) molecule via three hydrogen bonds. Source: https://en.wikipedia.org/wiki/File:Base_pair_AT.svg and Base_pair_GC.svg

1.1 The Structure of RNAs

Nucleic acids are chains of nucleotides, which in turn are molecules that consist of a nitrogenous base and a molecule each of sugar and phosphoric acid. The bases can be divided into two types: purine and pyrimidine. In DNA, the purine bases are adenine and guanine, while the pyrimidines are thymine and cytosine. In RNA, uracil replaces thymine as one of the pyrimidine bases. The bases are all attached in a sequence to a ribose-phosphate backbone. The bases differentiate one nucleotide from another. Because of this, they essentially define the behavior of the nucleotide. When an RNA strand is translated to a protein, each (disjoint) group of 3 consecutive bases maps to an amino acid. Such a group of bases is known as a codon, and each of the 64 possible codons maps to one of 20 amino acids.

In this survey, we are interested in what happens when two nucleotides interact with each other. Two nucleotides interact when their bases form hydrogen bonds with each other. This phenomenon is known as base pairing, and is the basis of the DNA helix and RNA interaction and folding [51]. Base pairing, however, is regulated by some rules. In the DNA helix, we have only guanine-cytosine bonds (pairs) and adenine-thymine bonds (or adenine-uracil in the case of RNA). These two types of pairs are known as Watson-Crick base pairs, named after James Watson and Francis Crick who discovered them. The geometry of nucleotides requires that in most cases, purine bases may bond only with pyrimidine bases; however, in certain conditions, we may have purine-purine and pyrimidine-pyrimidine pairs. In RNAs, one non Watson-Crick base pair is found more often: the uracil-guanine pair. Bases which pair together are called complimentary bases.

This survey is a bioinformatics survey, and it must start leaning towards computer science, away from the biological subtleties which we have been focusing on up till now. We will also be concerned with RNAs, not DNAs. As a first step of abstraction we should point out that these bases can be thought of as symbols of the alphabet \( \Sigma = \{A, C, G, U\} \). This immediately leads to the idea that an RNA can be represented as a string \( S = s_1s_2 \ldots s_n \) of length \( n \) from this alphabet. This representation
1.2 Secondary Structure

Figure 2: The different types of substructures that form in an RNA secondary structure.

has the benefit that many RNA problems can be reduced to string alignment, substring search, and even graph algorithms with labeled vertices. A simplistic folding or interaction prediction algorithm can try to maximize the number of pairs between \{A–U\}, \{C–G\} and \{G–U\}, and more complex algorithms consider interactions between substrings.

1.2 Secondary Structure

With this abstraction, it becomes easy to define the secondary structure. It should be noted that there are at least three ways of representing RNA secondary structures, each with its own advantages. From a computer science perspective, the secondary structure is an outerplanar graph\(^1\) with degree at most three with each vertex labeled by a letter from the alphabet \{A, C, G, U\}, and edges representing base pairs. An isomorphic rearrangement of the vertices may render a visually helpful representation (for example in *arc diagrams* and *circle plots*), but the graph may no longer be planar (this happens only in the case of *pseudo knots*, which are described later). Figure 3 shows the different ways in which secondary structures (of single RNAs) are represented. The function of a folded RNA or an RNA interaction complex is defined by parts of its structure; in corresponding secondary structures, we can identify these parts as subgraphs with a particular arrangement of vertices and edges. These features of a folded RNA or RNA interaction complex are called *secondary structure motifs*.

On a single RNA \(R\), let \(1 \leq i \leq |R|\), be the \(i^{th}\) base on RNA \(R\) in the 5’ → 3’ direction, where \(|R|\) is the length of \(R\). A subsequence or region of \(R\) from base \(i\) to base \(j\) is denoted by \(R[i, j]\), or simply \([i, j]\) if it is clear from the context what \(R\) is. In the case of two RNAs \(R\) and \(S\), let \(R_i\) be the \(i^{th}\) base on RNA \(R\) in the 5’ → 3’ direction, and \(S_j\) be \(j^{th}\) base on RNA \(S\) 3’ → 5’ direction. This is important because RNAs interact in

---

\(^1\)In graph theory, an undirected graph is an outerplanar graph if it can be drawn in the plane without crossings in such a way that all of the vertices belong to the unbounded face of the drawing. That is, no vertex is totally surrounded by edges.
5’ AGUUAGUCAUGACCUUUUGCACCUGGCGUUCUCUGCGAAAGCAGUUAGCGGUACGUGGAUCAUACACCGAUGAGUGAUCUCGGACAACAAGGGGUUGUUGCACGACAUCACUCCGACA 3’

Figure 3: Various representations of RNA secondary structures, using fhlA as an example. From top to bottom: the actual fhlA Sequence; outer planar graph (derived from RNAfold), arc diagram (drawn by our tool), circle plot (drawn by our tool), dot-bracket notation (balanced parentheses expression - generated by RNAfold).
1.3 Free Energy and the Nearest Neighbor Model

The thermodynamic stability of an RNA structure is governed not by the individual basepairs but rather by loops. The stability is measured in terms of the Gibbs free energy \[ \Delta G \]. The Gibbs free energy \( G \) is used to describe the energetics of molecules.
Figure 4: Free energy according to Freier rules (from Durbin)

in a solution, and is defined by

\[ G = H - TS \]  

where \( H \) is the enthalpy of the system (in joules), \( T \) is the temperature (in Kelvin) and \( S \) is the entropy (in joules/kelvin). Since it is not possible to measure the Gibbs free energy of the system, we measure changes in the energy (\( \Delta G = \Delta H - T \Delta S \)). Henceforth we will refer to \( \Delta G \) simply as the free energy. Vis-a-vis RNA folding and interaction, we measure the change in the system when a complex forms from one or more unfolded RNA strands. A favorable chemical process is one where \( \Delta G < 0 \), and therefore a system tries to minimize the free energy. Theoretically, the correct structure is the one with the lowest free energy. However, the structure of an RNA may depend on other factors as well, so this may not always be true. These may include structural features such as coaxial stacking, or the presence of proteins. Often, the structure may be caught in a kinetic trap during the folding process, where it is required to gain free energy before moving to a more favorable, lower energy state \([21, 48]\). Such structures are known as locally optimal.

Each loop in an RNA structure makes some contribution to the free energy, which we will refer to as the free energy of the loop. Stacking pairs and 1-nt bulges have \( \Delta G < 0 \) and stabilize the structure, whereas other loops have \( \Delta G > 0 \). These contributions are additive, and the free energy of a structure is the sum of the free energies of its loops.

\[ E(S) = \sum_{I \in L} E(I) \]  

where \( L \) is the loop decomposition of a structure \( S \). Figure 4 from \([20]\) shows how the components of the (hypothetical) structure contributes a positive or negative energy. The sum of the free energies of all components is -4.6 kcal/mol. The free energies used in the figure are from the Freier nearest neighbor model, which was one of the first ever compilation of such energies.
1.4 Canonical Ensemble and Boltzmann Distribution

<table>
<thead>
<tr>
<th>Stacked pair</th>
<th>5'-AC-3' 3'-UG-5'</th>
<th>5'-CC-3' 3'-GG-5'</th>
<th>5'-AG-3' 3'-UC-5'</th>
</tr>
</thead>
<tbody>
<tr>
<td>Free energy (kcal/mol)</td>
<td>-2.24</td>
<td>-3.26</td>
<td>-2.08</td>
</tr>
</tbody>
</table>

Table 1: Some examples of stacked pairs showing how the free energy varies with different bases involved. Courtesy Waldispühl et al [50].

In nearest the nearest neighbor model the contribution of a loop depends not only on the type of loop but also on the bases involved in the loop and their neighbors. The number of possibilities is large, but independent of the sequence length. These values were measured by Tinoco et al in 1971 [46], Freier et al in 1986 [22] and by Turner and Matthews in 1987, 1999 and 2004 [49, 31, 30], and the model is known as the nearest neighbor model. Physics based models for structure prediction use these values when considering the different types of loops that could appear in a subsequence of RNA. The free energies of structures and their components depend of the temperature of system. Values have been recorded for 37° and a few other temperatures; others can be interpolated.

Table 1 shows some values of a few stacked pairs. These values are computed for all possible small configurations, and for larger configurations they can be approximated. For example, if we have a hairpin loop of \( N > 30 \) nucleotides, the free energy is approximated as \( c_T \ln(N/30) \), where \( c_T \) is a temperature-dependent constant (\( c_{37} = 1.079 \)) [50].

