RNA Pseudoknot Modeling Using Intersections of Stochastic Context Free Grammars with Applications to Database Search

Michael Brown, Computer and Information Sciences
Charles Wilson, Department of Biology
University of California
Santa Cruz, CA 95064, USA

October 3, 1995

Abstract

A model based on intersections of stochastic context free grammars is presented to allow for the modeling of RNA pseudoknot structures. The model runs relatively fast, having the same order running time as stochastic context free grammar parsers. The model is shown to be able to perform database searches and find RNA sequences which resemble RNA pseudoknots which bind biotin. The problem domain of RNA biotin binders has significance in the support of the RNA world model of early life on earth.

1 Introduction

In light of the enormous generation of sequence data by projects such as the human genome project, the task of searching sequence databases has become increasingly important. The fields of computer science and biology have collaborated to design highly sensitive filters which can be used to sieve through this massive collection of data and return only those precious sequences for which the filters are designed.

The pseudoknot is an important structure of RNA that occurs in virtually all classes of RNA and has shown to be important in many biological functions [29]. Unfortunately, programs used to search large sequence databases for these pseudoknot structures in their general form seem to require running times that are prohibitive computationally. Examples are David Searls modeling of DNA repeats in any form [25] and Mamitsuka and Abe modeling beta sheets using Stochastic Ranked Node Rewriting Grammars [18].

This paper presents a model based on intersections of stochastic context free grammars which allows highly sensitive and relatively fast database searching for pseudoknotted RNA structures. It is shown that this model can

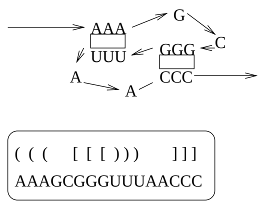


Figure 1: A pseudoknot.

discriminate RNA sequences containing a pseudoknotted structure for binding biotin from random RNA sequences which do not contain this structure and therefore can perform database searching.

The RNA pseudoknot is the representative problem domain for which our models provide a solution. A RNA pseudoknot is formed when bases outside a hairpin structure pair with bases within the hairpin loop to create a second stem and loop structure [5]. See figure 1. Pseudoknots have been found in small subunit ribosomal RNA and seem to be very important in its functioning. Pseudoknots have been observed in many structures including telemerases, 7SL RNA, U2 snRNA, group I introns, and viral RNA. Pseudoknots seem to have importance in messenger RNA's with frame shifts in gene expression and suppressors [29]. Pseudoknotted RNA also frequently occurs in *in vitro* RNA selection experiments. Recent examples include a RNA pseudoknot inhibitor to HIV-1 reverse transcriptase [10] and a pseudoknot binder of biotin [31].

Stochastic Context Free Grammars (SCFG's) form the basis of our modeling. Stochastic Context Free Grammars are an extension of Hidden Markov Models (HMM's) [20, 19] which have enjoyed much success in the modeling biological sequence structure [15, 14, 3, 2, 1, 28, 21, 13, 8]. Stochastic Context Free Grammars have been used to model mainly RNA structure. Recent examples include modeling tRNA [7, 23, 9, 17] and spliceosomal snRNA [30]. Both SCFG's and HMM's models are grounded in a rigorous probabilistic framework which captures in a cohesive framework aspects of sequence similarity such as sequence mutation, deletion, and insertion.

2 Methods

2.1 Stochastic Context Free Grammars

Stochastic Context Free Grammars are a probabilistic extension of Context Free Grammars from formal language theory. Formal language theory deals with sets of strings called languages and different mechanisms for generating and recognizing them [11]. A context free grammar, G, is defined as a 4-tuple: $G = (V, \Sigma, P, S)$ [12], where V is a set of nonterminals, Σ is the terminal alphabet, S is the start symbol, and P is a set of productions of the form $A \rightarrow \alpha$ where $A \in V$ and $\alpha \in (V \cup \Sigma)^*$.

A context free grammar generates strings through a rewrite process in which productions are used to rewrite nonterminal symbols with strings of nonterminal and terminal symbols starting from the start symbol. The sequence of rewrite productions used to derive a terminal string from the start symbol is called the string's derivation. A context free grammar can recognize a string using a parsing algorithm. The parsing algorithm given a context free grammar, G, and a string will return true if the string can be derived with G and false otherwise.

