RNA Secondary Structure Prediction using Probabilistic Context-Free Grammar

Aly Valliani AVALLIA1@SWARTHMORE.EDU
Swarthmore College, 500 College Ave., Swarthmore, PA 19081 USA

Michael Superdock MSUPERD1@SWARTHMORE.EDU
Swarthmore College, 500 College Ave., Swarthmore, PA 19081 USA

Abstract

The functional role of RNA in cellular processes has been receiving increasing interest over the years. Knowledge of tertiary structure is to-date the most accurate predictor of biological function. However, experimental means for determining RNA tertiary structure are both expensive and time-consuming. As such, methods for predicting RNA secondary structure as a proxy for tertiary structure have been garnering greater interest in recent years.

In this paper we present a probabilistic context-free grammar (PCFG) based approach for RNA secondary structure prediction. Beginning with a grammar classically used for RNA structure prediction, we provide a detailed description of our processes for translating it into Chomsky Normal Form and for adding relevant production rules. After defining our grammar and re-estimating its parameters, we analyze the accuracy of our system by comparing it to similar secondary structure predictors, testing on transfer RNAs (tRNAs) specifically. Experimental results show that we can identify stems with 89.78% sensitivity and 88.44% positive predictive values (PPV), outperforming the most widely used minimum free energy (MFE) models today.

1. Introduction

The prediction of RNA secondary structure has been a long standing challenge facing Bioinformatics research. The task entails predicting the two-dimensional RNA structure from one-dimensional RNA sequence data. Although incomplete, knowledge of RNA secondary structure is a good proxy for approximating tertiary structure and therefore a good model for predicting the functional role of RNA in cellular processes. Current experimental techniques for determining RNA structure, such as X-ray crystallography and Nuclear Magnetic Resonance (NMR), are time-consuming and expensive. As such, computational methods for predicting secondary structure are of particular importance in the field.

RNA differentiates itself from its more well-known macromolecular counterpart, DNA, in that it is a single-stranded molecule, which allows it the flexibility to fold onto itself via pairings between complementary bases. This results in a diverse set of secondary structures that include but are not limited to loops, stems and pseudoknots (Figure 1).

Although an understanding of these structures does not elucidate their relative positions in three-dimensional space, it allows us to identify the locations of important sites within the RNA structure and generalize about the thermodynamically stable conformation of the macromolecule, thereby indicating its importance in biological processes (Higgs, 2000).

Currently, there exist diverse computational approaches aimed at resolving the task of RNA structure prediction. Energy-based methods utilizing dynamic programming are some of the earliest approaches to tackle this problem. These methods aim to predict secondary structure by minimizing thermodynamic energies associated with base-stacking to imitate RNA folding to its most energetically favorable state (Zuker & Stiegler, 1981). Subsequent refining of energy calculations via the incorporation of melting point, temperature, and other thermodynamic features led to the prediction of pseudoknot secondary structures and made these algorithms some of the most widely used today (Zuker, 2003).

Maximum weighted matching (MWM) methods are a more recent approach to RNA secondary structure prediction. Such algorithms utilize various kinds of evidence, such as

CPSC 68 Bioinformatics Final Report Proceedings, Swarthmore College, Fall 2014.

