Stochastic context-free grammars for tRNA modeling

Yasubumi Sakakibara+, Michael Brown, Richard Hughey[1], I.Saira Mian[2], Kimmen Sjölander, Rebecca C.Underwood and David Haussler*
Computer and Information Sciences, [1]Computer Engineering and [2]Sinsheimer Laboratories, University of California, Santa Cruz, CA 95064, USA

Received April 29, 1994; Revised October 12, 1994 and Accepted October 13, 1994

ABSTRACT

Stochastic context-free grammars (SCFGs) are applied to the problems of folding, aligning and modeling families of tRNA sequences. SCFGs capture the sequences' common primary and secondary structure and generalize the hidden Markov models (HMMs) used in related work on protein and DNA. Results show that after having been trained on as few as 20 tRNA sequences from only two tRNA subfamilies (mitochondrial and cytoplasmic), the model can discern general tRNA from similar-length RNA sequences of other kinds, can find secondary structure of new tRNA sequences, and can produce multiple alignments of large sets of tRNA sequences. Our results suggest potential improvements in the alignments of the D- and T-domains in some mitochdondrial tRNAs that cannot be fit into the canonical secondary structure.

INTRODUCTION

Both computer science and molecular biology are evolving rapidly as disciplines, and predicting the structure, function and folding of macromolecules by theoretical or experimental means remains a challenging problem. Determining common or consensus patterns among a family of sequences, producing a multiple sequence alignment, discriminating members of the family from non-members and discovering new members of the family will continue to be some of the most important and fundamental tasks in mathematical analysis and comparison of macromolecular sequences (1, 2). Here we examine RNA from the perspective of formal language theory using computational linguistics to encompass various *structural* phenomena observed in RNA rather than any information it might encode. The general approach is highly related to our earlier work on using Hidden Markov Models (HMMs) to model protein families and domains (3, 4) and to parse *E.coli* contigs into genes that code for protein separated by intergenic regions (5).The HMM architecture we used (4) captures the intuition of a family of related sequences in terms of: a) a sequence of common positions each with its own distribution over the residues; b) the possibility for either skipping a position or inserting extra residues between consecutive positions; and c) allowing for the possibility that continuing an insertion or deletion is more likely than starting one. All the parameters in the HMM, i.e., the transition probabilities and the residue distributions, are learned entirely automatically from a set of *unaligned* primary sequences. These features are also present in the stochastic models used in this paper. Although in principle HMMs could be used to describe a family of RNA sequences, a major limitation of HMMs is that all positions are treated as having independent, non-interacting distributions. Clearly, this is unsuitable for RNA because if two positions are base-paired, the bases at these positions are likely to be highly correlated.

Here we describe a method for analyzing a family of RNA sequences that uses formal language theory to generalize HMMs to model most RNA interactions. The technique is applied to the problems of statistical modeling, multiple sequence alignment, discrimination and prediction of the secondary structure of RNA families in general and tRNA in particular. Tools for performing these functions are increasingly important as *in vitro* evolution and selection techniques produce greater numbers of synthesized RNA families to supplement those related by phylogeny. To date, two principal methods have been established for predicting RNA secondary structure: phylogenetic analysis of homologous RNA molecules (6, 7, 8) and thermodynamics (9, 10, 11, 12). When several related sequences are available that all share a common secondary structure, combinations of such approaches have been used to obtain improved results (13, 14, 15, 16, 17, 18, 19, 20). Several groups have enumerated schemes or programs to search for patterns in proteins or nucleic acid sequences (21, 22, 23, 24, 25, 26, 27, 28). String pattern-matching programs based on the UNIX **grep** search for secondary structure elements in a sequence database (29, 30). If there is prior knowledge about sequence and structural aspects of an RNA family, this can be employed to create a *descriptor* (discriminating pattern) for the family, which can then be used for database searching or generating an alignment for the family. This has been demonstrated most clearly for tRNA (31, 32, 33), where approximate string matching proved to be important. Our method of descriptor construction, multiple alignment and folding differs markedly from conventional techniques because it builds a

*To whom correspondence should be addressed

+Present address: ISIS, Fujitsu Labs Ltd., 140, Miyamoto, Numazu, Shizuoka 410-03, Japan statistical model *during* rather than *after* the process of alignment and folding, and the resultant model is itself a descriptor for the RNA family.

Recently, Searls has argued the benefits of viewing the character strings representing DNA, RNA and protein as sentences derived from a formal grammar (34). The simplest kind of grammar is a *regular* grammar, in which strings are derived from productions (rewriting rules) of the forms $S \rightarrow aS$ and $S \rightarrow a$, where S is a *nonterminal symbol* which does not appear in the final string, and a is a *terminal symbol*, which appears as a letter in the final string. *Context-free grammars* (CFGs) allow a broader class of productions which are crucial to model the base-pairing structure in RNA. CFGs allow for productions of the form $S \rightarrow$ **A** S **U** , $S \rightarrow$ **U** S **A** , $S \rightarrow$ **G** S **C** and $S \rightarrow$ **C** S **G**, which describe the structure in RNA due to Watson–Crick base pairing. Using productions of this type, a CFG can specify the language of biological palindromes. Although CFGs cannot describe all RNA structure, they can account for enough to make useful models. (Currently, pseudoknots, base triples involving three positions, and interactions in parallel [versus the more usual anti-parallel] are not modeled.)

A stochastic grammar specifies a probability for each production in the grammar and thus assigns a probability to each string (sequence) it derives. Stochastic *regular* grammars are equivalent to HMMs and suggest an interesting generalization from HMMs to *stochastic context-free grammars* (SCFGs) (35). Searls has proposed some analogs of stochastic grammars and methods for creating the grammar from training sequences in the form of costs and other trainable parameters used during parsing (36, 37) (Searls, unpublished manuscript). We have developed an integrated probabilistic framework for estimating the parameters of an SCFG which may prove to be a simpler and more effective approach. Our method is closely related to the 'covariance models' (CMs) of Eddy and Durbin (29). CMs are equivalent to SCFGs, but the algorithms we use for training are different. An in-depth comparison of the two methods is given elsewhere (38, 39).

In this paper, we focus on validating the use of SCFGs to model RNA by applying them to tRNA. We create a statistical model of tRNA by generating a SCFG incorporating base-pairing information in a manner similar to our construction of an HMM (4). This model is used to discriminate tRNA from other RNAs of similar length, obtain a multiple sequence alignment and ascertain the secondary structure of new tRNAs. The results indicate that the RNA SCFG model appears to be a good discriminator. A detailed examination of the SCFG-produced secondary structures and alignments indicates that they agree extremely well with previously determined ones (40) and in some cases suggest improvements. Further details on our methods and results can be found in (38, 39).

METHODS

Overview of SCFGs

A context-free language is simply a set of sequences of characters drawn from an alphabet and specified through a representation called a *grammar*. Formally, a grammar is composed of three parts. The first is a finite alphabet Σ of *terminal* symbols. For RNA sequences, this alphabet comprises the nucleotides **A**, **U**, **G** and **C**. The second is a finite set N of nonterminal symbols $S_1, \ldots, S_n$ and a special *start* symbol S_0. The third is a set P of rewrite rules (or *productions*) that specify how sequences containing nonterminals may be rewritten by expanding those embedded nonterminals to new subsequences. The language represented by the grammar is the set of all sequences of terminal symbols that can be derived from the start symbol S_0 by repeatedly applying productions from P.