A SCFG extends the definition of context free grammars by associating a probability to every production in the grammar. Consequently every string that the grammar can generate is assigned a probability which is equal to the product of the probabilities of the productions used in the string's derivation.

Finding the probability of a observation sequence, O, given a SCFG, G can be done with an extension of the CYK parsing algorithm [12]. The CYK algorithm uses a table, $T[s, i, j]$, which stores the probability of nonterminal s deriving the subsequence, $O_i \ldots O_j$. A dynamic programming algorithm can be used to find the probability of the start symbol deriving the entire string in $O(n^3)$ time where n is the length of the observation sequence.

A correspondence can be made between a derivation in a SCFG and RNA secondary structure [23]. See figure 2. As the figure shows, there are nonterminals that generate basepairs, loop positions, and branch productions.

David Searls was the first to fully develop the connection between formal language theory and biosequence analysis [26, 27, 25]. Searls uses grammars that capture the linguistics of DNA such as start and stop codons, introns, and splice sites. Grammars give the flexibility to economically express the complex machinery of DNA.

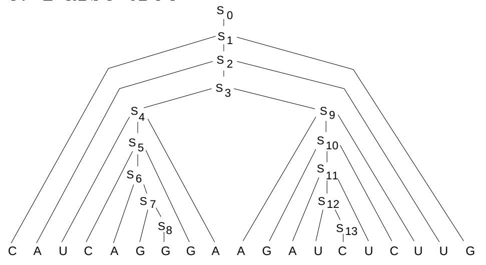
a. Productions

$$P = \{ \begin{array}{llll} S_0 \to S_1, & & S_7 \to \texttt{G}\ S_8, & \\ S_1 \to \texttt{C}\ S_2\ \texttt{G}, & & S_8 \to \texttt{G}, & \\ S_2 \to \texttt{A}\ S_3\ \texttt{U}, & & S_9 \to \texttt{A}\ S_{10}\ \texttt{U}, & \\ S_3 \to S_4\ \ S_9, & & S_{10} \to \texttt{G}\ S_{11}\ \texttt{C}, & \\ S_4 \to \texttt{U}\ S_5\ \texttt{A}, & & S_{11} \to \texttt{A}\ S_{12}\ \texttt{U}, & \\ S_5 \to \texttt{C}\ S_6\ \texttt{G}, & & S_{12} \to \texttt{U}\ S_{13}, & \\ S_6 \to \texttt{A}\ S_7, & & S_{13} \to \texttt{C} & \} \end{array}$$

b. Derivation

$$\begin{array}{ll} S_0 & \Rightarrow S_1 \Rightarrow \texttt{C}S_2\texttt{G} \Rightarrow \texttt{CA}S_3\texttt{UG} \Rightarrow \texttt{CA}S_4S_9\texttt{UG} \\ & \Rightarrow \texttt{CAU}S_5\texttt{A}S_9\texttt{UG} \Rightarrow \texttt{CAUC}S_6\texttt{GA}S_9\texttt{UG} \\ & \Rightarrow \texttt{CAUCA}S_7\texttt{GA}S_9\texttt{UG} \Rightarrow \texttt{CAUCAG}S_8\texttt{GA}S_9\texttt{UG} \\ & \Rightarrow \texttt{CAUCAGGGA}S_9\texttt{UG} \Rightarrow \texttt{CAUCAGGGAA}S_{10}\texttt{UUG} \\ & \Rightarrow \texttt{CAUCAGGGAAG}S_{11}\texttt{CUUG} \\ & \Rightarrow \texttt{CAUCAGGGAAGA}S_{12}\texttt{UCUUG} \\ & \Rightarrow \texttt{CAUCAGGGAAGAU}S_{13}\texttt{UCUUG} \\ & \Rightarrow \texttt{CAUCAGGGAAGAUCUCUUG}. \end{array}$$