1.4 Canonical Ensemble and Boltzmann Distribution

Quite often, our interest is in computing not the minimum free energy structure, but rather the distribution of all possible structures (known as states in statistical mechanics) in the canonical ensemble. This can be used to compute highly probable structures as well as the probability of any specific substructure such as a certain base pair - this is, in principle, simply the sum of probabilities of all structures that contain this substructure. The probability of a structure is related to its energy, i.e., the probability of observing structure \( S_i \) in equilibrium is

\[
P(S_i) \propto e^{-\frac{\Delta G_i}{RT}}
\]

where \( R = 0.001987 \text{ kcal/mol/K} \) is the Boltzmann constant and \( T \) is the temperature. To compute the probability, we need a normalization constant, which is simply the partition function \( Z \), i.e.,

\[
Z = \sum_{S_i \in \mathcal{S}} e^{-\frac{\Delta G_i}{RT}}
\]

A state \( i \) in the Boltzmann distribution has the probability \( e^{-\Delta G_i/RT}/Z \). Assuming that states follow the Boltzmann distribution has benefits (profound weighting of...
structures in the ensemble, ability to calculate probabilities of substructure); this assumption is quite sound, because the Boltzmann distribution makes the least number of assumptions. For a formal proof using Shannon entropy, see [52].

Eqs. 3 and 4 give us some information about the ensemble. The lower the energy, the higher the probability of observing that state; hence the most probable structure is the MFE structure. Also, since the probability is dependent of the temperature, at very high temperatures all states become equally probable, and at very low temperatures only the MFE structure occurs.

2 Secondary Structure Prediction: RNA Folding

RNA secondary structure prediction research dates back to the 1971, with the first algorithm developed by Tinocho et al [46]. Since then, a host of algorithms have been developed, which can also predict, to some extent, non standard features such as pseudo knots. Often, information about the entire set of possible configurations and their probabilities is required in terms of the Boltzmann weight and partition function, and methods to compute the partition function efficiently have also been developed. Most algorithms can be classified in one of three categories: 1) Free energy minimization, 2) algorithms that compute the partition function or make use of it, and 3) non physics models based on stochastic context free grammars (SCFGs).

The first two categories are physics based models, in that the hyperparameters used in the computation were experimentally determined in vitro (i.e., the Nearest Neighbor model). As discussed earlier, more accurate models assume RNA free energy is the sum of all loops free energies, and contemporary physics based tools are all loop based. Popular RNA structure prediction tools are Mfold, RNAfold and RNAstructure; the MFE prediction is based on the Zuker algorithm [56] and the partition function computation is by way of McCaskill’s algorithm [32]. All implementations run in $O(n^3)$ time with $O(n^2)$ space complexity. RNAfold is part of the ViennaRNA suite [28] which contains many other tools for RNA analysis, including RNAup which computes RNA interaction regions for a pair of RNAs, RNAsubopt [53] which computes a number of structures energetically close to the MFE structure by adapting the inside-outside algorithm vis-a-vis Zuker’s algorithm, RNAeval to compute the energy of a given structure over a given sequence, and RNAcofold which uses a concatenation approach to predict the joint structure of two RNAs. A statistical algorithm to draw suboptimal structures from the canonical ensemble was developed by Ding and Lawrence and implemented in the program Sfold [16].

The non-physics based models we discuss use an approach similar to a loop based model instead of base pair maximization, although the values (probabilities) assigned to stacks and loops are learned from data instead of being adapted from the nearest neighbor model (hence the term ‘non-physics’). Most non-physics based models are based on SCFGs, though some newer tools such as CONTRAfold [19] use more probabilistic methods such as Conditional Log-Likelihood Models, and others such as
2.1 Nussinov Base Pair Maximization Algorithm

[41] use a mix of SCFGs and MFE methods.

A limitation of most physics and non-physics based models is that they cannot predict pseudoknots. The prediction of pseudoknots has been proven to be an NP-complete problem [29]. However, it is possible to predict a restricted set of pseudoknots using polynomial time deterministic algorithms and by sampling methods. Deterministic algorithms have been developed by Rivas et al [40], Lyngsø et al [29], Dirks et al [18], Chen et al [11], and others, while a sampling algorithm based on a simple SCFG was developed by Metzler et al [33]. Deterministic algorithms run in at least $O(n^4)$ time, so there is no known algorithm that matches the time complexity of pseudoknot-free folding algorithms.

To motivate the dynamic programming approach used by most algorithms, we first discuss the Nussinov base pair maximization algorithm, which although not biologically accurate, still forms the basis of all dynamic programming approaches. The Nussinov algorithm seeks to maximize the number of non-crossing base pairs, without associating a score to a single arc or to a stack. After that we discuss Zuker’s algorithm for MFE, followed by McCaskill’s algorithm for the partition function, and some grammar based methods.

2.1 Nussinov Base Pair Maximization Algorithm

Simply maximizing the number of arcs in an RNA reduces to finding a maximum matching in the secondary structure graph. However, this allows non-crossing edges. Now one may argue that this may actually be a benefit because it can predict pseudoknots. However, unguided addition of non-crossing base pairs leads to structures that are biologically meaningless and have “pseudoknots” that are unrealistic. For example, consider the sequence 5’ ACAC...UGUG 3’. It is impossible to redraw the arc diagram into an outer planar graph with straight edges. Another reason why we want avoid crossing edges is that such an algorithm will motivate dynamic programming approaches for energy minimization, where prediction of pseudoknots is NP-complete. Moreover, a base pair is very rarely stable by itself - usually, they are coupled with other basepairs in stacks or bulges, even in pseudoknots.

To prevent crossing arcs, Nussinov et al [38] developed a dynamic programming algorithm that, in principle, recursively computes the best structure of a certain subsequence based on even smaller sequences. The recursive formulation is described below. The score - maximum number of non-crossing base pairs - of the subsequence $i \ldots j$ is stored in a $O(n^2)$ matrix $N$. 

11
2 SECONDARY STRUCTURE PREDICTION: RNA FOLDING

\[
N(i,j) = \begin{cases} 
N(i+1, j-1) + w(i, j) & \text{max}_{i \leq j < k} N(i,k) + N(k+1, j) \quad \text{(bifurcation)} \\
N(i+1, j) & \\
N(i, j-1) & 
\end{cases} 
\]  

(5)

\[
N(i,i) = 0 \quad \forall \quad 1 \leq i \leq n 
\]  

(6)

where

\[
w(i, j) = \begin{cases} 
1 & \text{if } i \text{ and } j \text{ are complementary} \\
0 & \text{otherwise} 
\end{cases} 
\]  

(7)

The first case of the recursion corresponds to inserting an the arc \( i \cdot j \) if one exists, while the second case corresponds to having multiple disjoint arcs. The time and space complexities of this algorithm are \( O(n^3) \) and \( O(n^2) \) respectively, owing to filling out \( O(n^2) \) entries, with a worst case time of \( O(n) \) (in case 2) for each entry. Note that the algorithm can be easily modified to satisfy the constraint that a hairpin loop must have at least 3 unpaired bases, by requiring that \( j - i > 3 \). A variant of the Nussinov algorithm, with the same complexity but shorter formulation, is known as the Nussinov-Jacobson algorithm. It is described below.

\[
N(i,j) = \begin{cases} 
N(i+1, j-1) + w(i, j) & \text{max}_{i \leq j < k} N(i,k) + N(k+1, j) \\
N(i+1, j) & \\
N(i, j-1) & 
\end{cases} 
\]  

(8)

\[
N(i,i) = 0 \quad \forall \quad 1 \leq i \leq n 
\]  

(9)

\[
N(i,i-1) = 0 \quad \forall \quad 2 \leq i \leq n 
\]  

(10)

The Nussinov algorithm has a very naive scoring mechanism, for at least two reasons: 1) the scoring function \( w(.) \) is binary, 2) the algorithm does not take into account neighboring arcs (and therefore stacking energies and internal loops). The scoring function could be modified, so that weights are derived from the distribution of different types of base pairs in large datasets, or so that the score of a base pair depends on the strength of its hydrogen bonds. However, that does not solve the problem of independence of arcs. The next algorithms we discuss are based on the loop-energy model, and do not suffer from these problems.