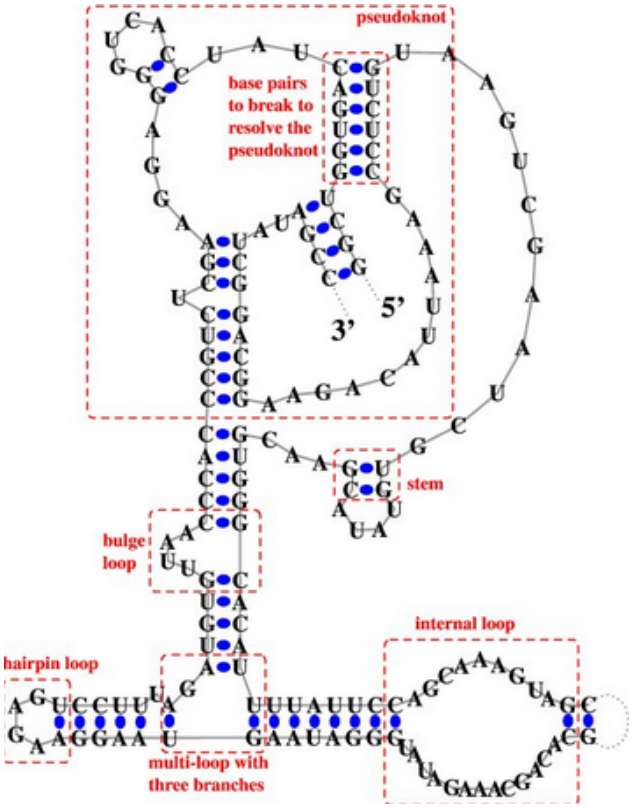

Figure 1: RNA with labeled secondary structures, adapted from (Andronescu et al.).

phylogenetics, thermodynamics and NMR, among others, to determine the prevalence of single bases and base pairs in various types of RNA structure. Using this information, these algorithms determine the best structure to be the one for which the most corroborating and least conflicting evidence exists. As such, these algorithms are able to predict pseudoknots and other secondary interactions (Tabaska et al., 1996). Iterated loop matching is a more recent extension on MWM methods that is more sensitive and specific than its predecessors (Ruan et al., 2004a;b).

Grammar-based approaches are another set of methods aimed at tackling the problem of RNA structure prediction through the use of probabilistic context-free grammars (PCFGs) (Sakakibara et al., 1994). Such algorithms apply formal language theory to extend Hidden Markov Models (HMMs) to the problem of RNA structure prediction. They utilize a standardized grammar assigned with probabilities to represent the frequency with which a grammarized production represents a sequence. These algorithms, however, cannot detect pseudoknot secondary structures. Algorithmic approaches utilizing context-interaction grammars were subsequently introduced to extend the grammar utilized within PCFG algorithms to enable the detection of pseudoknots (Rivas & Eddy, 1999). Such approaches utilize a more context-sensitive grammar to aid in the detection of such secondary structures.

A variety of ensemble methods utilize combinations of the methods discussed above to create a model that better represents the true nature of RNA. For example, the KH-99 algorithm utilizes PCFGs in addition to maximum likelihood trees to account for evolutionary history when predicting secondary structure (Knudsen & Hein, 1999). Methods have since extended upon such algorithms (Knudsen & Hein, 2003).

In this paper we describe our approach for resolving RNA secondary structure, in particular loops and stems, using PCFGs. This paper is organized as follows: we provide theoretical details on PCFGs and the algorithms required for their implementation in the following section. Next, we provide implementation details regarding the determination of priors, the formalization of our grammar and the specific applications of our PCFG algorithms. Subsequently, we evaluate our resulting algorithm against those in existence. We conclude with future directions and a brief meta-discussion of our project.

2. Probabilistic Context-Free Grammar

We provide here a definition of context free grammars, in mathematic notation[1][2]:

A context-free grammar (CFG) G = (Σ, V, R), where Σ is a set of terminals (alphabet) and V is a set of non-terminal symbols. Σ and V are both finite disjoint sets. R is a finite set of rule productions of the form $A \rightarrow \alpha$, $A \in V$, $\alpha \in (V \cup \Sigma)$. A special symbol $S \in V$ that denotes the final parse. A probabilistic context-free (PCFG) is a four-tuple G = (Σ, V, R,P) similar in definition to CFGs with the inclusion of a probability P, which is a probability function assigning a probability in the range (0, 1] to all rules in R. P should satisfy for all non-terminals A in V such that: $\sum_{\alpha} P(A \rightarrow \alpha) = 1$, which indicates that the probabilities of the productions from the same non-terminals should add up to 1. As with CFGs, there is a special symbol $S \in V$ denoting the final parse.

2.1. Chomsky Normalization

In order to make the use of PCFG easier, the grammar must be normalized using Chomsky Normal Form (CNF) (Chomsky, 1959). This normalization limits non-terminal productions to only yield two non-terminals, which can be applied to simplify algorithmic development. A CFG (or PCFG) is in CNF if all of the production rules are one of

[1] http://cs.au.dk/ cstorm/students/Knudsen_Jun2005.pdf
[2] http://www.cs.jhu.edu/ jason/465/iobasics.pdf

two types:

$$A \rightarrow BC$$

$$A \rightarrow t$$

where A, B and C are non-terminals and t is a single terminal symbol[1].