c. Parse tree



d. Secondary Structure

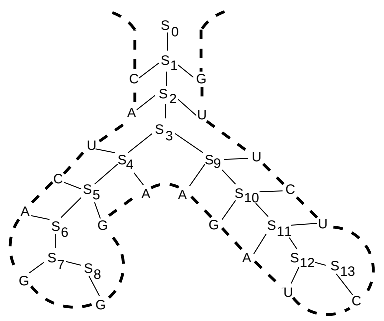


Figure 2: A simple CFG which may be used to derive a set of RNA molecules including the specific example illustrated here, `CAUCAGGGAAGAUCUCUUG`. *a.* A set of productions P which generates RNA sequences with a certain restricted structure. S_0 (start symbol), $S_1, \ldots, S_{13}$ are nonterminals; `A`, `U`, `G` and `C` are terminals representing the four nucleotides. *b.* Application of the productions P could generate the given sequence by the derivation indicated. For example, if the production $S_1 \to CS_2G$ is selected, the string CS_2G replaces S_1 and the derivation step is written $S_1 \Rightarrow CS_2G$. *c.* The derivation in *b* may be arranged in a tree structure called a *parse* or *derivation tree*. *d.* The physical secondary structure of the RNA sequence is a reflection of the parse tree (or syntactic structure).

2.2 Pseudoknot Modeling Problem and Solution

There are problems with modeling pseudoknots. The basic structure of a pseudoknot can be represented abstractly by the language, $\{w|w$ has the form $[^i(^j]^i)^j\}$ [1]. Here $[,]$ and $(,)$ denote abstract base-paired residues. This language is not context free by the pumping lemma [12]. There is NO context free grammar that can generate this language. However, consider the following language,

$$\{w|w \text{ has the form } [^i(^*]^i)^*\} \cap \{v|v \text{ has the form } [^*(^j]^*)^j\}$$

The intersection of these two languages is exactly the pseudoknot language.

Note that this argument uses context free languages without a stochastic component. When modeling RNA, we do not have the alphabet, $[,], (,)$ but rather the symbols, A, U, G, C which may either basepair or not. To see how this argument relates to stochastic context free grammars with the alphabet, A, U, G, C consider the language, $\{w|w$ has the form $A^iG^*U^iC^*\} \cap$ $\{v|v$ has the form $A^*G^jU^*C^j\}$. This language is the same except the alphabet has been changed. This can be interpreted as the language describing pseudoknots in which one helix is composed entirely of $A-U$ basepairs and the other helix is composed entirely of $G-C$ basepairs. The stochastic context free grammars which make up this intersection would have probability one for $A-U$ basepairs in one grammar and probability one for $G-C$ basepairs in the other. Obviously, in nature, there will rarely be instances in which a basepair will be $G-C$ with probability one. To make the model more realistic, relax the distribution on basepairs to allow for any basepair (each having its own respective probability). Now a sequence can be classified as a pseudoknot if it can be identified as having one helix which has high probability under one statistical model and simultaneously having the other helix which has high probability under the other statistical model.

This intersection argument is important for efficiency reasons. Consider designing a non-context free pseudoknot grammar that could generate the pseudoknot language. Normally, in a context free grammar, a nonterminal derives a subword, $i \dots j$. However, in pseudoknots, the derivations overlap because the helices are not properly nested and a nonterminal must derive the "subword", $(B \dots b)(e \dots E)$ where $B \dots b$ and $e \dots E$ are not contiguous in the string. In order to determine the running time of a CYK-like parser for

[1]The terminology $]^i$ represents the symbol, $]$ repeated i times. $]^4 =]]]]$

such a language consider the running time of context free grammar parser. The context free grammar parser must fill in a table, $T[s, i, j]$, which is the probability of the nonterminal s deriving the subword $i \ldots j$. Assume that there are T nonterminals and the length of the string to be parsed is N. There are order (TN^2) entries to be filled. The pseudoknot grammar parser, on the other hand, must fill in a table, $T[s, B, b, e, E]$, in which there are order (TN^4) entries. Assuming that $T \approx N$ then the running time is at least $O(N^5)$. This is prohibitive computationally. However, because of the intersection argument used above, the pseudoknot language can be approximated by using four $O(n^3)$ context free parsing operations. The sense of this approximation is described below.