2.2 Zuker Free Energy Minimization Algorithm

For a given RNA \( R \), the Zuker algorithm [56] finds one optimal secondary structure, i.e., the secondary structure with lowest free energy:

\[
mfe(R) = \sum_{S \in \mathcal{S}} E(S) 
\]  

(11)

where \( \mathcal{S} \) is the set of all possible structures of RNA \( R \). While Zuker’s algorithm does not explicitly consider each structure over \( R \), it does cover all cases by finding the best substructure for all subsequences of \( R \), and by considering all types of loops that can be formed in that subsequence. The algorithm, which is recursive in nature, can be
implemented with dynamic programming. It will be obvious that Zuker’s algorithm has the major components of Nussinov’s, specially with respect to the bifurcation technique. However, since Zuker’s algorithm computes loop energies, the computation involves more bookkeeping. It does so by keeping three matrices, which assume different things about the subsequence they represent.

The base cases for Zuker’s algorithm are formulated by the observation that there is no substructure that involves less than 5 nucleotides. This follows our earlier constraint that hairpins need at least 3 unpaired bases. If \( j - i = 4 \), then \( i \) and \( j \) may close a hairpin loop, but no other types of loop. Thus, the first step is to compute the energies of all "pentanucleotide" substructures. To let the algorithm compute loop energies, we need to consider cases where a loop is closed by bases \( i \) and \( j \), for all \( i \leq j - 4 \). In other words, we require that such \( i \) and \( j \) form an arc. Since this condition is required only to form loops, we keep two matrices \( V \) and \( W \). Let their entries be defined as follows:

\[
V(i, j) = \min \text{ free energy of all possible substructures formed from } [i, j] \\
\text{where } (1 \leq i < j \leq n) \text{ and } i \text{ and } j \text{ form an arc.} \tag{12}
\]

\[
W(i, j) = \min \text{ free energy of all possible substructures formed from } [i, j] \\
\text{where } (1 \leq i < j \leq n) \tag{13}
\]

We set \( V(i, j) = \infty \) if \( i \) and \( j \) cannot form an arc (i.e., they are not complementary). To compute \( V(i, j) \), we need to consider the different types of loops that \( i \) and \( j \) can close. We consider all possibilities, and then pick the loop that is optimal. As described in section 1.2 there are four types - stack, hairpin, interior, or multi loop. For hairpin loops, the formulation is very simple: \( V(i, j) = E^H(i, j) \), because there are no substructures in the interval \([i + 1, j - 1]\). When a base pair \( i \cdot j \) closes a stack, the energy of \([i, j]\) is the energy of this stack plus the energy of all loops between \( i + 1 \) and \( j - 1 \). Therefore, \( V(i, j) = E^S(i, j, i + 1, j + 1) + V(i + 1, j - 1) \). We can use \( V(i + 1, j - 1) \) instead of \( W(i + 1, j - 1) \) because we assume that \( i + 1 \) and \( j - 1 \) form an arc, since we also assume it is part of a stack. The computation for interior loops is similar, except that we need to take into account all possible positions of the adjacent arc \( i' \cdot j' \) and select the one with the lowest energy. Therefore, \( V(i, j) = \min E^L(i, j, i', j') + V(i', j'), i < i' < j' < j \).

The last case, that of closing a multi loop, requires us to first compute multi loop energies. To do that, define the entries of the matrix \( M \) as follows:

\[
M(i, j) = \min \text{ free energy of all possible substructures formed from } [i, j] \text{ where } [i, j] \text{ is part of a multiloop, } 1 \leq i < j \leq n \\
\text{but } i \text{ and } j \text{ do not necessarily form an arc.} \tag{14}
\]

Having the condition that \( i \) and \( j \) form an arcs make for a simply recursive decomposition. There are several variables that our recursions need to take into account to compute \( M(i, j) \). For every unpaired base we penalize \( c \), and for every stem (called branches of the multiloop), we penalize \( b \). Also note that a multi loop with \( k > 2 \)
stems can be described as “one stem and \((k - 1)\)-stem multi loop”. This allows us to recursively define \(M(i, j)\):

\[
M(i, j) = \min \begin{cases} 
M(i + 1, j) + c \\
M(i, j - 1) + c \\
\min_{i < k < j} M(i, k) + W(K + 1, j) \\
V(i, j) + b
\end{cases}
\] (15)

The first 2 cases recursively reduce \(S_{i,j}\) to one helix, while penalizing unpaired bases. The third case ensures there are at least two helices, and the last case closes a helix as well as counts one. We can now finish our definition of \(V\). When considering \(i, j\) closing a multi loop, we split the multi loop into sub multi loops and pick the optimal split. We also penalize multi loop formation with \(a\).

\[
V(i, j) = \min \begin{cases} 
E^H(i, j) \\
E^S(i, j, i + 1, j + 1) + V(i + 1, j - 1) \\
\min_{i < i' < j} E^I(i, j, i', j') + V(i', j') \\
\min_{i < k < j} M(i + 1, k) + M(k + 1, j - 1) + a
\end{cases}
\] (16)

The matrix \(V\) only computes energies of optimal substructures if \(i\) and \(j\) form an arc, whereas \(W\) contains the actual optimal substructure. We can define \(W\) again using Nussinov’s algorithm as a basis - either \(i\) is unpaired, \(j\) is unpaired, both are unpaired, both are paired together, or both are paired but with other bases. Therefore,

\[
W(i, j) = \min \begin{cases} 
W(i + 1, j) \\
W(i, j - 1) \\
V(i, j) \\
W(i, i') + W(i' + 1, j)
\end{cases}
\] (17)

The time complexity of this algorithm is \(O(n^4)\), owing to expensive calculations of interior loop energies in \(V(i, j)\). We can can bound the length of the sides of the interior loop by a constant, and thus bring the time complexity down to \(O(n^3)\). The space complexity is \(O(n^2)\) for each matrix. However, the complexity of \(W\) can be reduced to linear if we instead keep a vector, whose entries represent the optimal structures \(S_{1,i}\).

It should be noted that the substructures considered by Zuker’s algorithm are not disjoint. However, since this is a minimization problem, it does not affect the result.

### 2.3 McCaskill’s Partition Function

As discussed in section 1.4, we need to compute the partition function to determine probabilities of states and substructures. The partition function \(Q\) of RNA sequence
2.3 McCaskill’s Partition Function

\( R \) is defined by

\[
Z(R) = ZQ = \sum_{S \in \mathcal{S}} e^{-\left[ F(S) / RT \right]} \quad (18)
\]

\[
= \sum_{S \in \mathcal{S}} \prod_{L \in S} e^{-\left[ F(L) / RT \right]} \quad (19)
\]

where \( \mathcal{S} \) is the set of all possible structures of RNA \( R \). In other words, the partition function is the sum of the free energy of all possible structures. The problem of enumerating all possible structures and adding their energies grows exponentially with the sequence length, hence a more efficient method has to be applied.

In section 2.2 we already saw a method to decompose a given RNA structure into substructures in polynomial time. However, the decomposition is ambiguous and the substructures are not disjoint, which is acceptable for a minimization problem but not for a summation problem, since we will over count very easily. While the general framework remains the same, McCaskill’s algorithm [32] provides a decomposition that is unambiguous and disjoint.

We denote the matrices used in this computation by \( Q \) or a variant. As a first step, we compute the restricted partition function \( Q_{ij}^b \), where we sum over all structures of the subsequence \( R[i,j] \) in which \( i \) forms an arc with \( j \) (i.e., close a loop). The loop closed by \( i \) \( \cdot \) \( j \) could be of any type: hair pin, stack, interior, or a multiple loop of degree \( k \). We sum over all possible types to obtain:

\[
Q_{ij}^b = \sum_{(h,l) \in L} e^{-\left[ F_{L} / \beta T \right]} \prod_{i \leq h < l \leq j} Q_{ih}^b \quad \prod_{\text{degree}(\text{loop})} e^{-\left[ F(\text{kth branch}) / \beta T \right]} \quad (20)
\]

where \( L = \emptyset \) in the case of a hairpin loop, \( \{(h_1, l_1)\} \) in case of stack or interior loops, and \( \{(h_t, l_t)\}^{k}_{t=1} \) in case of a multi loop of degree \( k - 1 \). \(^3\)

It is now easy to compute the partition for sequence \( S_{ij} \) without the constraint that \( i \) and \( j \) form a base pair:

\[
Q_{ij} = 1.0 + \sum_{L} e^{-\left[ F_{L} / \beta T \right]} \prod_{i \leq h < l \leq j} Q_{ih}^b \quad (21)
\]

where \( 1.0 \) is the contribution of the empty structure, and \( Q_{ii} = Q_{i+1,i} = 1.0 \). Computing bottom up, we can arrive at \( Q_{1N} = Z \). However eq. (20) is intractable since it is exponential in the degree of the multi loops. To avoid this, we can use the same linearity assumption made by Zuker et al.: \( F_{L} = a + b(k - 1) + cu \) when \( k > 2 \). Thus

\(^2\)For a detailed description of converting Zuker’s decomposition to McCaskill’s disjoint decomposition, see http://math.mit.edu/classes/18.417/Slides/mccaskill.pdf.