2.2. Algorithms

There are three key algorithmic questions motivated by PCFGs once a sequence and a grammar with prior probabilities are known:

1. What is the probability that the sequence can be generated by the grammar?
2. What is the most probable parse that corresponds to the sequence?
3. How can production probabilities be learned from sequence data?

The algorithmic solutions to these questions are the Inside or Outside Algorithm, the CYK Algorithm, and the Inside-Outside Algorithm, respectively. All algorithms assume CNF and employ dynamic programming methods to avoid the re-computation of overlapping sub-problems. Such algorithms work by calculating the probabilities of all subsequences $x_i...x_j$ beginning with a sub-sequence of length zero and iterating recursively to longer and longer subsequences.

2.2.1. Inside Algorithm

The Inside Algorithm is a dynamic programming algorithm used to determine the likelihood of a particular sequence. In addition, it is used to estimate fractional counts for production rules in order to learn their probabilities as will be described in section 2.2.4.

The algorithm aggregates probabilities (α) of subsequences of increasing length in a three-dimensional matrix, aTable[j,i,V], where j represents the starting index of the sub-sequence, i is the length of the current subsequence and V is the non-terminal production rule. We use the following recurrence[1,2] :

Initialization:

$$\alpha_{ii}(A) = \phi(A \rightarrow t)$$

Recurrence:

$$\alpha_{ij} = \sum_{B,C} \sum_{i \leq k \leq j} \phi(A \rightarrow BC)\ \alpha_{ik}(B)\ \alpha_{k+1,j}(C)$$

for i < j where k represents two sub-spans within the subsequence and ϕ represents the probability of a production.

2.2.2. Outside Algorithm

The Outside Algorithm, like the algorithm discussed in the previous section, is used to determine the likelihood of a particular sequence. However, it is most often incorporated in conjunction with the Inside Algorithm to learn production probabilities.

This algorithm aggregates probabilities (β) of subsequences of decreasing length in a three-dimensional matrix, bTable[j,i,V], similar to the one utilized by the Inside Algorithm. We use the following recurrence[1,2] :

Initialization:

$$\beta_{in}(S) = 1,\ \beta_{1n}(A) = 0$$

where n is the length of the sequence.

Recurrence:

$$\beta_{ij}(A) = \sum_{B,C} \sum_{n \geq k > j} \phi(B \rightarrow CA)\ \alpha_{k,i-1}(C)\ \beta_{kj}(B) + \sum_{B,C} \sum_{n \geq k > j} \phi(B \rightarrow AC)\ \alpha_{j+1,k}(C)\ \beta_{ik}(B)$$

2.2.3. CYK Algorithm

The CYK (Cocke-Younger-Kasami) Algorithm is a dynamic programming algorithm used to determine the most probable parse for a given sequence by taking the maximum probability of a particular production rule at every iteration of a sub-sequence. In addition, it stores backpointers to produce the most probable parse following its probabilistic determination. Otherwise, it is identical to the Forward Algorithm in its implementation.

2.2.4. Inside-Outside Algorithm

The Inside-Outside Algorithm is a type of estimation-maximization (EM) algorithm that incorporates the Inside and Outside algorithms to estimate fractional counts. The counts are then used to update production probabilities within a grammar. We estimate fractional counts (c_ϕ) for each sequence (W) using the following formulation[1,2] :

$$c_\phi(A \rightarrow BC, W_n) = \frac{\phi(A \rightarrow BC)}{\alpha_{1n}(S)}(\alpha_{ij}(B)\alpha_{ij}(C)\beta ij(A))$$

We then calculate updated production probabilities ($\phi^{'}$) using the previously estimated fractional counts in the following manner:

$$\phi^{'}(A \rightarrow BC) = \frac{\sum_{i=1}^{N} c_\phi(A \rightarrow BC, W_i)}{\sum_{j=1}^{N} c_\phi(A \rightarrow XY, W_j)}$$

where N is the number of sequences used for training and X and Y are all non-terminals produced by rule A.