2.3 Approximating a Pseudoknot

Consider a pseudoknot with two helixes, $S1$ and $S2$, and three loops, $L1, L2, L3$, placed between the helixes. See figure 3. Note that this paper deals with problems in which $L1, L2$, and $L3$ are simple loops. However, in general, $L1, L2$, and $L3$ can be arbitrarily complex stochastic context free grammars containing multiple stem loop complexes. A full pseudoknot grammar can be decomposed into two different context free grammars which will be called the 5' grammar and the 3' grammar. The 5' grammar breaks the helix closest to the 3' end, keeping the 5' helix intact. The 3' grammar breaks the helix closest to the 5' end, keeping the 3' helix intact. See figure 4. Breaking a helix involves converting a basepairing production like $s \rightarrow AU$ into two productions like $g \rightarrow A$ and $h \rightarrow U$ in which the two halves of the basepair are derived independently. If the probabilities on basepairs for the nonterminal s is given in $P[i, j]$ where $i, j \in \{$A,U,G,C$\}$, then the probabilities for g and h should be marginal distributions. Probabilities for g will be given by $P[i] = \sum_j P[i, j]$ while the probabilities for h will be given by $P[j] = \sum_i P[i, j]$.

In this way, the pseudoknot grammar can be decomposed into two SCFG's and allow $O(n^3)$ parsing algorithms to be used to align the model instead of the $O(n^5)$ operation which would be required if the full pseudoknot grammar were used.

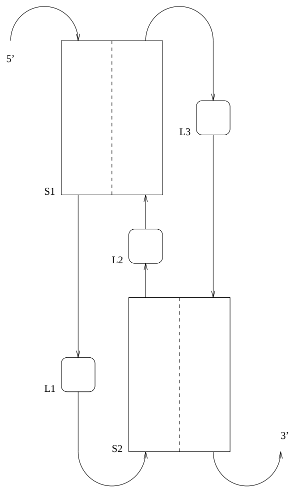


Figure 3: A pseudoknot. Rounded boxes represent loops and rectangles represent helixes.

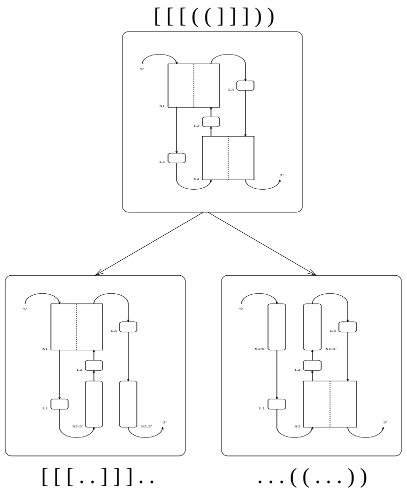


Figure 4: A pseudoknot broken into two grammars.

2.4 Finding Probabilities and Predicted Structures of Sequences

In order to perform a database search, a probability for each of the sequences must be computed given the model. The true probability of the sequence under the full pseudoknot grammar still appears to be difficult to compute exactly in $O(n^3)$ time. Therefore we compute a Viterbi-like approximation to this true probability. This approximate probability is computed in a seven stage process.

1. Compute the probability of the sequence under the 5' grammar storing that probability and the location of the 5' helix.

2. Compute the probability of the sequence under the 3' grammar storing that probability and the location of the 3' helix.

3. Compute the conditional probability of the sequence under the 5' grammar given that the 3' broken helix approximation is forced to be in the position specified by the 3' grammar parse. Store the location of the 5' helix under this constrained parse.

4. Compute the final 5' conditional probability by using the the locations of the 3' and 5' helixes to compute the probability under the full pseudoknot grammar.

5. Compute the conditional probability of the sequence under the 3' grammar given that the 5' broken helix approximation is forced to be in the position specified by the 5' grammar parse. Store the location of the 3' helix under this constrained parse.

6. Compute the final 3' conditional probability by using the the locations of the 3' and 5' helixes to compute the probability under the full pseudoknot grammar.

7. Return the largest of the final 3' and 5' conditional probabilities as the true probability of the sequence under the pseudoknot grammar.

Note that this procedure uses the "efficient" SCFG parses to place the constituent grammar features and then uses these placements to score the sequence under the full pseudoknot grammar. The predicted structure of the sequence can be easily constructed once parsing has finished. Simply take the parse of the most likely grammar and record where the helixes and loops are placed. This gives the complete predicted structure of the pseudoknot.