\(^3\)It is fairly straightforward to derive this recursive equation.
we can rewrite eq. (20) as

\[
Q_{ij}^b = e^{-E^b(i,j)/\beta T} + \sum_{i<h<l<j} e^{-E^b(i,j,h,l)/\beta T} Q_{hl}^b
\]

(22)

\[
+ \sum_{i<h<l<j} Q_{i+1,h-1}^m Q_{hl}^b e^{-(a+b+c(j-l-1))/\beta T}
\]

(23)

where

\[
Q_{ij}^m = \sum_{i<h<l<j} \left( e^{-(c(h-i-1))/\beta T} + Q_{i,h-1}^m \right)
\]

(24)

\[
\times Q_{hl}^b e^{-(b+c(j-l-1))/\beta T}
\]

(25)

with \(Q_{ii}^m = 0\) and \(Q_{i+1,i}^m = 0\). The complexity of computing \(Q_{1N}\) now becomes \(O(n^4)\). This can be reduced to \(O(n^3)\) by limiting the size of internal loops, in a similar fashion to Zuker’s algorithm. For another, more realistic, approach to reducing to \(O(n^3)\), see [32].

The probability of a structure \(S\) can now be computed using this partition function:

\[
P(S) = \frac{e^{-E(S)/RT}}{Q}
\]

(26)

A useful result property of the above equation is that the probability of each structure is linearly proportional to the energy factor \(e^{-E(S)/RT}\). This is useful when one needs to sample structures from this prohibitively large distribution.

Although the probability of a structure is a useful quantity, we are also interested in the probability of individual base pairs that may exist for a given RNA sequence. We will call this quantity \(P(h \cdot l)\), and it simply the sum of the probabilities of all structures that contain the base pair \(h \cdot l\), i.e., \(P(h \cdot l) = \sum_{S \ni h \cdot l} P(S)\). To compute this with dynamic programming, we can define it as decomposition into three cases.

In the first case, there is no arc exterior to \(h \cdot l\). Thus we only need to consider structures on the left of, right of, and within \(h \cdot l\):

\[
P_{\text{ext}}(h \cdot l) = \frac{Q_{1h-1}^b Q_{hl}^b Q_{l+1,N}^i}{Q_{1N}}
\]

(27)

In the second case, the base pair \(h \cdot l\) is the interior arc of a stack of interior loop. We can sum over all possible closing arcs \(i \cdot j\).

\[
P_{\text{int}}(h \cdot l) = \sum_{i<h,l>j} P_{ij} \times P(h \cdot l|i \cdot j) \times e^{-(E^l(i,j,h,l))/\beta T}
\]

(28)

\[
= \sum_{i<h,l>j} P_{ij} \times \frac{Q_{hl}^b}{Q_{ij}^b} \times e^{-(E^l(i,j,h,l))/\beta T}
\]

(29)
In the third case, we consider \( h \cdot l \) to be part of a multi loop closed by arc \( i \cdot j \). Conditioning in the same way as the previous equation, we get

\[
P_{\text{multi}}(h \cdot l) = \sum_{i<h,l>j} P_{ij} \times P(h \cdot l|i \cdot j) \times e^{-(a+b)/\beta T} \times (e^{-(j-l-1)c/\beta T} Q_{i+1,h-1}^m + e^{-(h-i-1)c/\beta T} Q_{i+1,j-1}^m) + Q_{i+1,h-1}^m Q_{i+1,j-1}^m
\]

The sum of these three probabilities is the probability that \( h \cdot l \) exists.

2.4 Stochastic Context Free Grammars

In this section we discuss a non physics-based model - stochastic (or probabilistic) context free grammars - for RNA structure prediction. These models are called non physics-based since their parameters are learnt from the data (known sequences and structures) instead of being determined experimentally.

Grampars and the algorithms to parse them have been used extensively in natural language processing. For example, the structure of semantics of a sentence depend on how it can be derived from a given grammar, and these semantics define its structure. It is only natural that grammars have found applicability in RNA folding and interaction, since an RNA sequence is like a sentence of nucleotides instead of words, and the semantic structure determine the secondary structure.

Given a set of symbols \( \Sigma \) a context free language \( L \subseteq \Sigma^* \) is a set of strings that can be generated using a grammar \( G \) corresponding to \( L \). A stochastic context free grammar \( G = (N, \Sigma, R, S, P) \) has the set of symbols \( \Sigma \cup N \) and a set of production rules \( R \). \( N \) is a set of nonterminal symbols used in the derivation of strings in \( L \). How each non-terminal expands to a string of symbols in \( \Sigma \) is determined by productions or rules from the set \( R \). Productions ins context free languages take the form

\[
N \rightarrow (\Sigma \cup N)^*
\]

\( P \) is the set of probabilities associated with each rule. Note that the sum of all productions which have the same left hand side is 1. A parse tree for a string is a description of which productions were applied in what order to derive that string. The probability of generating a string with a certain parse tree or derivation is the product of each production in that parse tree. Note that if the grammar is ambiguous, there may be multiple parse trees for a string.

When our interest is in determining a structure instead of generating a sequence, we give additional meaning to each production. Consider the following simple RNA grammar from [19]:

\[
S \rightarrow aSu | uSa | cSg | gSc | gSu | uSg | aS | cS | gS | uS | \epsilon
\]

For rules of the form \( S \rightarrow cSd \), we say that \( c \) and \( d \) form a basepair. When a string
is generated, we know which productions were applied and thus we can generate a structure. For example, the string \( x = agucu \) can be generated by the following productions: 

\[
S \rightarrow aS \\
S \rightarrow gS \\
S \rightarrow uS \\
S \rightarrow \epsilon
\]

In the first step, we pair \( a \) and \( u \), then pair \( g \) and \( c \), then add an unpaired \( u \). The secondary structure in this case would be \( ((.)) \) in dot bracket notation. This grammar is syntactically ambiguous, since the same string \( x \) can be generated using the productions 

\[
S \rightarrow aS \\
S \rightarrow agS \\
S \rightarrow aguS \\
S \rightarrow agucS \\
S \rightarrow agucuS \\
S \rightarrow agucu
\]

because in each application we added an unpaired nucleotide. However, the grammar is semantically unambiguous, because each secondary structure \( y \) for a given string \( x \) has only one derivation.

The probability of generating this string and secondary structure is 

\[
P(S \rightarrow aSu) \times P(S \rightarrow gSc) \times P(S \rightarrow uS) \times P(S \rightarrow \epsilon)
\]

Thus a semantically unambiguous \(^4\) SCFG models a joint probability distribution \( P(x, y) \) where \( x \) is an RNA sequence and \( y \) the secondary structure, or equivalently, the parse tree that generated that sequence and structure. It follows that 

\[
P(y|x) = \frac{P(x,y)}{\sum_{y' \in \Omega(x)} P(x,y')}
\]

where \( \Omega(x) \) is the space of all possible parses of \( x \).

Using a grammar based model, we can determine the most likely structure of a sequence \( x \), which is simply the parse tree with the highest probability. This is a well studied problem in the general sense. A stochastic version of the Cocke?Younger?Kasami algorithm (CYK) takes as input a grammar in Chomsky normal form (CNF) and a sequence (sentence, RNA sequence, etc), and outputs the most likely parse tree in \( O(n^3|G|) \) where \( n \) is the length of the sequence and \( |G| \) is the size of the grammar. A grammar in CNF has rules only of the following three types: 

\[
X \rightarrow AB, X \rightarrow a, X \rightarrow \epsilon
\]

The condition that the grammar has to be in CNF does not preclude the possibility of using RNA grammars: any context sensitive grammar of size \( |G'| \) can be converted into a CNF with grammar of size at most \( |G'|^2 \).

The essence of the CYK algorithm is the following. We maintain a three dimensional array of nonnegative reals \( P[i, j, N] \) where \( i, j \) are the starting and ending indices of a subsequence, and \( N \) is a nonterminal from the grammar. The value of \( P[i, j, N] \) is the probability of the best parse subtree rooted at the nonterminal \( N \) that, given some sequence \( s = s_1s_2...s_n \), derives the subsequence \( s_i...s_j \). Since productions are of the form \( X \rightarrow AB \), this can be recursively split into two rules, dynamic program can solve this recursion.