3. Implementation

The model was estimated using the following steps:

1. Training data was obtained from a suitable RNA database.
2. Single base and base pair frequencies were estimated.
3. The grammar was constructed.
4. The grammar parameters were learned.

3.1. Training Data

We obtained tRNA sequences from a broad spectrum of organisms using the transfer RNA database (tRNAdb) (Sprinzl et al., 1998). Structure data, which was also provided by the tRNAdb, was obtained in dot-parentheses notation, in which dots represent loops and matching parentheses denote stem pairs. We processed all sequences in our data set by removing sequences with unknown nucleotides. For each species, we only saved one training example for each of the amino acids to which the tRNAs bind. After processing the data we were left with 4058 tRNA sequences for training.

Of the 4058, we trained using only the first 2000 sequences organized in alphabetical order as provided by the tRNAdb. We limited our training to 2000 sequences for efficiency purposes. We found that larger training sets produce negligible gains in prediction accuracy. The utilized sequences range from 64-95 bases in length, with an average length of 76 bases.

3.2. Frequencies

Single base frequencies were estimated from the counts of bases at each position in the training sequences. We counted bases found in the stem and loop secondary structures separately. In order to generalize our model to unseen data, we treated stems with bases XY as synonymous to stems with bases YX. These percentages are shown in Table 1, which shows a high frequency of GC/CG base pairs likely due to their strong base stacking interactions. In addition, the infrequently occurring "wobble pair" of UG/GU base pairs is also present.

Table 1: Stem and loop counts for all possible base pairs and bases respectively, generated using all sequences in our data set

	Stem		Loop		Overall
AU/UA	23.99%	A	28.52%	A	19.49%
GC/CG	69.56%	C	18.75%	C	27.51%
UG/GU	6.46%	G	23.30%	G	31.33%
Other	0.00%	U	29.43%	U	21.67%
Total:	54.61%	Total:	45.39%		

3.3. The Grammar

3.3.1. Foundations

We model our grammar off of that proposed by B. Knudson and J. Hein, who use both a stochastic context-free grammar and evolutionary history to predict RNA secondary structure (Knudsen & Hein, 1999). Their grammar consists of a set of three non-terminals, each with two production rules. We introduce them here and explain their function in the prediction of RNA secondary structure. In parentheses are the probabilities assigned to each of these production rule as indicated by Knudson and Hein (Knudsen & Hein, 1999).

The production rules:

$$S \rightarrow LS \text{ (0.869)} \mid L \text{ (0.131)}$$

$$F \rightarrow dFd \text{ (0.788)} \mid LS \text{ (0.212)}$$

$$L \rightarrow s \text{ (0.895)} \mid dFd \text{ (0.105)}$$

Here, d and s represent the terminal symbols of the grammar. An s corresponds to a base that is located in a loop. A d corresponds to a base that is found in a stem. As may be inferred from our grammar, the production of dFd contains two ds linked to one another in a stem.

At a high level, each of the three non-terminals holds a specific purpose in the prediction of RNA secondary structure. S is the start symbol of our grammar, from which we derive our string of terminals that are indicative of RNA structure. In this grammar we see that the S non-terminal allows primarily for the extension of loops, the F non-terminal allows primarily for the extension of stems, and the L non-terminal generally produces a base in a loop or allows for a transition into a stem.

3.3.2. Our Formulation

With this grammar as a foundation for our model, we convert it to CNF for algorithmic convenience. We also introduce new production rules to fully model the range of possible secondary structures. Our final grammar is of the form:

$$S \rightarrow LS \mid O_x E_y \mid s_x$$

$$F \rightarrow O_x E_y \mid LS$$

$$L \rightarrow s \mid O_x E_y$$

$$E_x \rightarrow FC_x$$

$$O_x \rightarrow o_x$$

$$C_x \rightarrow c_x$$

We allow the subscripts x and y to indicate unique rules for all values of x and y, where x and y are the set of bases $\{a,c,g,u\}$. Therefore, the notation for rule E_x, where $E_x \rightarrow FC_x$, is representative of four separate production rules:

$$E_a \rightarrow FC_a$$

$$E_c \rightarrow FC_c$$

$$E_g \rightarrow FC_g$$

$$E_u \rightarrow FC_u$$

To further clarify this notation, we indicate that the notation $F \rightarrow O_x E_y$ is indicative of 16 different production rules, one for each possible combination of two bases in $\{a,c,g,u\}$.