2.5 Database Searching Using the Pseudoknot Model

In order to perform a database search for pseudoknots we must, for each sequence in the database, classify the sequence as being generated by the pseudoknot model or as being generated by a competing NULL model [6, 22].

This classification can be done using Bayes decision theory by considering the following function:

$$g(sequence) = \log \frac{P(sequence|pseudoknot)}{P(sequence|NULL)} + \log \frac{P(pseudoknot)}{P(NULL)}$$

where $P(sequence|pseudoknot)$ is the probability of the sequence under the pseudoknot model as approximated in the procedure described in the previous section, $P(sequence|NULL)$ is the probability of the sequence under the null model which is simply a uniform multinomial distribution over residues, $P(pseudoknot)$ is the prior probability of the pseudoknot model, and $P(NULL)$ is the prior probability of the null model.

If the function g is bigger than zero then the *pseudoknot* model is the "best" hypothesis otherwise the $NULL$ model is "better". For fixed priors, this reduces to computing the first term, $\log \frac{P(sequence|pseudoknot)}{P(sequence|NULL)}$ which is equal to $(-\log P(sequence|NULL) + \log P(sequence|pseudoknot))$.

This is interpreted as the savings in bits of the pseudoknot model over the $NULL$ model in encoding the sequence. This savings in bits in encoding a sequence will be called the *score* of the sequence. High scores imply the sequence was more likely generated by the *pseudoknot* model while low scores imply the sequence was more likely generated by the $NULL$ model.

If we knew, *a priori*, the probability of the pseudoknot model and of the NULL model, we could use $g(sequence)$ to perform a database search by calculating the first term by computing the probabilities using the models, finding the second term which is a cutoff value using prior information, and then adding them to see how their sum compares to zero. Unfortunately our prior knowledge does not allow us to find the second term accurately. Instead, we use an empirical method which estimates the second term. This estimation is done by computing the scores of many random sequences. We estimate the cutoff value used in the function g as the savings in bits such that all the random sequences have scores less than that number.

Therefore to perform a database search on a long string (or even an entire genome), the string is broken into overlapping windows. For our software, a window size of 128 runs the fastest. As described above, a score is computed for each of these windows and a cutoff value is applied so that only those windows which have a high enough score are accepted. After this cutoff is made, the predicted structure for all accepted sequences is generated and analyzed.

2.6 Parameter Estimation of the Pseudoknot Model

Estimating parameters of a pseudoknot grammar using intersections of SCFG's involves combining the estimates from the two constituent SCFG's into estimates for the pseudoknot grammar and then generating two SCFG's from these combined counts. These estimates are based on frequencies of occurrence of various events, such as the number of times a certain production was used.

Computing these estimates for SCFG's can be performed with an algorithm called the inside/outside algorithm [16]. Both the inside and outside variables can be computed in $O(n^3)$ time. These variables can be used to estimate the probability of a production being used. A more computationally efficient SCFG parameter estimation technique has been developed [24]. This algorithm is called the Tree-Grammar EM training algorithm. This algorithm uses labeled trees as input rather than observation sequences. These labeled trees represent "folded" sequences which are a set of trees in which the frontier, or leaves, of the tree are known but the internal nodes are not. The algorithm is able to estimate parameters of a grammar given the trees in $O(n^2)$ time.

Once these estimates are computed for the constituent SCFG's, the question is how to combine the counts. One heuristic that is used is as follows: For loop regions, simply combine the counts for the respective loop regions from the two SCFG's. The loop regions should coincide between the two SCFG's so they should share parameters and therefore the estimates are added. For broken helixes, counts from the broken helix grammar and marginal counts from the full helix are combined. The broken helix is represented as a broken helix in one grammar and as one half of a full helix in the other grammar. Therefore the full counts from the broken helix and the marginal counts from the full helix are combined. For full helixes, the counts for the full helix are used while the counts for broken helix are not. For full helixes, we found that the broken helix approximation did not produce as good estimates as the full helix representation and therefore estimates from the broken helix approximation are not used. Once estimate counts have been combined, two SCFG's are produced, each of which approximates one of the helixes in the pseudoknot with independently derived helix approximations. This heuristic has been implemented as an extension to the Tree-Grammar EM training algorithm.