As with physics based models, we are interested in both the best structure and a distribution of structures and probabilities of base pairs. This is indeed possible to compute using the Inside-Outside algorithm, which is analogous to the Forward-Backward al-
algorithm. The inside algorithm computes the inside term \( \alpha(i, j, N) = P(A \Rightarrow s_i \ldots s_j) \), which is the probability that non terminal \( A \) derives the subsequence \( s_i \ldots s_j \) after some number of steps. Therefore, \( \alpha(1, n, S) = P(S \Rightarrow s_1 \ldots s_n) \) is the probability of generating the entire sequence using this grammar. The term \( \alpha(1, n, S) \) is also equivalent to the sum of the probability of each parse tree of the given sequence. Note that inside terms are not conditioned on subsequences \( s_1 \ldots s_{i-1} \) and \( s_{j+1} \ldots s_n \). The outside variables \( \beta(i, j, N) \), on the other hand, compute the probability of generating everything outside \( s_i \ldots s_j \) given that \( N \) generates \( s_i \ldots s_j \), i.e. \( \beta(i, j, N) = P(S \Rightarrow s_1 \ldots s_{i-1}N s_{j+1} \ldots s_n) \).

The grammars used for computing RNA secondary structures are usually more complex than the one described earlier. For example, the KH grammar

\[
\begin{align*}
S & \rightarrow LS | L \\
L & \rightarrow x | xFy \\
F & \rightarrow x | xFy | LS
\end{align*}
\]

(36) (37) (38)

generates loops \( L \) and stems \( F \). Note that the above is condensed representation where \( x \) and \( y \) are placeholders for \( a, c, g, u \), and each rule has a probability derived from known sequences. Using the inside and outside probabilities for this grammar, we can compute the probability that bases \( i \) and \( j \) form a base pair in the secondary structure:

\[
P(i \bullet j) = \frac{\beta(i, j, L)P(L \Rightarrow x_iFy_j)\alpha(i + 1, j - 1, F)}{\alpha(1, n, S)} + \frac{\beta(i, j, F)P(F \Rightarrow x_iFy_j)\alpha(i + 1, j - 1, F)}{\alpha(1, n, S)}
\]

(39)

This follows from the fact there are two productions that can generate a base pair \( i \bullet j, L \rightarrow x_iFy_j \) and \( F \rightarrow x_iFy_j \). Therefore, we have two cases. In the first case, we consider the probability that the \( S \) derives \( s_1 \ldots s_{i-1}L s_{j+1} \ldots s_n \), then the production \( L \rightarrow x_iFy_j \) adds a base pair, and then the remaining subsequences are generated by \( F \) (because the production’s right hand side has the non terminal \( F \)). The second case is handled similarly.

3 RNA Interaction Prediction

In section 1, we motivated the biological problem of RNA-RNA interaction. In this section, we describe the computational aspects of this problem in detail, and then offer some hardness results, approximation algorithms and exact algorithms operating under certain assumptions. Recall that in the case of RNA-RNA interaction, we have two types of nucleotide-nucleotide interactions: those within the same RNA, and across two or more RNAs. We call the second type bonds. A nucleotide may only be involved in one interaction, so it is either unpaired, it forms an arcs, or it forms a bond.
Like the RNA folding problem, RNA interaction prediction has been studied in detail, but without outstanding results. This is obvious because the interaction version is harder than the folding problem. RNA interaction prediction is a relatively new problem compared to RNA folding, only a decade old as opposed to 40 years of research in RNA folding. Moreover, results vary depending on which RNAs their techniques are applied to. For example, small interfering RNAs (siRNAs) silence gene expression by binding to a longer messenger RNA (mRNA); the internal structure of the mRNA may prevent the siRNA from directly binding to any complimentary site, and thus algorithms have to consider the feasibility of binding via the overall reaction energy. RNAs such as CopA and their anti-sense targets like CopT both form internal structure during interaction; the internal structure is important because it can inhibit the formation of certain intermolecular bonds, thus giving rise to a kissing loop that is functionally important [27, 26]. The OxyS-fhlA interaction complex has similar characteristics [6]. A completely different type of interaction takes place in the spliceosome, which is a complex molecular machine consisting of the RNAs U1, U2, U4, U5 and U6. These RNAs interact (with often more just one other RNA) to remove introns from a transcribed pre-MRNA - this process is known as splicing. Algorithms have been developed in [54, 44] and others to study U6-U2 interaction and in [1, 35] to study multiple RNA interactions in the spliceosome.

We will start by discussing models in which we ignore arcs, then proceed towards models which consider both intermolecular and intramolecular bonds but only to maximize base pairs, and finally discuss more powerful models that are loop-based but for interaction.

### 3.1 Ignoring Intramolecular Structures

In the past, attempts have been made to ignore the intramolecular structure and focus the intermolecular bonds, using approaches more sophisticated than simply maximizing complementary basepairs. If complementarity is the only criteria, then BLAST [3] is sufficient for long complementary sequences in a target RNA. Tools, such as GUUGle [23] and TargetRNA [47], have also been developed that give scores to (independent) basepairs based on the frequency of that pair in the genome of interest, and other parameters.

Sophisticated algorithms (which still ignore intramolecular bonds) include methods to drop the independence assumptions and compute the free energies of loops and stacks. This approach uses the same loop and stack model as Zuker and McCaskill algorithms, and the energy parameters are used from the nearest neighbor model. The three possible substructures are stacks, bulges or interior loops. As in the case of Zuker/McCaskill, long loops are ignored since they are energetically unfavorable and increase the time complexity. However, we can consider this approach to be a more restricted form of Zuker/McCaskill algorithms because a nucleotide is only permitted to form bonds with nucleotides on the other RNA. These approaches were implemented in RNAHybrid [39] and RNAplex [45]. RNAplex differs in its treatment of
interior loops, using an affine gap penalty instead of length-dependent recursions. It also gives some weight (though implicitly) to the accessibility of the interaction site by introducing a penalty proportional to the length of the region under consideration, although this may be not be correct if the interval does not form bonds.

While more accurate than the approaches described earlier, these techniques (including RNAplex) do not explicitly consider the accessibility of binding sites. This has some disadvantages. Firstly, these structures are not realistic. The predicted window with the minimum energy may lie in a region with strong intramolecular bonds, and this would make this structure very unlikely. Secondly, the predicted window may be too long. It is indeed possible that there are enough stacking base pairs to predict a very long window, which in reality might also have intramolecular base pairs.

As the reader will see later on, the tool RNAup calculates interaction energy in the same way but also considers the probability of such a region being unpaired, thus taking care of the first problem of this method. More sophisticated and accurate methods involve computing three types of partitions functions: one where only intramolecular base pairs are allowed, one where only intermolecular ones are allowed, and one for the joint structure.

### 3.2 A Model to Maximize Arcs and Bonds

To motivate algorithms that consider inter- and intra-molecular basepairs, we begin by introducing a simplified model that tries to maximize the number of arcs and bonds between compatible base pairs. There are two suitable ways in which we can diagrammatically represent RNA-RNA interactions: 1) outer planar graph similar to that for RNA folding, 2) arc and bond diagrams but for two RNAs. Using the second representation allows us to better visualize this as a computational problem. Figure 5 shows an interaction graph, where each edge represents a possible interaction. Figure X (ADD) shows a simple interaction between two RNAs.

**Definition 3.1.**\(^\text{[34]}\) Simplified RNA-RNAi problem: Given an RNA-RNA interaction graph \((V, E)\), where the nodes of \(V\) are partitioned into two ordered sets \(X = \{x_1, \ldots, x_m\}\)
and \( Y = \{ y_1, \ldots, y_n \} \), and every edge \( e \in E \) has a weight \( w(e) \in \mathbb{Q} \), find a set of node disjoint edges \( S \) that maximized \( \sum_{e \in S} w(e) \) such that (intersection is avoided):

- If \((x_i, x_j) \in S \) and \((x_k, x_l) \in S\), the NOT \( i < k < j < l \).
- If \((y_i, y_j) \in S \) and \((y_k, y_l) \in S\), the NOT \( i < k < j < l \).
- If \((x_i, x_j) \in S \) and \((y_k, y_l) \in S\), the NOT \( i < k \) and \( j > l \).

This problem can be further simplified along the lines of Nussinov’s algorithm by setting \( w(e) = 1 \) for all \( e \in E \). Note that this model is also referred to as the base-pair energy model. It must be stressed that the most accurate model for RNA-RNA interaction is still the loop based model, with more types of loops defined within the context of interaction. However, the diagrammatic representation will remain the arc diagram.