Let S denote any nonterminal symbol, a denote any terminal symbol, and α denote any sequence of terminal and nonterminal symbols. Each production has the form $S \rightarrow \alpha$, indicating that the nonterminal S can be replaced by the sequence α. See Figure 1a for a small set of possible productions in a grammar representing a set of RNA sequences having a common structure. These productions can be classified into four types. $S \rightarrow aSa$ describe base-pairs (for example, $S \rightarrow GSC$ represents a G-C base-pair). $S \rightarrow aS$ and $S \rightarrow a$ describe unpaired bases. $S \rightarrow S$, called *skip productions*, are equivalent to deletions, in that their use allows no nucleotide to appear at that position in a multiple alignment. These productions are also employed to omit substructures. For example, adding the production $S_3 \rightarrow S_4$ to the grammar shown in Figure 1a allows the option of deriving a string that does not contain the right arm shown in Figure 1d. Finally, $S \rightarrow SS$ productions describe branched secondary structures. For example, in the example grammar, the production $S_3 \rightarrow S_4\ S_9$ derives a branched structure, the left branch derived by productions starting from nonterminal S_4, and the right branch derived by productions starting from nonterminal S_9.

A *derivation* is a rewriting of a sequence using the rules of the grammar. It begins with a sequence that consists only of the start symbol S_0. In each step of the derivation, a nonterminal from the current sequence is chosen and replaced with the right-hand side of a production for that nonterminal. This replacement process is repeated until the sequence consists of terminal symbols only. A simple derivation is shown in Figure 1b.

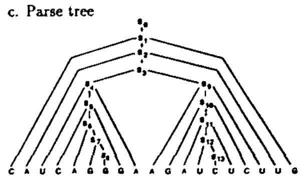
a. Productions

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

b. Derivation

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

c. Parse tree



d. Secondary Structure

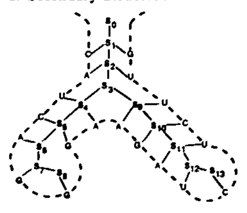


Figure 1. A simple CFG which may be used to derive a set of RNA molecules including the specific example illustrated here, CAUCAGGGAAGAUCUCUUG. **a.** A set of productions P which generates RNA sequences with a certain restricted structure. S_0 (start symbol), $S_1, \ldots, S_{13}$ are nonterminals; **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 $S_1 \rightarrow C\ S_2\ G$ 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).

For every derivation there exists a corresponding parse tree, representing the syntactic structure of the sequence produced by the derivation (see Figure 1c). For RNA, this corresponds to the physical secondary structure (solid lines in Figure 1d). The primary sequence of the derived RNA may be obtained by tracing the letters from left to right along the frontier of the tree (dashed line in Figure 1d). The secondary structure can be seen in the links between nucleotides that are derived from base-pairing productions. Contiguous stretches of these base pairs are helices. The nesting structure of the helices in Figure 1d is apparent in the derivation: reading the sequence left-to-right, the bottom two helices are clearly nested within the topmost helix, giving the nesting '(()())'.(As mentioned above, CFGs can only represent secondary structure with properly nested helices; they cannot represent pseudoknots. When we model a family with pseudoknots, these are currently ignored.)

In a SCFG, every production for a nonterminal S has an associated probability value such that a probability distribution exists over the set of productions for S. These probabilities are used to define a probability for each parse tree. The probability of a parse tree is the product of the probabilities of the production instances applied to produce it. Thus in an SCFG, one can think of a parse tree as the result of a stochastic process in which each nonterminal symbol is replaced independently at random according to a specified distribution over possible replacements for that symbol. Further details on SCFGs can be found in the tutorial paper (41).

Finding the best parse

CFGs are generally ambiguous because a grammar may give more than one parse tree for a sequence and alternative parse trees reflect alternative secondary structures (foldings). In a SCFG, this ambiguity can be resolved by selecting the parse tree with the highest probability, which we call the *most likely parse tree*. There is a well-known dynamic-programming procedure that, for SCFGs of the type we use, can determine the most likely parse tree for a sequence s in time proportional to the cube of the length of s (41). We call it the modified CYK algorithm, because it is a variant of the CYK algorithm used to parse context-free grammars (42). Finding the most likely parse tree using this CYK variant is similar to the use of the Viterbi algorithm to find the most likely path through an HMM (43). The algorithm is also used in (29), and is related to other dynamic programming algorithms used to fold RNA sequences (44, 45).

To predict the secondary structure for a sequence s in the family modeled by an SCFG, we find the most likely parse tree for s with the modified CYK algorithm, and then predict that s has the secondary structure represented by that parse tree. The output of the modified CYK algorithm is designed so we can display this predicted secondary structure graphically using XRNA, an X Windows-based program for editing and display of RNA primary, secondary and tertiary structure (46). See (39) for examples. The modified CYK algorithm can also be used to obtain multiple sequence alignments just as the Viterbi algorithm (43) can for HMMs (4). Key nonterminal symbols (called *match nonterminals*) are identified that each derive either a single letter or two base-paired letters in the consensus structure common to the RNA family being modeled. Each match nonterminal is associated with a column (or pair of columns) in the multiple alignment that corresponds the position of its letter(s) in the consensus structure. To produce the multiple alignment, we find the most likely parse tree for each sequence s individually, and then print s such that the letter(s) derived from each match nonterminal that occurs in the parse tree of s appears in the column(s) associated with that match nonterminal, inserting dashes and deleting extra inserted characters as necessary.

Negative log likelihood of a sequence

A stochastic context free grammar G also defines a probability distribution over all sequences that can be derived from G. The probability of a sequence s is the sum of probabilities of all possible parse trees for s that can be obtained from the grammar G. This probability is denoted Prob $(s \mid G)$. It can be calculated using a variant of the modified CYK algorithm known as the *inside* algorithm, analogous to the *forward* algorithm used to calculate the probability of a sequence given an HMM model (43). This procedure is described in (41). The time for this procedure is also cubic in the length of the sequence. We call the quantity $-\log(\text{Prob }(s \mid G))$ the *negative log likelihood* (NLL) *score* of the sequence s. The NLL score quantifies how well the sequence s fits the grammar—how likely it is that the grammar would produce the sequence s. Thus it is natural to use this score to discriminate sequences in the family modeled by the grammar from those not in the family. However, this raw NLL score is too dependent on the test sequence's length to be used directly in discrimination. Hence, it is normalized to produce a *Z-score*: the difference between the NLL score of a sequence and the average NLL score of a typical RNA sequence of the same length *not* in the family, measured in standard deviations (4). We use the Z-score to discriminate tRNA sequences from non-tRNA sequences by choosing a Z-score cutoff and classifying sequences above the cutoff as tRNAs. Although it is not clear that the normalized scores actually exhibit Gaussian tails for non-tRNAs, this kind of Gaussian approximation has worked for HMMs (4).