3 Results

3.1 Biotin Binder Pseudoknot

Our methods were first tested on modeling RNA sequences which contained pseudoknots which bind biotin [31]. These sequences are important because of their potential support of the 'RNA World' model [4]. The RNA World model proposes that much of modern metabolism evolved prior to the evolution of encoded protein synthesis, and that early ribozyme-catalyzed metabolic transformations form the basis of our present protein-catalyzed metabolism. This proposal requires that ribozymes should be able to catalyze a broad range of chemical transformations. The search for "new" ribozymes leads us to the biotin binder RNA sequences. These sequences were generated from *in vitro* selection experiments. A pool of approximately 5×10^{14} different random sequence RNA's was generated by *in vitro* transcription of a DNA template containing a central 72-nucleotide random sequence region, flanked at both ends by 20-nucleotide constant regions. On average, any given 28 nucleotide sequence has a 50 % probability of being represented in a pool of this complexity. This pool was applied to an agarose column derivatized with 2-6 mM biotin and then washed with 15 column volumes of binding buffer. Specifically-bound RNA's were then eluted by washing the column with binding buffer containing 5 mM biotin. Reverse transcription of the specifically eluted RNA, followed by PCR amplification and *in vitro* transcription yielded and enriched pool of RNA for subsequent re-selection.

```
   ((((( [[[[[[[)))))                              ]]]]]]]
GGCACCGACCATAGGCTCGGGTTGCCAGAGGTTCCACACTTTCATCGAAAAGCCTATGCTA
```

Figure 5: A pseudoknot for binding biotin.

After several rounds of selection, the RNA was sequenced. The biotin binder structure is short and contains one pseudoknot. See figure 5 for a representative sequence.

A model for the biotin binder was constructed by analyzing the sequences derived from the *in vitro* selection experiments and several mutational experiments. This data was used to construct the overall structure of the model. The biotin binder sequences were then used as a training set and parameters of the model were estimated using an extension of the Tree-Grammar EM training algorithm [23]. This model was then hand tuned using biological knowledge to capture the structure of the biotin binder sequence without over-fitting the biases which were present in the training sequences.

In order to judge significance of sequence scores, the model was used to compute the scores of 4096 random sequences which had lengths uniformly distributed over $60 \ldots 79$ and had a uniform distribution over bases. The average score of these random sequences was -6.03836 with a variance of 4.9239. The highest score of a random sequence was 16.571716.

The model's discriminative power was first tested by computing the scores of the biotin binding sequences which were generated by the *in vitro* selection experiments. The highest score of a biotin binder was 31.919823. The lowest score was 18.493336. See figure 6 for a histogram of scores for both the random and biotin binding sequences. This figure shows that choosing a cutoff between 16.58 and 18.49 (almost 2 bits of separation between random and biotin binding sequences) will discriminate between biotin binders and random sequences. This cutoff would correspond to the *a priori* belief that there is approximately one biotin binder for every 2^{17} sequences in a database search.

Our database search involved searching through all GenBank database sequences which had the keyword "biotin". These sequences were broken into overlapping windows of length 76 yielding 1737 windows. Scores were computed for each of the windows and a cutoff of 16.75 was chosen. Only

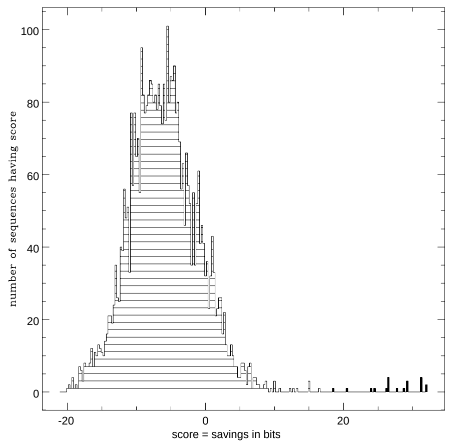


Figure 6: A histogram showing number of sequences having a certain sequence score. Dark areas are biotin binders. Light areas are random sequences.

one window had a score above this cutoff. The score of this window is 17.023933. This window corresponded to the sequence for the gene for biotin carboxyl carrier protein in Mycobacterium leprae (EMBL: MLBCCPG. Accession: X63470).