The hardness of RNA-RNA Interaction models has been studied before by Mneimneh [34] and Alkan et al [2]:

**Theorem 3.1.** The decision version of RNA-RNA Interaction is NP-complete (even when uniformly weighted).

**Approximation for Basepair Model**

In [34], Mneimneh presents a polynomial time algorithm that achieves \( \frac{2}{3} \) approximation to the base pair version of the RNA-RNA interaction problem. The algorithm generates three structures; it is described as follows:

1. Optimally solve RNA-RNAi while ignoring binding edges, i.e., fold both RNAs independently.
2. Optimally solve RNA-RNAi while ignoring the folding edges for RNA\(_2\).
3. Optimally solve RNA-RNAi while ignoring the folding edges for RNA\(_1\).

In case 1, one may optionally align the remaining unpaired bases. In cases 2 and 3, one may optionally fold the non interacting bases of the earlier ignored RNA. The highest scoring structure amongst these is guaranteed to have a score of \( \frac{2}{3} \) OPT, where OPT is the score of the optimal RNA-RNAi solution. Let the scores of the above structures be \( w_1, w_2, w_3 \) respectively. We now prove the \( \frac{2}{3} \)-approximation claim.

**Lemma 3.1.** \( \max(w_1, w_2, w_3) \geq \frac{2}{3} OPT \)

**Proof.** Consider the optional RNA-RNAi solution, OPT. Let \( A \) be the score of the folding edges of RNA\(_1\) in OPT, i.e., edges of the form \((x_i, x_j)\). Let \( B \) be the score of the interaction edges between RNA\(_1\) and RNA\(_2\) in OPT, i.e., edges of the form \((x_i, y_j)\), and \( C \) be the score of the folding edges of RNA\(_2\) in OPT, i.e., edges of the form \((y_i, y_j)\). Then OPT = \( A + B + C \). By optimality of the structures, \( w_1 \geq A + C, w_2 \geq A + B, \) and \( w_3 \geq B + C \). Then \( 3 \max(w_1, w_2, w_3) \geq w_1 + w_2 + w_3 \geq A + C + A + B + B + C = 2(A + B + C) = 2OPT \).

This proves the \( \frac{2}{3} \)-approximation. However, we still need to prove that the above algorithm is polynomial time. Option 1 of the algorithm requires folding both RNAs independently. Algorithms for folding have \( O(n^3) \) time complexity. It remains to show
3.3 Concatenation Approaches

Figure 6: Concatenation of two RNAs using a phantom linker.

that solving RNA-RNAi while ignoring folding in one RNA can be done in polynomial time. To this end, Mneimneh presents a polynomial time dynamic programming algorithm that, given two RNAs $x = x_1x_2\ldots x_m$ and $y = y_1y_2\ldots y_n$, considers $x$ to be folding and $y$ to be non folding. Let $V(i, j, k, l)$ be the optimal solution achieved for $x[i\ldots j]$ and $y[k\ldots l]$. There are three cases for $x_j$: 1) $x_j$ does not bond, 2) $x_j$ bonds with some $x_p$ (edge $(x_p, x_j) \in S$) and $i \leq p \leq j$, or 3) $x_j$ bonds with some $y_q$ (edge $(x_j, y_q) \in S$) and $k \leq q \leq l$. Therefore

$$V(i, j, k, l) = \begin{cases} 
V(i, j - 1, k, l) \\
V(i, j - 1, k, q - 1) + w(x_j, y_q) \\
V(i, p - 1, k, q) + V(p + 1, j - 1, q + 1, l)
\end{cases} \quad (40)$$

where $i \leq p \leq j$ and $k \leq q \leq l$ and $w$ is the weight function. The complete solution is given by $V(1, m, 1, n)$.

Optimal solutions to RNA-RNAi problem sometimes contain a substructure known as the entangler or zigzag.

**Definition 3.2.** An entangler is a set of five nonintersecting edges that contain two folding edges $(x_i, x_j)$ and $(y_k, y_l)$, and three binding edges $e_1, e_2, e_3$ such that

- $e_1 = (x_p, y_q) \Rightarrow p \in (i, j), q \notin (k, l)$,
- $e_2 = (x_p, y_q) \Rightarrow p \in (i, j), q \in (k, l)$,
- $e_3 = (x_p, y_q) \Rightarrow p \notin (i, j), q \in (k, l)$,

where $(i, j)$ denotes $\{i + 1, \ldots, j - 1\}$.

It is not possible to capture entanglers by the given dynamic programming, since most dynamic programming algorithms for RNA-RNAi recursively divide the problem into two independent subproblems. It is not possible to break an entangler into smaller subproblems, as at least one folding edge must be dropped.

3.3 Concatenation Approaches

An approach based on RNA folding methods is to simply concatenate the two RNAs, using a short chain of nucleotides known as a phantom linker. This is consistent with the ordering required for RNA interaction - the 5' end of the linker is attached to the 3' end of RNA 1, and the 3' end of the linker is attached to the 5' end of RNA 2, and if the linker is now "bent", RNA 1 and 2 are in opposite directions. This chain of three RNAs can be considered a single, long RNA by itself, since the ends of each internal RNA align correctly. This single RNA is now folded by a modified version of Zuker's algorithm or its variants. The modification is necessary since the contributions of loops (or stacks) involving the linker need to be ignored. Specifically, there
are two cases that demand special attention: 1) some of the linker’s bases participate in bonding, or 2) the linker is part of some loop (of any type). In the first case, the entire structure should be ignored, since such a structure would be impossible to obtain with the original two RNAs. The second case can be solved by ignoring the contribution of the loops that enclose the linker.

The concatenation approach is used in tools such as PairFold [4], RNAcofold [7], and [17] using Zuker’s algorithm. A partition function based approach was employed by Cao and Chen in [10] to compute energy landscapes of joint structures of interacting RNAs instead of MFE structures. Adapting McCaskill’s partition function to concatenated RNAs follows along the same line as the approach described here.

While concatenation approaches incorporate accessibility of bonding sites, they fall short of accurate prediction due to their inability to predict kissing loops. A kissing loop between the two RNAs would appear as a pseudo knot on the long RNA with bases on both sides. However, it is possible to use pseudoknot prediction algorithms on the concatenated RNAs to get reasonable results. An $O(n^5)$ algorithm for prediction of kissing loops on the same RNA strand was developed in [11], and it could be extended to concatenated RNAs, but the running time would make such an algorithm undesirable.

3.4 Other MFE approaches with Intramolecular Bonds

In [2] Alkan et al introduce methods to predict the secondary structure. One is a base pair (both arcs and bonds) maximization algorithm. However, the other approach, the emphstacked pair energy model, is more realistic since it is based on free energy minimization using the Nearest Neighbor model. In this model, Alkan et al compute the structure with minimum free energy sum of all stacked pairs. Other types of loops are not taken into account. However, the recursive formulation is over the joint structure, so internal arcs and external bonds are both considered. The mathematical formulation is quite complex, so instead we describe the algorithm using recursion diagrams in Figure 7. For computing the overall minimum energy, we need to define four other energy functions - two which condition on internal arcs, and two which condition on bonds between the two RNAs:

1. $E_s(S[i,j],R[i',j'])$ denotes the free energy between $S$ and $R$ such that $S_i$ bonds with $S_j$.

2. $E_p(S[i,j],R[i',j'])$ denotes the free energy between $S$ and $R$ such that $R_{i'}$ bonds with $R_{j'}$.

3. $E_l(S[i,j],R[i',j'])$ denotes the free energy between $S$ and $R$ such that $S_i$ bonds with $R_{j'}$.

4. $E_r(S[i,j],R[i',j'])$ denotes the free energy between $S$ and $R$ such that $S_j$ bonds with $R_{i'}$. 

24
3.4 Other MFE approaches with Intramolecular Bonds

**Figure 7**: Definition of $E_s$ and $E_l$ using recursion diagrams.

$E_s(S[i,j], R[i',j'])$ is pictorially defined in Figure 7(a). Recall that in this model, a bond contributes energetically only if it is part of a stacked pair (and in general, never by itself). Therefore we can split this definition into two cases: either bases $i+1$ and $j-1$ form an arc or they do not. If we assume that $i+1$ and $j-1$ form a bond, then we sum the energy of the stacked pair $S[i, i+1, j, j-1]$ and the energy of the region $S[i+1, j-1]$, which is equivalent to $E_s$. Otherwise, we assume the general case and set the energy to $E$. The energy $E_s(S[i,j], R[i',j'])$ can be defined similarly, assuming the arc $i' \cdot j'$ instead of $i \cdot j$.