Here we explain the production rules of our grammar formulation. In our grammar, s_x, o_x, and c_x are terminal symbols. As before, s_x represents a base that is located in a loop. But for bases in stems we now use two separate production rules, o_x and c_x. The terminal o_x represents the first base of a stem, and the terminal c_x represents the second base of a stem.

We also introduce the non-terminals E_x, O_x, and C_x. These allow us to capture stem productions, which were represented before as dFd. These alternate production rules allow us to capture the same detail in CNF, while also being specific to the bases in a given stem pairing.

3.4. Grammar Parameter Estimation

In order to determine the prior probabilities of our grammar, we modified the probabilities proposed by Knudsen and Hein to fit our set of production rules. For rules that were conserved in our formulation, we used the exact same probabilities as those proposed by Knudsen and Hein. For rules that we added, we calculated priors using their proposed probabilities along with the loop and stem base probabilities that we calculated from our data set shown in Table 1.

We used five-fold cross validation over our set of 2000 training instances to re-estimate the grammar probabilities across each fold. These probabilities were re-estimated using the previously discussed Inside-Outside Algorithm. An average of the updated grammar probabilities across all five folds were then generated, and the results were used as the final probabilities for our grammar. We report these probabilities in the condensed form of our grammar below. A complete listing of each production rule and its corresponding probability can be found in Appendix A.

$$S \rightarrow LS \text{ (0.9056)} \mid O_x E_y \text{ (0.0315)} \mid s_x \text{ (0.0629)}$$

$$F \rightarrow O_x E_y \text{ (0.8038)} \mid LS \text{ (0.1962)}$$

$$L \rightarrow s \text{ (0.8313)} \mid O_x E_y \text{ (0.1682)}$$

$$E_x \rightarrow FC_x \text{ (1.0000)}$$

$$O_x \rightarrow o_x \text{ (1.0000)}$$

$$C_x \rightarrow c_x \text{ (1.0000)}$$

4. Evaluation

We evaluate our system using one of the common paradigms for evaluating RNA secondary structure prediction, which explicitly utilizes stems to determine prediction accuracy. A correctly identified stem match in our system is one in which the index of both the opening and closing stem positions in our prediction are identical to those in the accepted secondary structure. Through this method, loop matches are indirectly taken into account.

Within this evaluation paradigm, we calculate both the sensitivity and PPV (Positive Predictive Value) of our predictions. Sensitivity is the percent of correct stem predictions out of the total number of stems in our accepted tRNA secondary structure. PPV is the percentage of correct stem predictions out of the total number of predictions made. We provide the formulations for both of these metrics below:

$$Sensitivity = \frac{TP}{TP+FN}$$

$$PPV = \frac{TP}{TP+FP}$$

We use the remaining 2058 tRNA examples in our data set to measure the sensitivity and PPV for our prediction algorithm. We compare it against predictions made on the same data using the *pKISS* and *RNA-vienna* packages (Janssen & Robert Giegerich"; Zuker, 2003). Specifically, we compare against the RNA-vienna package's implementation of the classic minimum free energy algorithm of Zuker and Stiegler. We also compare against pKISS's minimum free energy implementation, utilizing their enforce parameter to suppress the prediction of pseudoknots.

5. Results and Discussion

We list the sensitivity and PPV measures for all three structure prediction methodologies in Table 2.

Table 2: Sensitivity and PPV measures for stem prediction using our probabilistic context-free grammar system and two minimum free energy systems (RNA-vienna and PKISS) on a set of 2058 tRNA sequences

	Sensitivity	PPV
PCFG (Ours)	**89.78%**	**88.44%**
MFE	73.09%	63.92%
PKISS	70.67%	63.54%

We find that the sensitivity of our system exceeds that of the other systems by over 16% and the positive predictive value of our system exceeds that of the others by over 24%. Both are substantial improvements over these commonly used structure prediction methods. These results indicate that our system is the preferred approach for the estimation of tRNA secondary structure.