The pseudoknotted structure started at position 449 in this sequence (the position of the 5' base in the first basepair) and has the following predicted structure:

```
     [[[[     {{{{]]]]      }}}}
cacaatcgatccgcgacctcggcgacaaggtcaccgc
```

This sequence's predicted structure is similar to other biotin binding RNA. To test whether this sequence could bind biotin, an experiment similar to the *in vitro* selection process described above was performed. A pool made up of this sequence was applied to a column derivatized with biotin and then washed with binding buffer. Then the column was washed with biding buffer containing biotin to elute the specifically-bound RNA. Unfortunately there was not any significant binding of the RNA and attempts to optimize the experiment by changing temperature and Mg concentration did not yield any more promising results.

3.2 Importance of Intersected SCFG's

The pseudoknot structure cannot be modeled by one SCFG alone. The processes of constraining the parses in the intersection yields discriminative power. In order to show this, the 5' and 3' grammars were not intersected but were used alone to perform discrimination. Again, as with the intersected model, a cutoff value must be chosen. This was done by scoring random sequences described in the previous section and recording the highest scoring sequences. For the 5' grammar the highest scoring random sequence was 14.012839. For the 3' grammar the highest scoring random sequence was 15.061969. These two numbers serve as the cutoff values for the two respective grammars.

The biotin binding sequences derived from *in vitro* selection were then analyzed. Whereas the intersected model was able to completely discriminate these sequences from random by choosing a cutoff anywhere between 16.58 and 18.49 (as much as a 2 bit difference), the 5' and 3' models alone could not completely discriminate these sequences. The 5' model had ten sequences below the cutoff. The highest scoring of these was 13.630875 (0.4 bit below cutoff). The 3' model did much better having only one sequence with a score below the cutoff. That score was 14.672070 (0.3 bit below cutoff). This shows that the 3' model contains much of the information as captured by the intersected model, but not enough to discriminate true biotin binding sequences. The intersection process must be used to get maximum discrimination.

4 Conclusion

A new method for modeling RNA pseudoknots using intersections of stochastic context free grammars has been presented. This method is novel in it's use of intersected SCFG's for approximating the location of the pseudoknot. The model has successfully discriminated biotin binding RNA from random RNA sequences and has found in a database search a sequence which closely resembles biotin binders derived from *in vitro* selection experiments. The pseudoknot model gives discriminative power which is greater than simple stochastic context free grammars when modeling pseudoknots.

References

[1] P. Baldi, S. Brunak, Y. Chauvin, J. Engelbrecht, and A. Krogh. Hidden Markov models of human genes. In J.Cowan, G. Tesauro, and J. Alspector, editors, *Advances in Neural Information Processing Systems*, volume 6, pages 761–768. Morgan Kaufmann, 1994.

[2] P. Baldi and Y. Chauvin. Smooth on-line learning algorithms for hidden Markov models. *Neural Computation*, 6(2):305–316, 1994.

[3] P. Baldi, Y. Chauvin, T. Hunkapillar, and M. McClure. Hidden Markov models of biological primary sequence information. *PNAS*, 91:1059–1063, 1994.

[4] S. Benner, A. Ellington, and A. Tauer. Modern metabolism as a palimpsest of the rna world. *Proc. of the Nat. Acad. of Sci.*, 86:7054–8, 1989.

[5] J.-H. Chen, S.-Y. Le, and J. V. Maizel. A procedure for rna pseudoknot prediction. *CABIOS*, 8:243–248, 1992.

[6] R. O. Duda and P. E. Hart. *Pattern Classification and Scene Analysis.* Wiley, New York, 1973.

[7] S. R. Eddy and R. Durbin. RNA sequence analysis using covariance models. *NAR*, 22:2079–2088, 1994.

[8] Y. Fujiwara, M. Asogawa, and A. Konagaya. Stochastic motif extraction using hidden markov model. In *ISMB-94*, pages 121–129, 1994.

[9] L. Grate, M. Herbster, R. Hughey, I. Mian, H. Noller, and D. Haussler. RNA modeling using Gibbs sampling and stochastic context free grammars. In *ISMB-94*, Menlo Park, CA, Aug. 1994. AAAI/MIT Press.