The second type of energy function assumes that there is a bond between $S_i$ and $R_{i'}$. Again, this bond is energetically meaningful only if the bond $i+1 \cdot j-1$ exists. Therefore, we can split this into cases: the stacked pair $[i, i+1, i', i'+1]$ exists, in which case the energy of this pair is $E_l(S[i+1,j], R[i'+1,j'])$ (because now $i+1 \cdot i'+1$ is a bond), or if it doesn’t exist, then the general case $E$. As before, instead of exclusively computing only case based on the existence of the bond, we compute the energies of both cases and choose the smaller one. This is because even if there is a bond, one or both of the bases may be participating in stronger intermolecular loops, a case which would be covered by $E$. The term $E_r(S[i,j], R[i',j'])$ can be defined similarly.

The actual energy term $E(S[i,j], R[i',j'])$ can now be defined in terms of nine sub-solutions. The first four are $E_s, E_l, E_r, E_{rr}$, as discussed above. However, we also have to consider other bonds or arcs that $i, j, i', j'$ may participate in, so we need to split this structure in other ways. We consider the five cases shown in Figure 8. In the first case, all possible values of $k$ and $k'$ are considered, whereas in cases 2, 3 and 4, 5 only all values of $k$ or $k'$ (respectively) are considered. Owing to case where all values of $k$ and $k'$ have to be considered, the running of this algorithm becomes $O(n^6)$, which is not feasible for longer RNAs.
In this section, we describe some methods to compute the joint partition function of interacting RNA sequences and how these values can be used to determine probabilistically likely structures. When computing likely interaction structures, a technique that has proved successful is to first determine the regions on each RNA that are likely to be unpaired during folding. Even though this is a computation over a single RNA, a different partition function than McCaskill’s has to be derived for this problem. However, it’s root is still McCaskill’s algorithm.

### 4.1 Computing Accessibility

The earliest solutions using this approach consider the viability of two sites interacting by considering the possibility, or rather probability, of both sites being available for interacting with each other. Unlike the previously discussed methods, neither do they completely ignore internal bonds nor do they consider only free regions. Instead, it is determined whether the energy contribution of interaction outweighs the energy required to free those regions, i.e., remove all intramolecular bonds and make the regions accessible for interaction.

Suppose such a region of interest is \([i, j]\) on an RNA \(R\). To determine its probability of being unpaired, we have to consider all possible structures of \(R\) in which \([i, j]\) remains unpaired. We can compute the restricted partition function of this ensemble; it is similar to McCaskill’s partition function algorithm. Let this partition function be \(Z_{R}^{U(i,j)}\). Using the idea of computing the probability of a certain substructure existing...
in the ensemble, as discussed in section 1.4, we get

\[ P_A^R(i, j) = \frac{Z_{U(i,j)}^R}{Z_R^R} \]  

(41)

where \( Z_R^R \) is the partition function of RNA \( R \).

There is another way to derive this quantity in terms of energies. Define the ensemble energy of structures with \([i, j]\) unpaired as \( E_{U(i,j)}^R = -RT \ln(Z_{U(i,j)}^R) \), and the ensemble energy of all structures as \( E_R = -RT \ln(Z_R^R) \). Then the energy required to make the region \([i, j]\) accessible is

\[ E_A^R(i, j) = E_{U(i,j)}^R - E_R \]  

(42)

It is easy to verify that \( E_A^R(i, j) = -RT \ln P_A^R(i, j) \), or equivalently, \( P_A^R(i, j) = \exp^{-E_A^R(i, j)/RT} \). These quantities are computed for all possible regions \([i, j] \in R \) and \([k, l] \in S \).

However, to determine whether it is feasible to pair a region on \( R \) with a region on \( S \), the energy of interaction of those regions needs to be computed. Indeed, the energetics can be viewed as a two step process: opening up of a region on either RNA and then hybridization. This can represented with the following equation:

\[ \Delta E = E_A^R(i, j) + E_A^S(i, j) + E_{RS}^I \]  

(43)

where \( E_{RS}^I \) is the energy of interaction between two regions on \( R \) and \( S \), which can also be thought of as the gain in energy due to hybridization). If \( \Delta E < 0 \) (recall that energies are negative) then interaction is feasible.

Efficiently computing \( P_U^{U(i,j)} \) is not as straightforward as implied by eq (41). In [36] a restricted partition function is used with two cases - either \([i, j]\) is not covered by an arc, or it is. In the former case, the partition function is easy to compute, since it is just the product of partition functions on either side of \([i, j]\) (the partition function of the substring \([i, j]\) is 1). Computing the latter case requires considering all types of loops that may be involved: \([i, j]\) could be on a hairpin loop’s unpaired region, it could be on an interior loop, or it may a part of a multiloop. Assuming that \([i, j]\) is covered by \( p \cdot q \) in the second case, we have:

\[ P_U^{U(i,j)} = \frac{Z_{[i, i-1]} \times Z_{[j+1, N]} \times Z^R}{Z} + \sum_{p < i \leq q} P_{pq} \times \frac{Z_{pq}^U[i, j]}{Z^R_{[P, q]}} \]  

(44)

In the summation, we compute the conditional probability that \([i, j]\) is unpaired given that \( p \) and \( q \) form an arc, hence the division by \( Z^R_{[P, q]} \), which is the partition function of all structures in which \( p \) and \( q \) form an arc. For a breakdown of \( Z_{pq}^U[i, j] \) into cases involving loops and implementation details, see [36]. This has a running time of \( O(n^4) \), however, in reality long unpaired regions are rare, and if their length is bounded by \( w \), the running time scales down to \( O(n^3 w) \).
The above idea is employed in RNAup by Muckstein et al [36] and in IntaRNA [9], although they differ in how the final quantities are computed. Muckstein et al consider an ensemble of simple structures for the interaction partition function. They consider only interior loops formed between the two RNA strands, and thus have:

\[ Z^I[i,j,i^*,j^*] = \sum_{i<k<j \atop i^*<k^*<j^*} Z^I[i,k,i^*,k^*] \exp^{-I(k,k^*;j,j^*)} \]  

Thus the partition function of all structures where \([i,j] \in R\) binds to \([k,l] \in S\) is given by

\[ Z^*[i,j,k,l] = P^U_{R(i,j)} \times P^U_{S(i,j)} \times Z^I[i,j,k,l] \]  

Taking the log on both sides is equivalent to eq (43). The quantity \(-RT \ln Z^*[i,j,k,l]\) is ensemble energy of interaction in the window \([i,j],[k,l]\). After computing these for all \(1 \leq i < j \leq n\) and \(1 \leq k < l \leq n\), RNAup outputs the window with least energy. It’s assumption is that there is only one binding site, and multiple binding sites are not independent. However, Ahmed et al have extended this model and used these windows, assuming independence and additivity as a heuristic, to compute the optimal [1] and suboptimal [35] structures of multiple interacting RNAs, with satisfactory results.

### 4.2 Joint Partition Function

The accessibility methods described in the previous section offer good heuristics, but they are not ideal, since the joint structure of two RNAs is not considered. Although Alkan et al did consider the joint consider to compute MFE (section 3.4), the MFE structure is not always correct - for reasons described earlier, a distribution of such structures is desirable. Solving this problem requires a method to compute the partition function of the joint structure. Such a method was derived independently by Chitsaz et al (piRNA) [14] and Huang et al (RIP) [25]. Having a joint partition function allows one to compute melting point temperatures and base pairing probability in the interaction structure. Let \(Z^J(T)\) be the joint partition function of two RNAs:

\[ Z^J(T) = \sum_{S \in \mathcal{S}} \exp^{E(S)/RT} \]  

where \(\mathcal{S}\) is the set of all possible since or duplex secondary structures that do not contain pseudoknots, crossing bonds and zig-zags.