Estimating the parameters of an SCFG

There is a standard method for estimating the parameters of an SCFG (i.e. the probabilities of the productions) from a set of training sequences. This procedure is known as the *inside–outside* algorithm (41). Just like the forward–backward algorithm for HMMs, this procedure is an expectation-maximization (EM) method for obtaining either maximum likelihood or maximum *a posteriori* estimates of the grammar's parameters. Because each iteration of the main loop in the

```
1. Start with an initial grammar $G_0$.
2. Use grammar $G_0$ and the modified CYK algorithm to parse the raw
input sequences, producing a tree representation for each sequence
indicating which nucleotides are base-paired. This set of initial trees
is denoted $T_0$. Set $T_{old} = \emptyset$ and $T_{new} = T_0$.
3. While $T_{new} \neq T_{old}$ do the following: {
    3a. Set $T_{old} = T_{new}$.
    3b. Use $T_{old}$ trees as input to the Tree-Grammar Reestimator
    algorithm, which iteratively re-estimates the grammar parame-
    ters until they stabilize. The grammar with the final stabilized
    probability values is called new grammar $G_{new}$.
    3c. Use grammar $G_{new}$ and the modified CYK algorithm to parse
    the input sequences, producing a new set of trees $T_{new}$.
}
```

Figure 2. Pseudocode for the Tree-Grammar EM training algorithm used to estimate the parameters of a SCFG from unaligned training RNA sequences. After designing an appropriate rough initial grammar, training sequences are employed only to refine estimates of the probabilities of the productions used in the grammar.

			Number of Sequences			
Data Set	Type of tRNA	Total	ZeroTrain	MT10CY10	MT100	RandomTRNA618
ARCHAE	archaea	103	0	0	0	50
CY	cytoplasm	230	0	10	0	100
CYANELCHLORO	cyanelle and chloroplast	184	0	0	0	100
EUBACT	eubacteria	201	0	0	0	100
VIRUS	viruses	24	0	0	0	10
MT	mitochondria	422	0	10	100	200
PART III	Part III	58	0	0	0	58
	Totals	1222	0	20	100	618

Figure 3. Details on the tRNA sequences used for training and testing the grammars. Sequences in the first six groups can be fitted into a cloverleaf structure but those in PART III have 'unusual' secondary structures. This group include tRNAs from mitochondria of parasitic worms (no T- or D-domain), mammalian mitochondria (no D-domain), mitochondria of mollusc, insect and echinoderm (extended anticodon and T-stems), single cell organisms and fungi and *Trypanosoma brucei*. The **ZeroTrain** grammar is a control that has not been trained on any sequences and only has prior information on tRNA. The three trained grammars **MT10CY10**, **MT100** and **RandomTRNA618** contain randomly chosen subsets of sequences from the various groups (**RandomTRNA618** has all sequences in PART III).

```
          [     ]  <     D-domain     > <   Anticodon   >< Extra ><    T-domain   >[      ]
          (((((((  ((((              )))) (((((  ===  )))))          (((((        ))))))))))))
1 DC0380  -GCCAAGGTGGCAGAGTTCGGCCTAACGCGGCGGCCTGCAGAGCCGCTC----ATCGCCGGTTCAAATCCGGCCCTTGGCT---
2 DA6281  -GGGCGTGTGGCGTAGTC-GGT--AGCGCGCTCCCTTAGCATGGGAGAG----GTCTCCGGTTCGATTCCGGACTCGTCCA---
3 DE2180  --GCCCCATCGTCTAGA--GGCCTAGGACACCTCCCTTTCACGGAGGCG----A-CGGGGATTCGAATTCCCCTGGGGGTA---
4 DC2440  -GGCGGCATAGCCAAGC--GGT--AAGGCCGTGGATTGCAAATCCTCTA----TTCCCCAGTTCAAATCTGGGTGCCGCCT---
5 DK1141  -GTCTGATTAGCGCAACT-GGC--AGAGCAACTGACTCTTAATCAGTGG----GTTGTGGGTTCGATTCCCACATCAGGCACCA
6 DA0260  -GGGCGAATAGTGTCAGC-GGG--AGCACACCAGACTTGCAATCTGGTA----G-GGAGGGTTCGAGTCCCTCTTTGTCCACCA
7 DA3880  -GGGGCTATAGTTTAACT-GGT--AAAACGGCGATTTTGCATATCGTTA----T-TTCAGGATCGAGTCCTGATAACTCCA---
8 DH4640  -AGCTTTGTAGTTTATGTG-----AAAATGCTTGTTTGTGATATGAGTGAAAT-------------------TGGAGCTT---

          (((((((  ((((               )))) (((((  ===  )))))         (((((        ))))))))))))
1 DC0380  -GCCAAGGUGGCAG.AGUUcGGccUAACGCGGCGGCCUGCAGAGCCGCUC---AUCGCCGGUUCAAAUCCGGCCCUUGGCU---
2 DA6281  -GGGCGUGUGGCGU.AGUC.GG..UAGCGCGCUCCCUUAGCAUGGGAGAGG---UCUCCGGUUCGAUUCCGGACUCGUCCA---
3 DE2180  -GCCCC-AUCGUCU.AGAG.GCc.UAGGACACCUCCCUUUCACGGAGGCG----ACGGGGAUUCGAAUUCCCCU-GGGGGU--A
4 DC2440  -GGCGGCAUAGCCA.AGC-.GG..UAAGGCCGUGGAUUGCAAAUCCUCUA---UUCCCCAGUUCAAAUCUGGGUGCCGCCU---
5 DK1141  -GUCUGAUUAGCGC.AACU.GG..CAGAGCAACUGACUCUUAAUCAGUGGG---UUGUGGGUUCGAUUCCCACAUCAGGCACCA
6 DA0260  -GGGCGAAUAGUGUcAGCG.GG..-AGCACACCAGACUUGCAAUCUGGUA----GGGAGGGUUCGAGUCCCUCUUUGUCCACCA
7 DA3880  -GGGGCUAUAGUUU.AACU.GG..UAAAACGGCGAUUUUGCAUAUCGUUA----UUUCAGGAUCGAGUCCUGAUAACUCCA---
8 DH4640  -AGCUUUGUAGUUU.A--U.GU..GAAAAUGCUUGUUUGUGAUAUGAGUGA--AAU-----------------UGGAGCUU---
```

Figure 4. Comparison of the alignment of several representative tRNAs produced by trained grammar RandomTRNA618 (bottom) with that from **EMBLTRNA** (40) (top). Parentheses indicate base-paired positions; = = = the anticodon; and '[ ]' the 5′ and 3′ sides of the acceptor helix. For **RandomTRNA618**, capital letters correspond to nucleotides aligned to the match non-terminals of the grammar, lower case to insertions, – to deletions by skip productions and . to fill characters required for insertions. The sequences are taken from the seven groups in Figure 3 and are denoted by their databank codes: 1. ARCHAE (*Halobacterium cutirubrum*), 2. CY (*Saccharomyces cerevisiae*), 3. CYANELCHLORO (*Cyanophora paradoxa*), 4. CYANELCHLORO (*Chlamydomonas reinhardtii*), 5. EUBACT (*Mycoplasma capricolum*), 6. VIRUS (phage T5), 7. MT (*Aspergillus nidulans*) and 8. PART III (*Ascaris suum*).