[10] L. Green, S. Waugh, J. Binkley, Z. Hostomska, Z. Hostomsky, and C. Tuerk. Comprehensive chemical modification interference and nucleotide substitution analysis of an rna pseudoknot inhibitor to hiv-1 reverse transcriptase. *JMB*, 247:60–68, 1995.

[11] M. A. Harrison. *Introduction to Formal Language Theory.* Addison Wesley, 1978.

[12] J. Hopcroft and J. Ullman. *Introduction to Automata Theory, Languages and Computation.* Addison-Wesley, 1979.

[13] K. Karplus. Using Markov models and hidden Markov models to find repetitive extragenic palindromic sequences in *Escherichia coli.* Technical Report UCSC-CRL-94-24, UCSC, July 1994.

[14] A. Krogh, M. Brown, I. S. Mian, K. Sjölander, and D. Haussler. Hidden Markov models in computational biology: Applications to protein modeling. *JMB*, 235:1501–1531, Feb. 1994.

[15] A. Krogh, I. S. Mian, and D. Haussler. A Hidden Markov Model that finds genes in *E. coli* DNA. *NAR*, 22:4768–4778, 1994.

[16] K. Lari and S. J. Young. The estimation of stochastic context-free grammars using the inside-outside algorithm. *Comp. Speech and Lang.*, 4:35–56, 1990.

[17] F. Lefebvre. An optimized parsing algorithm well suited to rna folding. In *ISMB-1995*, 1995.

[18] H. Mamitsuka and N. Abe. Predicting location and structure of beta-sheet regions using stochastic tree grammars. In *ISMB-94*, pages 276–284, 1994.

[19] L. R. Rabiner. A tutorial on hidden Markov models and selected applications in speech recognition. *Proc. of the IEEE*, 77(2):257–286, Feb. 1989.

[20] L. R. Rabiner and B. H. Juang. An introduction to hidden Markov models. *IEEE ASSP Magazine*, 3(1):4–16, Jan. 1986.

[21] R. Raman and C. Overton. Application of hidden Markov modeling to the characterization of transcription factor binding sites. In *Proc. 27th Hawaii Int. Conf. on System Sciences*, pages 275–283. IEEE Computer Society Press, 1994.

[22] J. Rissanen. *Stochastic Complexity in Statistical Inquiry.* World Scientific, 1989.

[23] Y. Sakakibara, M. Brown, R. Hughey, I. S. Mian, K. Sjölander, R. C. Underwood, and D. Haussler. Stochastic context-free grammars for tRNA modeling. *NAR*, 22:5112–5120, 1994.

[24] Y. Sakakibara, M. Brown, R. Hughey, I. S. Mian, K. Sjölander, R. C. Underwood, and D. Haussler. Recent methods for RNA modeling using stochastic context-free grammars. In *Proc. Asilomar Conf. on Combinatorial Pattern Matching*, New York, NY, 1994. Springer-Verlag.

[25] D. B. Searls. The linguistics of DNA. *American Scientist*, 80:579–591, Nov.–Dec. 1992.

[26] D. B. Searls. The computational linguistics of biological sequences. In L. Hunter, editor, *Artificial Intelligence and Molecular Biology*, chapter 2, pages 47–120. AAAI Press, 1993.

[27] D. B. Searls and S. Dong. A syntactic pattern recognition system for DNA sequences. In H. A. Lim, J. Fickett, C. R. Cantor, and R. J. Robbins, editors, *Proc. of the 2nd Int. Conf. on Bioinformatics, Supercomputing, and Complex Genome Analysis*, pages 89–101. World Scientific, 1993.

[28] H. Tanaka, M. Ishikawa, K. Asai, and A. Konagaya. Hidden Markov models and iterative aligners. In *ISMB-93*, pages 395–401, Menlo Park, 1993. AAAI Press.

[29] E. ten Dam, K. Pleij, and D. Draper. Structural and functional aspects of rna pseudoknots. *Biochemistry*, 31:11665–11676, 1992.

[30] R. C. Underwood. Stochastic context-free grammars for modeling three spliceosomal small nuclear ribonucleic acids. Master's thesis, University of California, Santa Cruz, 1994.

[31] C. Wilson and J. Szostak. In vitro evolution of a self-alkylating ribozyme. *Nature*, 374:777–782, 1995.