The joint partition function has to be defined using sophisticated decomposition into substructures taking into account both folding arcs and interaction bonds. Recall that a partition function has to be defined as a sum over disjoint substructures. The recursions work by splitting a structure into two (bifurcation) but only if there are no arcs that are being split. If there is an arc that is spanning the split, the structure is first decomposed into the arc and it's inner region. Arcs on either RNA have to be
4.3 From Joint Partition Function to Structure

Once the partition function of all duplex structures is available, one can derive from it the partition function of all structures in which there is some intermolecular interaction. On its own this quantity would require decomposition into many cases and may be computationally intensive; however, the equation

\[
Z_{R_x,S_y}^I = Z_{R_x,S_y}^J - Z_{R_x} Z_{S_y} \tag{48}
\]

computes the partition function of all joint structures excluding (minus) the partition functions of structures where only intramolecular arcs exist. From this, we can derive the probability of interaction:

\[
P(R_x \circ S_y) = \frac{Z_{R_x,S_y}^I}{Z_{R_x,S_y}^J} \tag{49}
\]

and thus the ensemble energy of interaction is \(E(R_x,S_y) = -RT \ln P(R_x \circ S_y)\). Note that \(P(R_x \circ S_y)\) (and by extension \(E(R_x,S_y)\)) is only a measure of how likely two subsequences will interact given the fact that there is no intramolecular folding, so to determine the actual likelihood of interaction, we have to use the probability that both regions \(R_x\) and \(S_y\) are unpaired, as computed in section 4.1. The total cost of interaction is

\[
C(R_x,S_y) = E^U(R_x) + E^U(S_y) + E^I(R_x,S_y) \tag{50}
\]

4.4 Joint Probability of Unpaired Regions

The dynamic programming algorithms used to compute a set of nonconflicting interaction windows consider windows independent of each other. In reality, if there are two unpaired regions \(X, Y\) on an RNA \(R\) then \(P(u[X], u[Y]) \neq P(u[X]) \times P(u[Y])\), since the opening up of one region may positively or negatively influence the opening of another region. Thus one needs another method to compute joint probabilities of two regions \(X\) and \(Y\) of being unpaired. This is computed using Baye’s rule

\[
P(u[X], u[Y]) = P(u[X] | u[Y]) \times P(u[Y]) \tag{51}
\]

The first term in the above product can be computed using special partition functions that keep \([k, l]\) unpaired during recursions. However, the cost of computing this for all pairs of intervals on a single RNA is \(O(n^4w)\), and extending the algorithm to com-
Table 2: This Table shows the sensitivity, PPV and F-measure for RNA-RNA binding sites prediction by (1) inRNAs, (2) IntaRNA [16], and (3) RNAup [15]. The dataset is compiled by Kato et al. [13] and Busch et al. [16]. We have obtained this table from Salari et al [42], where they show the performance of their algorithm inRNAs against other prediction software.

Salari et al use the above probability to provide an improved dynamic programming algorithm. In their algorithm, a region on one RNA can interact with two disjoint regions on the second RNA. Other interaction windows can be added based on the independence assumption. However, the algorithm does not allow crossing bonds nor dependent windows which do not share regions (see fig).

Chitsaz et al incorporate this model in their program biRNA [13] to compute sets of non intersecting windows while not ignoring their dependence on each other. The complexity of the algorithm grows exponentially in the number of windows that are being considered per structure.
4.5 Performance

In Tables 2 and 3 we show the performance of several prediction softwares relative to each other. In Table 2, we list the following measures:

\[ \text{sensitivity} = \frac{\text{number of correctly predicted base pairs}}{\text{number of true base pairs}} \]  \hspace{1cm} (52)

\[ \text{PPV} = \frac{\text{number of correctly predicted base pairs}}{\text{number of predicted base pairs}} \]  \hspace{1cm} (53)

\[ F = \frac{2 \times \text{sensitivity} \times \text{PPV}}{\text{sensitivity} + \text{PPV}} \]  \hspace{1cm} (54)

Sensitivity is also known as precision, and PPV, which stands for positive predictive value, is also known as recall.

What is of interesting to note here is that the problem of RNA interaction prediction is still not completely solved - the average F-measure of the RNA pairs in the dataset using inRNAs is 0.845, and is as low as 0.555 for the GcvB-lvK pair. However, for many other pairs it is close to 1. One of the major factors for low measures is the fact that energy model itself is weak, whereas the algorithm may be reasonable enough. None of the prediction algorithms account for all possible factors that may influence structure formation, such as coaxial stacking and the presence of proteins.

<table>
<thead>
<tr>
<th>Pair</th>
<th>Binding Site(s)</th>
<th>biRNA Site(s)</th>
<th>-G</th>
<th>RNAup Site(s)</th>
<th>G</th>
</tr>
</thead>
<tbody>
<tr>
<td>GcvB gltI</td>
<td>[66,77]</td>
<td>[64,81]</td>
<td>11.3</td>
<td>[75,93]</td>
<td>18.7</td>
</tr>
<tr>
<td>GcvB argI</td>
<td>[75,91]</td>
<td>[71,90]</td>
<td>13.1</td>
<td>[72,91]</td>
<td>20.2</td>
</tr>
<tr>
<td>GcvB livI</td>
<td>[63,87]</td>
<td>[66,84]</td>
<td>13.1</td>
<td>[71,90]</td>
<td>14.9</td>
</tr>
<tr>
<td>GcvB livK</td>
<td>[68,77]</td>
<td>[67,86]</td>
<td>12.2</td>
<td>[67,86]</td>
<td>19.0</td>
</tr>
<tr>
<td>GcvB oppA</td>
<td>[65,90]</td>
<td>[67,86]</td>
<td>9.3</td>
<td>[67,86]</td>
<td>15.3</td>
</tr>
<tr>
<td>GcvB STM4351</td>
<td>[70,79]</td>
<td>[69,77]</td>
<td>9.6</td>
<td>[69,87]</td>
<td>17.7</td>
</tr>
<tr>
<td>DsrA rpoS</td>
<td>[8,36]</td>
<td>[21,40]</td>
<td>9.4</td>
<td>[33,14]</td>
<td>16.3</td>
</tr>
<tr>
<td>RpoA rpoS</td>
<td>[33,62]</td>
<td>[25,7]</td>
<td>9.4</td>
<td>[33,14]</td>
<td>16.3</td>
</tr>
<tr>
<td>IstR tisA</td>
<td>[65,87]</td>
<td>[66,85]</td>
<td>18.1</td>
<td>[66,85]</td>
<td>29.0</td>
</tr>
<tr>
<td>MicC ompC</td>
<td>[1,30]</td>
<td>[1,16]</td>
<td>18.5</td>
<td>[119,104]</td>
<td>18.7</td>
</tr>
<tr>
<td>MicF ompF</td>
<td>[1,33]</td>
<td>[14,30]</td>
<td>8.0</td>
<td>[116,100]</td>
<td>14.7</td>
</tr>
<tr>
<td>RhbB sdhD</td>
<td>[9,50]</td>
<td>[22,41]</td>
<td>15.8</td>
<td>[116,98]</td>
<td>21.5</td>
</tr>
<tr>
<td>RhbB sodB</td>
<td>[36,46]</td>
<td>[38,46]</td>
<td>9.7</td>
<td>[60,45]</td>
<td>10.3</td>
</tr>
<tr>
<td>Spot42 galK</td>
<td>[1,61]</td>
<td>[126,52]</td>
<td>20.5</td>
<td>[84,68]</td>
<td>14.6</td>
</tr>
<tr>
<td>IncRNA54 repZ</td>
<td>[16,42]</td>
<td>[19,38]</td>
<td>35.3</td>
<td>[19,38]</td>
<td>37.5</td>
</tr>
<tr>
<td>OxyS fhaA</td>
<td>[22,30]</td>
<td>[95,87]</td>
<td>7.9</td>
<td>[-96,104]</td>
<td>10.3</td>
</tr>
<tr>
<td>CopA CopT</td>
<td>[22,33]</td>
<td>[70,59]</td>
<td>25.9</td>
<td>[43,24]</td>
<td>23.9</td>
</tr>
</tbody>
</table>

Table 3: Binding sites reported in the literature and predicted by biRNA and RNAup. \( \Delta G \) is in kcal/m. Two RNAs interact in opposite direction, hence, sites in the second RNA are presented in reverse order.
5 Conclusion and Future Work

In this survey, we have described some widely used methods to predict RNA secondary structures that arise as a result of folding or interaction. While much research has been put into the problems of RNA folding and interaction, the results can be improved. Recently Chitsaz et al. have been looking into the different conditions that can make a prediction model achieve 100% accuracy when the training and testing datasets are the same [12].

The problem of interaction prediction offers many more directions. Recently, we have explored the interaction of multiple RNA sequences in generally, and specifically in the spliceosome. While some work has been done in this area previously, e.g. [5, 17], the models are not robust enough to handle all possible structures. Another challenge faced by multiple RNA interaction prediction is that only a few examples are known in nature, which means that many algorithms have to rely on heuristics. Our focus has been solving this problem assuming the two RNA interaction prediction gives reasonable results. To this end we have developed a model called the Pegs and Rubberbands model that first computes energy values for two RNAs using RNAup and then computes the best structure for $n$ RNAs. Our algorithm has been extended to return suboptimal structures by considering all structures with energy $E$ such that $E_{opt} \leq E \leq (1-\epsilon E_{opt})$, and then clustering structures to return only “classes” of structures that are not too similar.

References


REFERENCES


REFERENCES