In Figure 2 we display a set of structure predictions for a single tRNA sequence. We also provide the associated 2-dimensional secondary structures generated using VARNA in Figure 3. Using these results, we discuss the advantages and disadvantages of our system in finer detail.

Before getting into the details of our system, we highlight the prediction examples shown in Figure 3. We stress that these examples indicate the importance of having a system that predicts RNA structure with high accuracy. Although the minimum free energy systems generate structures with four major stems — demonstrating some consistency with the true tRNA structure — the locations of these stems are far from correct. Our model, on the other hand, seems to mirror the true structure with greater accuracy. The difference between our prediction and those of the the minimum free energy approaches are significant, especially when using these estimations to make useful inferences.

We attribute the overall success of our model to the narrow range of variation across samples in our data set. In using strictly tRNA data we find that most training instances, when specific to the same amino acid, are nearly identical in structure. Our data set, therefore, can be roughly categorized into a set of 20 tRNA structure types, each corresponding to a specific amino acid. We maintain that this lack of variation across our data is reasonable for the purposes of this study. Our emphasis is accuracy over generalizability, noting that while the minimum free energy methods may be more generalizable than our own, they occasionally result in misleading predictions. We provide an alternative, through the use of a probabilistic context-free grammar, demonstrating that our system is consistently accurate for a specific RNA type. We believe that these results would be consistent upon applying our system to other RNA types as well, but leave confirmation of this hypothesis as part of our future work, which we expand upon in Section 6.

Having now emphasized some of the strengths of our system, we briefly consider some of its limitations. There are two in particular. First, we note that our system is only able to predict stems and loops in RNA secondary structures. Pseudoknots, as well as other secondary structures common to RNA, cannot be modeled by our system. Although many context-free grammars have previously been proposed for this task, we have not incorporated them into our system (Rivas & Eddy, 1999). Secondly, we note that our system currently predicts loops that are too small to be consistent with true RNA structure. This is a flaw shared by the minimum free energy methods as well, as made evident in Figure 3. Adjustments that might prevent our system from predicting loops of less than 7 bases might allow us to make predictions more accurately. These two issues are not without solutions, and could be reasonably extended given our current work.

6. Conclusions

In this paper we have discussed our implementation of a probabilistic context-free grammar for the prediction of RNA secondary structure, with a specific focus on the prediction of secondary structure in tRNA. Using the work of Knudsen and Hein as a foundation for our own, we developed a new grammar that outperforms classic minimum free energy approaches in the task of tRNA secondary structure prediction. With 89.78% sensitivity and 88.44% PPV, our algorithm accurately estimates stem location. These results lead us to conclude that, for secondary structure prediction, using an RNA type-specific context-free grammar can be more useful than using a generalizable minimum free energy approach. Nevertheless, if inferences are to be made based on secondary structure predictors, it would likely require more accurate systems than the models highlighted here. Our work suggests that the use of probabilistic context-free grammars is a viable algorithmic approach for the further advancement of RNA structure prediction methodologies. Below, we outline potential extensions to further our work for this purpose.

The next steps we hope to take for the improvement of our system focus first on improving accuracy and second on making our system more generalizable, which is a difficult task considering the trade-off that exists between the two. We would like to improve upon our accuracy by handling the unnatural loop sizes (those less than 7 bases) which are often generated by our current model. A potential solution for this problem would likely involve a simple post-processing step that would identify and remove these unnatural stem pairings automatically. We would also like to consider variations of our grammar, focusing specifically on which grammars are best suited for different RNA types,

```
sequence:          gguccuguagcucaguugggagaguaucaccuugacaugguggggggucgcugguucgagaccagucgggacca
structure:         (((((((..((((........))))..(((((.......)))))).....(((((.......)))))))))))).
our estimation:    (((((((..((((........))))..(((((.......))))).....(((((((...)))))))))))))).
MFE estimation:    (((((((...(((....))))).((.(.(((((.......))))).).))(((((((...)))))))))))).
PKISS estimation:  (((((((..((((........))))...(((((.((...)).)))))).(((((((...)))))))))))).
```

Figure 2: The structure predictions for tRNA corresponding to valine (codon: GAC) of species Listeria_monocytogenes_str._4b_F2365 with ID 'tdbD00011215'. The first two lines list the accepted sequence and structure for this tRNA molecule. The following lines represent our system's prediction, the minimum free energy predictions, and the PKISS predictions respectively.