procedure takes time cubic in the length of each training sequence, we have developed an alternate, faster method to estimate the parameters of our SCFGs, summarised in Figure 2. The inner loop, Tree-Grammar Reestimator (Step 3b), requires folded RNA sequences as training examples, rather than unfolded ones. Thus, some tentative 'base pairs' in each training sequence have to be identified before Tree-Grammar Reestimator can begin. To do this, we design a rough initial grammar (Step 1) that may represent only a portion of the base-pairing interactions, and parse the unfolded RNA training sequences using this grammar to obtain a set of partially folded RNA sequences (Step 2). Then we reestimate the SCFG parameters using the partially folded sequences and Tree-Grammar Reestimator (Step 3b); following this we refold the sequences using the refined parameters (Step

Sequence Set	ZeroTrain	MT10CY10	MT100	RandomTRNA618
ARCHAE	94.87%	100.00%	100.00%	100.00%
CY	98.28%	99.76%	99.89%	99.87%
CYANELCHLORO	96.22%	99.64%	99.64%	99.79%
EUBACT	99.69%	99.86%	99.86%	99.86%
VIRUS	96.83%	100.00%	100.00%	100.00%
MT	89.19%	98.33%	98.91%	98.93%
PART III	55.98%	81.10%	83.21%	83.00%

Figure 5. Fraction of base pairs specified by the **EMBLTRNA** alignment that matched in our grammars' predicted multiple alignments (**ZeroTrain** shows statistics for the pre-training initial grammar).

3c). In this way, Tree-Grammar Reestimator can be used even when precise biological knowledge of the base pairing is not available. The Tree-Grammar Reestimator algorithm iteratively finds the best parse for each sequence in the training set and then readjusts the production probabilities to maximize the probability of these parses. This takes only time linear in the sequence length per sequence per iteration, with the number of iterations typically on the order of 100. Step 3c takes time cubic in the length of each sequence, but is typically executed only 2 or 3 times for each sequence.

Data, training and testing

The sequences and alignments of 1477 tRNAs were taken from the database of Sprinzl and co-workers (40) (referred to as EMBLTRNA). Modified bases were converted to their unmodified form. Removal of duplicate sequences left a total of 1222 unique sequences between 51 and 93 bases in length for use in testing and training (Figure 3). For the discrimination experiments, we generated 2016 non-tRNA test sequences from the non-tRNA features (including mRNA, rRNA, and CDS) in NewGenBank 75.0$^+$ and GenBank75.0. We created 20 non-tRNA sequences for each sequence length between 20 and 120 bases. (The original size of the data set was 2020 because we discarded four anomalous tRNAs that appeared in this set of non-tRNAs through unusual labeling in GenBank. We discovered these when the trained grammars 'misclassified' them in discrimination experiments.)

The Tree-Grammar EM algorithm was used to reestimate the probability parameters in an initial grammar using varying numbers of unfolded and unaligned tRNA training sequences (Figure 3). For 100 training sequences, the run time was approximately 3–4 hours on a Sun Sparc station 10/30. Finding the most likely parse for each sequence given a grammar (Step 3c) required 2–3 CPU seconds for a typical tRNA sequence on a DEC AXP 3000/400 running OSF/1.

We examined the ability of each of the four grammars, **ZeroTrain**, **MT10CY10**, **MT100**, and **RandomTRNA618**, to perform three tasks: to discriminate tRNAs from non-tRNAs, to produce multiple sequence alignments and to ascertain the secondary structure of new sequences.

RESULTS

Multiple alignments and secondary structure

For each of the four grammars shown in Figure 3, a multiple sequence alignment was produced for all 1222 tRNAs. Figure 4 shows results for the best trained grammar, **RandomTRNA618**. The predicted alignment agrees substantially

```
            [     ]            <   D-domain    >   <   Anticodon domain    >< Extra  ><    T-domain    >[     ]
Base pairing(((((((           ((((         ))))   (((((                ))))))          (((((        ))))))))))))
 1 TRUSTED  GGGCUAU---------uaGCUCagcgguagagcgc--gCGCCC---------cugauaaGGGCG-----aggucUCUGG-uucaaauCCAGGAUAGCCCa---
   MT100    GGGCUAU---------uaGCUC--agcgguagagcgcgCGCCC---------cugauaaGGGCG-----aggucUCUGG-uucaaauCCAGGAUAGCCCa---
 2 TRUSTED  GCCCCUA---------uaGUUG---aaacacaacc--aAGAGC---------uuuucacGCUCU-----uaaguUUGAG-uuaaaauCUCAAUAGGAGCu---
   MT100    GCCCCUA---------uaGUUG----aaacacaac-caAGAGC---------uuuucacGCUCU-----uaaguUUGAG-uuaaaauCUCAAUAGGAGCu---
 3 TRUSTED  GUUUCAU---------gaGUAU-----agcaGUAC--aUUCGG---------cuuccaaCCGAA-----agguuuuugu-aaacaacCAAAAAUGAAAUa---
   MT100    GUUUCAU---------gaGUAU-----agcaGUAC--aUUCGG---------cuuccaaCCGAA------agguuuuuguaaacaacCAAAAAUGAAAUa---
 4 TRUSTED  aggacgu---------uaaaua---gauaagCUAU--gCCUAG---------uuacgguCUGGG---aagagag------------------ucgucuuu---
   MT100    aggacgu--------uaaauag----auaagCUAU--gCCUAG---------uuacgguCUGGG---aagagag------------------ucgucuuu---
   MT10CY10 ag-gacg-------uuaaauag----auaagCUAU--gCCUAG---------uuacgguCUGGG--aagagagu------------------cguc-uuu---
 5 TRUSTED  aacgagu----------ucaua-----------aa--gCAAGU---------cuucuaaAUUUG------uucu-agg---uuaaau--ccugcucguuu---
   MT100    aacgagu----ucauaaa-------------------gCAAGU---------cuucuaaAUUUG------uucu--agg--uuaaauccu--gcucguuu---
   RND618   aacga-g---uucauaaa-------------------gCAAGU---------cuucuaaAUUUG-------uuc-uagg--uuaaauccug-c-ucguuu---
 6 TRUSTED  AAGAAAG------------------------auug--cAAGAA---------cugcuaaUUCAU---gcuuccaug-uu--uaaaaaCAUGGCUUUCUUa---
   MT100    AAGAAAG-------auug-------------------cAAGAA---------cugcuaaUUCAU------gcuuccauguuuaaaaaCAUGGCUUUCUUa---
 7 TRUSTED  GAGAAAG------------------------cuca--caagaa---------cugcuaacucau---gcccccaug-uc--uaacaaCAUGGCUUUCUCacca
   MT100    GAGAAAG-------cuca-------------------caagaa---------cugcuaacucau------gcccccaugucuaacaaCAUGGCUUUCUCacca
   RND618   GAGAAAG--------cuc-------------------acaaga-------acugcuaacucaug-------cccccaugucuaacaaCAUGGCUUUCUCacca
 8 TRUSTED  -aaaucu-----------auu-----gguuuaccu---UAGUC---------cugcuaaGUCUA---aaggcuugcggu-ucaaucccguugaguuuuc----
   MT100    aaaucua---------uuggu--------uu-acc--uUAGUC---------cugcuaaGUCUA----aaggcuugcgg-uucaaucccguugaguuuuc---
 9 TRUSTED  GAAAUAU-----------guu------gauc-aag---AAAAG---------cugcuaaCUUUU----ucuuuaauggu-uuaauuccauuauauuucu-cca
   MT100    GAAAUAU----------g-uug-------aucaa---gAAAAG---------cugcuaaCUUUU-----ucuuuaaugg-uuuaauuccauuauauuucucca
   MT10CY10 GAAAUAU----------guug--------au-caa--gAAAAG---------cugcuaaCUUUU-----ucuuuaaugg-uuuaauuccauuauauuucucca
10 TRUSTED  GAAAAAG----------ucaug---gaggccaugg--gGUUGG---------cuugaaaCCAGC------uuugGGGGG-uucgauuCCUUCCUUUUUUg---
   MT100    GAAAAAG---------ucaugg------aggccaugggGUUGG---------cuugaaaCCAGC------uuugGGGGG-uucgauuCCUUCCUUUUUUg---
11 TRUSTED  AAAAUUA---------uauauu---uucuaguuug--aucgaa---------aaugcuuuucgauuugaaaauuuaaau-uaaauuuAAGUUUAAUUUUc---
   MT100    AAAAUUA---------uauauuuucuaguuugauc--gaaaaugcuuuucgauuugaaaauuua------aauuaaauu-------uAAGUUUAAUUUUc---
   MT10CY10 AAAAUUAuauauuuucuaguuu---gaucgaaaau--gcuuuu---------cgauuugaaaau---uuaaauuaaauu-------uAAGUUUAAUUUUc---
```