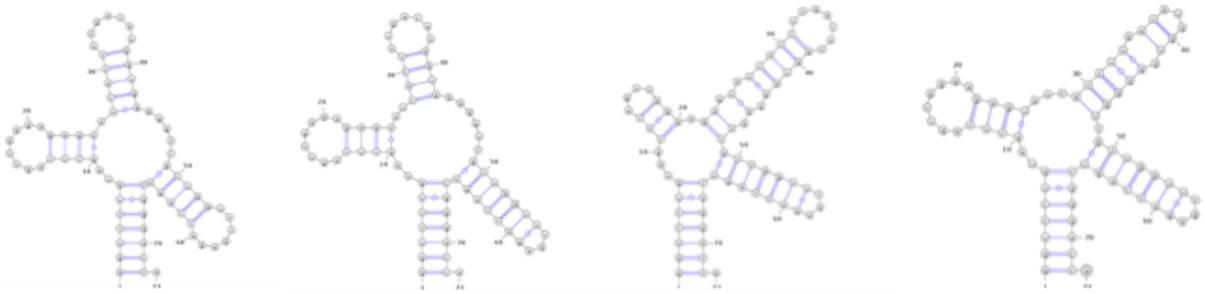

Figure 3: Two-dimensional structure predictions for tRNA corresponding to valine (codon: GAC) of species Listeria_monocytogenes_str._4b_F2365' with ID 'tdbD00011215'. Structures are presented in the same order as they are in Figure 2, starting with the true structure in the leftmost position, our prediction, the minimum free energy prediction, and the PKISS prediction.

thereby enabling our system to contain grammars specific to multiple RNA types. Steps to improve upon the generalizability of our system may involve training a single grammar across data sets with different RNA types. The complexities of balancing accuracy with generalizability is a task we look forward to tackling in subsequent works.

7. Project Remarks

The most challenging aspect of our project was implementing the Inside-Outside Algorithm. We implemented this algorithm using a parse chart to help us visualize the iteration process. However, even with the additional aid of a parse chart, the pattern of iteration was pretty difficult for us to translate into code. Once implemented, we found even then that we were not completely convinced that our implementation was working exactly as it should. We spent hours searching for bugs and revising this until it was working as intended. The easiest aspect of the project was probably the data-handling. Our data source, the tRNAdb, had an intuitive interface with a lot of data, all of which we could access in a FASTA format containing both the sequence and the structure in dot-parentheses notation. Once we parsed this data and stored it in a dictionary keyed on instance IDs, we were easily able to determine the base frequency data found in Table 1

In our original proposal we had intended to predict RNA structure using a multiple sequence alignment. We hoped to model our approach after that of Knudsen and Hein (Knudsen & Hein, 1999). However, we found that imitating their approach required having RNA sequences with structure information, a multiple sequence alignment, and phylogenetic data. We also found that the work of Knudsen and Hein did not go into great detail about the important implementation details necessary to replicate their work. Due to the lack of adequate information, both with respect to sequence data and implementation details, we simplified our project. We still used the grammar from Knudsen and Hein as a foundation for our own work, but we applied it to unaligned RNA sequences rather than multiple sequence alignments.

8. Acknowledgements

The authors would like to thank Professor Ameet Soni for his guidance and support throughout this project. We also thank Professor Rich Wicentowski for his assistance in the normalization of our grammar. And lastly, we thank Pro-

fessor Jeff Knerr for his technical assistance: specifically for his assistance with setting up both RNA-vienna and PKISS for our system evaluations.

References

Andronescu, Mirela, Bereg, Vera, Hoos, Holger H., and Condon, Anne.

Chomsky, Noam. On certain formal properties of grammars. *Information and Control*, 2(2):137–167, 1959.

Higgs, Paul G. *RNA secondary structure: phyiscal and computational aspects*. Cambridge University Press, 2000.

Janssen, Stefan and Robert Giegerich", title = The RNA shapes studio, journal = Bioinformatics year = 2014 volume = 31 number = 1 pages = 1–3.