		Sequence set		Anticodon and Organism
1	DI2620	CYANELCHLORO	Chloroplast	GAT COLEOCHAETE ORBIC.
2	DE5080	MT	Animal mitochondria	TTC STRONGYLOCEN.PURP.
3	DG5000	MT	Animal mitochondria	TCC ASTERINA PECTINI.
4	DR4640	PART III	Parasitic worm mitochondrial tRNAs lacking T-domain	ACG ASCARIS SUUM
5	DS4680	PART III	Parasitic worm mitochondrial tRNAs lacking D-domain	TCT CAENORHABDI. ELEG.
6	DS5321	PART III	Mammalian mitochondrial tRNAs (anticodon GCU) lacking D-domain	GCT MOUSE
7	RS5880	PART III	Mammalian mitochondrial tRNAs (anticodon GCU) lacking D-domain	GCU HUMAN
8	DS5041	PART III	Mollusc, insect and echinoderm mitochondrial tRNAs with extended anticodon and T-stems	GCT PARACENTROTUS LIV.
9	RS4800	PART III	Mollusc, insect and echinoderm mitochondrial tRNAs with extended anticodon and T-stems	GCU AEDES ALBOPICTUS
10	DS5880	PART III	Mammalian mitochondrial serine tRNAs/tDNA sequences with anticodon UGA/TGA	TGA HUMAN
11	DA3681	PART III	Sequence for which the secondary structure is especially unusual or is not established	TGC TRYPANOSOMA BRUCEI

Figure 6. Alignment of selected tRNAs (top) where the grammar-predicted alignments suggest (small) improvements over those in **EMBLtRNA**. **EMBLtRNA** (59) is shown first, followed by alignments produced by **MT100**, **MT10CY10** and **RandomTRNA618** (in cases where grammars produced identical alignments, only one example is shown). Residues in lower case in helical regions denote positions where the **EMBLtRNA** and grammar alignments differ (unpaired nucleotides are also depicted in lower case and are not shown aligned). Shown below is information on the sequences where 'Sequence set' is one of the groups listed in Figure 5. Using grammars **MT100** and **MT10CY10** only sequences 1–3 and 9–10 can be discriminated from non- tRNA (Z-score > 5) whereas the rest cannot (Z-score < 4). With **RandomTRNA618** 1–3 and 10 can be discriminated, 4–7 cannot and the remainder have Z-scores between 4 and 5.

with the **EMBLtRNA** alignment (40): boundaries of helices and loops are the same and the major difference is the extra arm, which is highly variable in its length and sequence.

To assess the accuracy of the predicted foldings, we computed the percentage of base pairs specified by the **EMBLtRNA** alignment that are also present in the secondary structure predicted by the grammar (Figure 5). In the sequence sets ARCHAE and VIRUS, every one of the three trained grammars captures all the base pairing present in **EMBLtRNA**. In the case of CY, CYANELCHLORO, EUBACT and MT, the agreement between **EMBLtRNA** and grammar-predicted base pairings is extremely good, but for PART III it is considerably poorer.

We examined in detail all cases where base pairing in alignments produced by **MT10CY10**, **MT100**, and **RandomTRNA618** had less than perfect agreement with base pairs specified by **EMBLtRNA** (data not shown). One EUBACT, two CYANELCHLORO, 12 MT and 30 PART III sequences were 'misaligned' by all three grammars. Disagreements with **EMBLtRNA** are not distributed uniformly across the entire length of the sequence but are confined to specific helices. In some sequences, the misalignment merely reflects differences in location of a gap between **EMBLtRNA** and grammar alignments on one or both sides of a helix. Other instances are examples of alternative, but equally plausible, base-pairing schemes in the various helices.

However, there are cases where the grammar-generated alignments suggest (small) improvements over the **EMBLtRNA** alignments, principally in the base pairing of the D- or T-helices. A selection of such sequences is shown in Figure 6. A notable example are the Part III mammalian mitochondrial tRNAs lacking the D-domain, and mollusc, insect and echinoderm mitochondrial tRNAs with extended anticodons and T-stems. Here, the grammar-generated alignment suggests a readjustment of residues in the 5′ side of the T-helix, and flanking unpaired residues, creating a T-stem with a greater number of Watson–Crick base pairs than in **EMBLtRNA**.

Discriminating tRNAs from non-tRNAs

To test the ability of our grammars to discriminate tRNA from other RNA sequences of similar length, for each of our trained grammars, we computed the Z-score of every sequence in our tRNA database and every sequence in our set of 2016 non-tRNAs. Although the highest Z-score of any non-tRNA is never much greater than 4, we do not consider a tRNA sequence to be succesfully discriminated from the non-tRNAs unless its Z-score is greater than 5. For each grammar, Figure 7 shows the number of tRNAs in each family that are successfully discriminated from the non-tRNAs using this criterion and Figure 8 gives histograms of the Z-scores for selected grammars.

Training on as few as 20 sequences yields a dramatic improvement in discrimination over what is achieved with an untrained grammar (compare histograms for the grammars **ZeroTrain** and **MT10CY10**, which was trained on ten CY and ten MT sequences). **MT10CY10** is able to perfectly discriminate between non-tRNA sequences and tRNA excluding MT and Part III sequences (the top pair of Z-scores histograms), where the **ZeroTrain** grammar fails. The **MT10CY10** grammar also does well in the more difficult task of discriminating MT tRNA from non-tRNA. Setting aside the Part III sequences, **MT10CY10** is able to discriminate 399 out of 422 mitochondrial sequences from non-tRNA, performing nearly as well as the grammars trained on many more tRNA sequences, **MT100** and **RandomTRNA618**. However, good discrimination of the Part III sequences from non-tRNA sequences is not achieved by any of the grammars, even the **RandomTRNA618** grammar, which is trained on these sequences. Training improves discrimination of some Part III sequences, but half of these sequences still have Z-scores below 5.

We examined all tRNA sequences that scored below the Z-score cutoff of 5 by some trained grammar or that were incorrectly aligned with respect to the **EMBLtRNA** alignment by all three trained grammars (data not shown). A total of 29 Part III tRNAs could not be discriminated from non-tRNA sequences by either **MT100**, **MT10CY10** or **RandomTRNA618** (8 of these have a Z-score between 4 and 5 in at least one grammar). Interestingly, none of the three grammars were able to successfully discriminate or align identically to **EMBLtRNA** 19 of the 29 Part III tRNAs. All but three of these sequences are mammalian and parasitic tRNAs that lack the D-domain. However, the grammars are able to discriminate Part III tRNAs lacking the T-domain. Overall, the trained grammars are able to generalize well in that they require few training examples to perform discrimination. Results from Part III tRNAs demonstrate that a grammar gains discriminative power from being trained on a large and varied sequence set.

Number of Sequences with Z-Scores

	Above 5 Standard Dev.				Between 4 and 5 Std. Dev.				Below 4 Standard Dev.			
Sequence Set	ZT	MT10	MT100	R618	ZT	MT10	MT100	R618	ZT	MT10	MT100	R618
ARCHAE	66	103	103	103	19	0	0	0	18	0	0	0
CY	135	230	230	230	53	0	0	0	42	0	0	0
CYANELCHLORO	61	184	184	184	52	0	0	0	71	0	0	0
EUBACT	160	201	201	201	30	0	0	0	11	0	0	0
VIRUS	16	24	24	24	4	0	0	0	4	0	0	0
MT (train)	N/A	10	99	193	N/A	0	1	6	N/A	0	0	1
MT (test)	64	389	313	218	89	10	7	3	269	13	2	1
Part III	0	9	7	29	1	15	14	8	57	34	37	21
NON-tRNA	0	0	0	0	0	0	1	1	2016	2016	2015	2015
Totals	502	1150	1161	1182	248	25	23	18	2488	2063	2054	2038

Z-Scores for Boundary Sequences

	ZeroTrain	MT10CY10	MT100	RandomTRNA618
Highest NON-tRNA	3.954	3.341	4.018	4.080
Lowest non-MT non-Part III tRNA	1.220	6.791	6.211	8.759
Group of the lowest tRNA	CYANELCHLORO	CYANELCHLORO	CY	CY

Figure 7. The partitioning of 3238 total sequences based on their Z-scores by the four grammars (**ZeroTrain** is abbreviated as ZT, **MT10CY10** as MT10 and **RandomTRNA618** as R618). The sum of each grammar's three 'Totals' entries is 3238 (1222 tRNA and 2016 non-tRNA). The first grouping of four columns indicates the number of tRNAs correctly discriminated from non-tRNA by each grammar. (Because all three grammars perfectly discriminated all non-MT tRNA sequences, only the results for MT tRNA sequences are partitioned into separate discrimination results for the training and test sets.) The bottom table shows the Z-scores for the highest-scoring non-tRNA sequence and lowest-scoring tRNA sequence (excluding the MT and Part III tRNA sequences), listing the group to which the lowest-scoring tRNA belongs, for each grammar.

DISCUSSION

Formal language theory has been proposed as a means for examining RNA (39, 38, 34, 29). Here we have applied SCFGs to tRNA and shown that they provide a flexible and highly effective statistical method for solving a number of RNA sequence analysis problems including discrimination, multiple alignment and prediction of secondary structures. We used our Tree-Grammar EM algorithm (38, 39) to derive several *trained grammars* from different training sets of tRNA sequences. These grammars are able to discriminate regular mitochondrial tRNA from non-tRNAs quite well. However, only 50% of the Part III tRNAs can be reliably distinguished from non-tRNAs by even our most heavily trained grammars. Here 'reliably distinguished' means having a score that is more than 5 standard deviations from that of a typical non-tRNA of the same length. The majority of the sequences that could not be discriminated are parasitic worm and mammalian mitochondrial tRNAs lacking the D-domain. In addition, these sequences cannot be aligned in the same manner as **EMBLtRNA**. However, inspection of their alignments indicates that a revision around the T-domain would create a T-stem with a greater number of Watson–Crick base pairs than in **EMBLtRNA**. In contrast, Part III mitochondrial sequences lacking the T-domain *can* be both discriminated from non-tRNAs and aligned identically to **EMBLtRNA**. Our results suggest potential improvements in the alignments of the D- and T-domains in mitochondrial tRNAs from parasitic worms and mammals that lack the D-domain, and mollusc, insect and echinoderm tRNAs with extended T-stems. This work demonstrates the usefulness of SCFGs with tRNA sequences and could prove useful in maintaining, updating and revising compilations of their alignments. Further classes of RNA sequences potentially appropriate to model using this method include group I introns (47, 48), group II introns (49), RNAse P RNA (50, 51), small nuclear RNAs (52) and 7S RNA (signal recognition particle RNA) (53).

SCFGs may be valuable tools for representing an RNA family or domain. The main difficulties in applying this work to other families of RNA will be the development of appropriate initial grammars and the computational cost of parsing longer sequences. The latter problem can only be solved by the development of fundamentally different parsing methods, perhaps relying more