Knudsen, B. and Hein, J. Rna secondary structure prediction using stochastic context-free grammars and evolutionary history. *Bioinformatics*, 15(6):446–454, 1999.

Knudsen, Bjarne and Hein, Jotun. Pfold: Rna secondary structure prediction using stochastic context-free grammars. *Nucleic Acids Research*, 31(13):3423–3428, 2003.

Rivas, Elena and Eddy, Sean R. The language of rna: a formal grammar that includes pseudoknots. *Bioinformatics*, 16(4):334–340, 1999.

Ruan, Jianhua, Stormo, Gary D., and Zhang, Weixiong. Ilm: a web server for predicting rna secondary structures with pseudoknots. *Nucleic Acids Research*, 32:146–149, 2004a.

Ruan, Jianhua, Stormo, Gary D., and Zhang, Weixiong. An iterated loop matching approach to the prediction of rna secondary structures with pseudoknots. *Bioinformatics*, 20(1):58–66, 2004b.

Sakakibara, Yasubumi, Brown, Michael, Hughey, Richard, Mian, Saira, Sjolander, Kimmen, Underwood, Rebecca C., and Haussler, David. Stochastic context-free grammars for trna modeling. *Nucleic Acids Research*, 22(23):5112–5120, 1994.

Sprinzl, M., Horn, C., Brown., M., Ioudovitch, A., and Steinberg, S. Compilation of trna sequences and sequences of trna genes. *Nucleic Acids Research*, 26:148–153, 1998.

Tabaska, Jack E., Cary, Robert B., Gabow, Harold N., and Stormo, Gary D. An rna folding method capable of identifying pseudoknots and base triples. *Bioinformatics*, 14 (8):691–699, 1996.

Zuker, Michael. Mfold web server for nucleic acid folding hybridization prediction. *Nucleic Acids Research*, 31(13):3406–3415, 2003.

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

A. Appendix A

Below we list the complete grammar that we developed for the prediction of tRNA secondary structure. The probability of each production rule is included in parentheses. Productions rules with a probability of 0 are omitted.

$S \rightarrow LS$ (0.9056)
$S \rightarrow O_a F_u$ (0.0058)
$S \rightarrow O_u F_a$ (0.0005)
$S \rightarrow O_g F_u$ (0.0062)
$S \rightarrow O_u F_g$ (0.0002)
$S \rightarrow O_g F_c$ (0.0134)
$S \rightarrow O_c F_g$ (0.0054)
$S \rightarrow s_a$ (0.0373)
$S \rightarrow s_g$ (0.0024)
$S \rightarrow s_c$ (0.0060)
$S \rightarrow s_u$ (0.0172)
$F \rightarrow LS$ (0.1962)
$F \rightarrow O_a F_u$ (0.0921)
$F \rightarrow O_u F_a$ (0.1150)
$F \rightarrow O_g F_u$ (0.0195)
$F \rightarrow O_u F_g$ (0.0164)
$F \rightarrow O_g F_c$ (0.3196)
$F \rightarrow O_c F_g$ (0.2412)
$L \rightarrow O_a F_u$ (0.0149)
$L \rightarrow O_u F_a$ (0.0143)
$L \rightarrow O_g F_u$ (0.0190)
$L \rightarrow O_u F_g$ (0.0017)
$L \rightarrow O_g F_c$ (0.0935)
$L \rightarrow O_c F_g$ (0.0248)
$L \rightarrow s_a$ (0.2022)
$L \rightarrow s_g$ (0.2154)
$L \rightarrow s_c$ (0.1654)
$L \rightarrow s_u$ (0.2488)
$F_a \rightarrow FC_a$ (1.0000)
$F_g \rightarrow FC_g$ (1.0000)
$F_c \rightarrow FC_c$ (1.0000)
$F_u \rightarrow FC_u$ (1.0000)
$O_a \rightarrow o_a$ (1.0000)
$O_g \rightarrow o_g$ (1.0000)
$O_c \rightarrow o_c$ (1.0000)
$O_u \rightarrow o_u$ (1.0000)
$C_a \rightarrow c_a$ (1.0000)
$C_g \rightarrow c_g$ (1.0000)
$C_c \rightarrow c_c$ (1.0000)
$C_u \rightarrow c_u$ (1.0000)