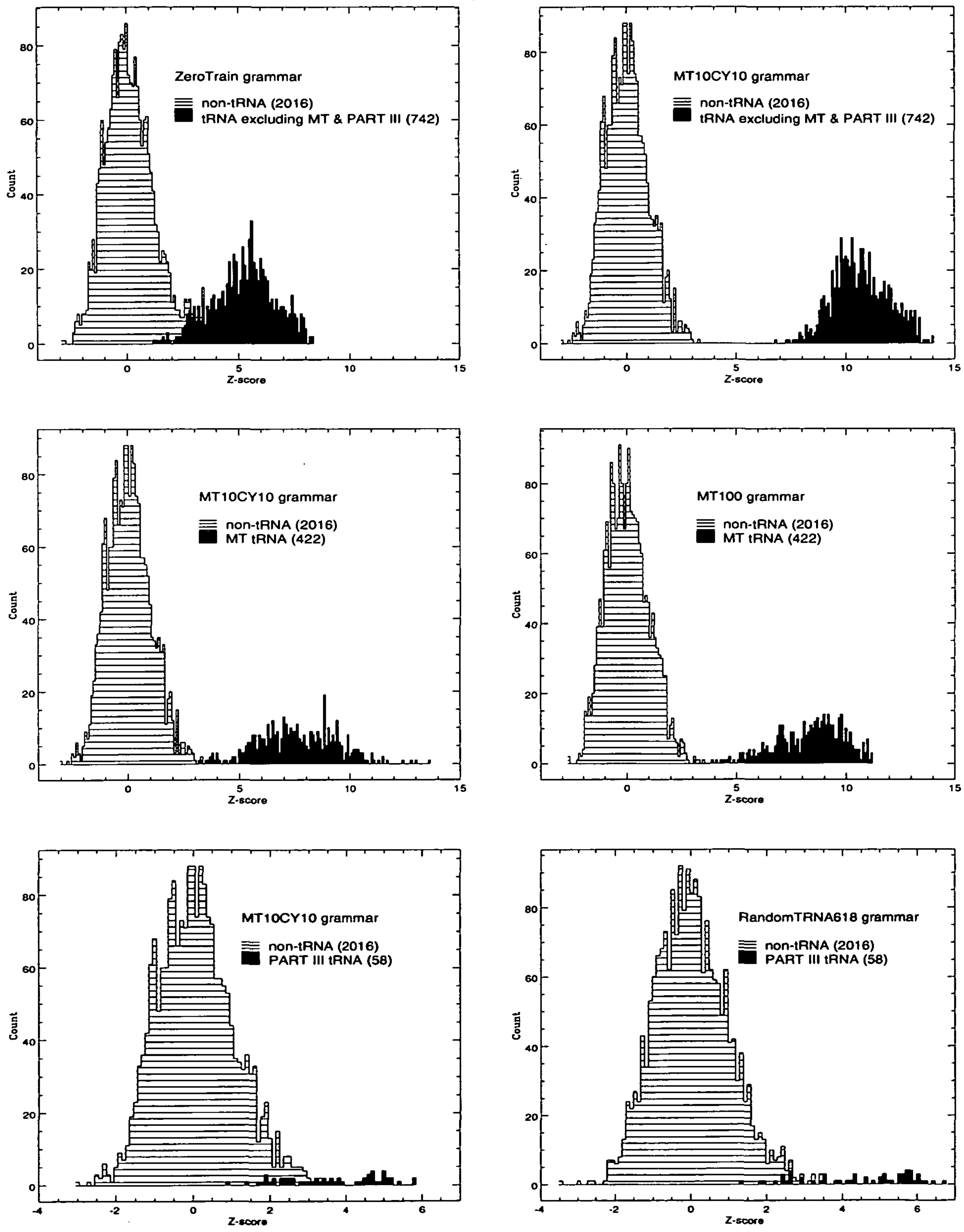


Figure 8. Histograms showing results of discrimination experiments with various grammars.

on branch-and-bound methods (54) or heuristics. It is currently not clear which approach will be best. The former problem might be solved by the development of effective methods for learning the grammar itself from training sequences. The work of Eddy and Durbin is an important step in this direction (29). Their method relies on correlations between columns in a multiple alignment (20, 19, 13, 18, 17, 45) to discover the essential base-pairing structure in an RNA family. Another approach would be to use a method like that proposed by Waterman (13) to find helices in a rough initial multiple alignment, use these helices to design a simple initial grammar in a semi-automated fashion using a high-level RNA grammar specification language, then use the grammar to obtain a better multiple alignment, and iterate this process until a suitable result is obtained (55).

Another important direction for further research is the development of stochastic grammars for tRNA and other RNA families that can be used to search databases for these structures at the DNA level. In order to do this, the grammar must be modified to allow for the possibility of introns in the sequence, and the parsing method modified so that it can efficiently search for RNAs that are embedded within larger sequences. Durbin and Eddy have implemented the latter modifications in their tRNA experiments and report good results in searching the GenBank structural RNA database and 2.2 Mb of *C.elegans* genomic sequence for tRNAs, even without using special intron models. In our earlier work (56), we reported very preliminary results on modifying tRNA grammars to accommodate introns. Since there is significant 'information content' in tRNA at both the primary sequence and secondary structure level, it might be possible to scan database sequences for these consensus sequences, perhaps using a conventional weight matix at low threshold, and then use the grammar on isolated hits rather than uniformly over the whole sequence. Although such a preprocessing approach could prune the search space greatly making the parsing more tractable, it may not be applicable to RNAs more complex than tRNA. Overall, we see no insurmountable obstacles in developing effective stochastic grammar-based search methods, but predict that the main practical problem will be dealing with the long computation time required by the present methods.

Finally, there is the question of what further generalizations of HMMs beyond SCFGs might be useful. The key advantage of our method over the HMM method is that it allows us to explicitly deal with RNA secondary structure. By extending stochastic models of strings to stochastic models of trees, we can model the base-pairing interactions of the molecule which determine its secondary structure. This progression is similar to the path taken by the late King Sun Fu and colleagues in their development of the field of syntactic pattern recognition (57). Modeling pseudoknots and higher-order structure would require still more general methods. One possibility would be to consider *stochastic graph grammars* (see the introductory survey by Engelfriet and Rozenberg [58]) in hopes of obtaining a more general model of the interactions present in the molecule beyond the primary structure. If a stochastic graph grammar framework could be developed that included both an efficient method of finding the most probable folding of the molecule given the grammar and an efficient EM method for estimating the grammar's parameters from folded examples, then extensions of our approach to more challenging problems, including RNA tertiary structure determination and protein folding, would be possible. This is perhaps the most interesting direction for future research suggested by the results of this paper.

ACKNOWLEDGEMENTS

We thank Anders Krogh, Harry Noller and Bryn Weiser for discussions and assistance, Michael Waterman and David Searls for discussions and anonymous referees for their suggestions. We also thank Sergey Steinberg, Daniel Gautheret, Robert Cedergren, and Mathias Sprinzl for providing us with their unpublished alignments of tRNA and tRNA gene sequences (59). This work was supported by NSF grants CDA-9115268 and IRI-9123692 and NIH grant number GM17129. This material is based upon work supported under a National Science Foundation Graduate Research Fellowship.

Readers may obtain this paper, its data and multiple alignments, and related papers via anonymous ftp from **ftp.cse.ucsc.edu in /pub/rna**.

REFERENCES

1. J. E. Dahlberg and J. N. Abelson, (ed.) (1989) *RNA processing. Part A. General Methods, Methods in Enzymology* **180**. Academic Press, New York.
2. R. F. Doolittle, (ed.) (1990) *Molecular Evolution, Methods in Enzymology* **183**. Academic Press, New York.
3. Haussler, D., Krogh, A., Mian, I. S., and Sjölander, K. (1993) In Proceedings of the Hawaii International Conference on System Sciences 1. Los Alamitos, CA: IEEE Computer Society Press. pp. 792–802.
4. Krogh, A., Brown, M., Mian, I. S., Sjölander, K., and Haussler, D. (1994) *Journal of Molecular Biology* **235**, 1501–1531.
5. Krogh, A., Mian, I. S., and Haussler, D. (1994) *Nucleic Acids Research* **22**, 4768–4778.
6. Fox, G. E. and Woese, C. R. (1975) *Nature* **256**, 505–507.
7. Woese, C. R., Gutell, R. R., Gupta, R., and Noller, H. F. (1983) *Microbiology Reviews* **47**(4), 621–669.
8. James, B. D., Olsen, G. J., and Pace, N. R. (1989) *Methods in Enzymology* **180**, 227–239.
9. Tinoco Jr., I., Uhlenbeck, O. C., and Levine, M. D. (1971) *Nature* **230**, 363–367.
10. Turner, D. H., Sugimoto, N., and Freier, S. M. (1988) *Annual Review of Biophysics and Biophysical Chemistry* **17**, 167–192.
11. Gouy, M. (1987) In M. J. Bishop and C. R. Rawlings, (ed.), Nucleic acid and protein sequence analysis, a practical approach, pp. 259–284 IRL Press Oxford, England.
12. Zuker, M. (1989) *Science* **244**, 48–52.
13. Waterman, M. S. (1989) In M. S. Waterman, (ed.), Mathematical Methods for DNA Sequences, chapter 8 CRC Press.
14. Le, S. Y. and Zuker, M. (1991) *J. Biomolecular Structure and Dynamics* **8**, 1027–1044.
15. Han, K. and Kim, H.-J. (1993) *NAR* **21**, 1251–1257.
16. Chiu, D. K. and Kolodziejczak, T. (1991) *CABIOS* **7**, 347–352.
17. Sankoff, D. (1985) *SIAM J. Appl. Math.* **45**, 810–825.
18. Winker, S., Overbeek, R., Woese, C., Olsen, G., and Pfluger, N. (1990) *CABIOS* **6**, 365–371.
19. Klinger, T. and Brutlag, D. (1993) In Lawrence Hunter, David Searls, and Jude Shavlik, (ed.), First International Conference on Intelligent Systems for Molecular Biology, Menlo Park: AAAI Press.
20. Gutell, R. R., Power, A., Hertz, G. Z., Putz, E. J., and Stormo, G. D. (1992) *NAR* **20**, 5785–5795.
21. Staden, R. (1990) In Doolittle (2) pp. 193–211.
22. Lathrop, R. H., Webster, T. A., and Smith, T. F. (1987) *Communications of the ACM* **30**(11), 909–921.
23. Sibbald, P. R. and Argos, P. (1990) *CABIOS* **6**(3), 279–288.
24. Abarbanel, R. M., Wieneke, P. R., Mansfield, E., Jaffe, D. A., and Brutlag, D. L. (1984) *NAR* **12**(1), 263–280.
25. Saurin, W. and Marliere, P. (1987) *CABIOS* **3**(2), 115–120.
26. Gautheret, D., Major, F., and Cedergren, R. (1990) *CABIOS* **6**(4), 325–331.
27. Cohen, F. E., Abarbanel, R. M., Kuntz, I. D., and Fletterick, R. J. (1986) *Biochemistry* **25**, 266–276.

28. Presnell, S. R. and Cohen, F. E. (1993) *Annual Review of Biophysics and Biomolecular Structure* **22**, 283–298.
29. Eddy, S. R. and Durbin, R. (1994) *Nucleic Acids Research* **22**, 2079–2088.
30. Macke, T., Cohen, P., Grate, L., Mian, I., and Noller, H. F. (in preparation) (1994).
31. Fichant, G. A. and Burks, C. (1991) *JMB* **220**, 659–671.
32. Staden, R. (1980) *NAR* **8**, 817–825.
33. Marvel, C. C. (1986) *NAR* **14**, 431–435.
34. Searls, D. B. (1992) *American Scientist* **80**, 579–591.
35. Baker, J. K. (1979) Speech Communication Papers for the 97th Meeting of the Acoustical Society of America pp. 547–550.
36. Searls, D. B. (1993) In L. Hunter, (ed.), Artificial Intelligence and Molecular Biology, chapter 2, pp. 47–120 AAAI Press.
37. Searls, D. B. and Dong, S. (1993) In H. A. Lim, J. Fickett, C. R. Cantor, and R. J. Robbins, (ed.), Proceedings of the 2nd International Conference on Bioinformatics, Supercomputing, and Complex Genome Analysis, World Scientific : pp. 89–101.
38. Sakakibara, Y., Brown, M., Hughey, R., Mian, I. S., Sjölander, K., Underwood, R. C., and Haussler, D. (1994) In Proceedings of the Asilomar Conference on Combinatorial Pattern Matching New York, NY: Springer-Verlag. In press.
39. Sakakibara, Y., Brown, M., Hughey, R., Mian, I. S., Sjölander, K., Underwood, R., and Haussler, D. The application of stochastic context-free grammars to folding, aligning and modeling homologous RNA sequences Technical Report UCSC-CRL-94-14 University of California, Santa Cruz Computer and Information Sciences Dept., Santa Cruz, CA 95064 (1993).
40. Steinberg, S., Misch, A., and Sprinzl, M. (1993) *NAR* **21**, 3011–3015.
41. Lari, K. and Young, S. J. (1990) *Computer Speech and Language* **4**, 35–56.
42. Hopcroft, J. and Ullman, J. (1979) Introduction to Automata Theory, Languages and Computation, Addison-Wesley, .
43. Rabiner, L. R. (1989) *Proc IEEE* **77**(2), 257–286.
44. Zuker, M. and Sankoff, D. (1984) *Bull. Math. Biol.* **46**, 591–621.
45. Waterman, M. S. (1988) *Methods in Enzymology* **164**, 765–792.
46. Weiser, B., Gutell, R., and Noller, H. F. In preparation (1994).
47. Michel, F. and Westhof, E. (1990) *JMB* **216**, 585–610.
48. Michel, F., Ellington, A. D., Couture, S., and Szostak, J. W. (1990) *Nature* **347**, 578–580.
49. Michel, F., Umesono, K., and Ozeki, H. (1989) *Gene* **82**, 5–30.
50. Brown, J. W., Haas, E. S., James, B. D., Hunt, D. A., Liu, J. S., and Pace, N. R. (1991) *J. Bact.* **173**, 3855–3863.
51. Tranguch, A. J. and Engelke, D. R. (1993) *JBC* **268**, 14045–1455.
52. Guthrie, C. and Patterson, B. (1988) *Ann. Rev. Gen.* **22**, 387–419.
53. Zwieb, C. (1989) *Prog. Nuc. Acids. Res. Mol. Biol.* **37**, 207–234.
54. Lathrop, R. H. and Smith, T. F. (1994) In Proceedings of the 27th Hawaii International Conference on System Sciences Los Alamitos, CA: IEEE Computer Society Press.
55. Grate, L., Herbster, M., Hughey, R., Mian, I., Noller, H., and Haussler, D. (1994) In Proc. of Second Int. Conf. on Intelligent Systems for Molecular Biology Menlo Park, CA: AAAI/MIT Press.
56. Sakakibara, Y., Brown, M., Mian, I. S., Underwood, R., and Haussler, D. (1994) In Proceedings of the Hawaii International Conference on System Sciences Los Alamitos, CA: IEEE Computer Society Press.
57. Fu, K. S. (1982) Syntactic pattern recognition and applications, Prentice-Hall, Englewood Cliffs, NJ.
58. Engelfriet, J. and Rozenberg, G. (1991) In E. Ehrig, H.J. Kreowski, and G. Rozenberg, (ed.), Lecture Notes in Computer Science, 532. pp. 12–23 Springer-Verlag.
59. Steinberg, S., Gautheret, D., Cedergren, R., and Sprinzl, M. (1993) In D. Söll and U. L. RajBhandary, (ed.), Transfer RNA, American Society of Microbiology Washington DC.