# Prediction of Beta-Sheet Structures Using Stochastic Tree Grammars

Hiroshi Mamitsuka　　Naoki Abe[1]
mami@sbl.cl.nec.co.jp　　abe@sbl.cl.nec.co.jp

Theory NEC Laboratory, RWCP[2]
c/o NEC C & C Research Laboratories, 4-1-1 Miyazaki Miyamae-ku,
Kawasaki,216 Japan.

### Abstract

*We empirically demonstrate the effectiveness of a method of predicting protein secondary structures, $\beta$-sheet regions in particular, using a class of stochastic tree grammars as representational language for their amino acid sequence patterns. The family of stochastic tree grammars we use, the Stochastic Ranked Node Rewriting Grammars (SRNRG), is one of the rare families of stochastic grammars that are expressive enough to capture the kind of long-distance dependencies exhibited by the sequences of $\beta$-sheet regions, and at the same time enjoy relatively efficient processing. We applied our method on real data obtained from the HSSP database and the results obtained are encouraging: Using an SRNRG trained by data of a particular protein, our method was actually able to predict the location and structure of $\beta$-sheet regions in a number of different proteins, whose sequences are less than 25 per cent homologous to the training sequences. The learning algorithm we use is an extension of the 'Inside-Outside' algorithm for stochastic context free grammars, but with a number of significant modifications. First, we restricted the grammars used to be members of the 'linear' subclass of SRNRG, and devised simpler and faster algorithms for this subclass. Secondly, we reduced the alphabet size (i.e. the number of amino acids) by clustering them using their physico-chemical properties, gradually through the iterations of the learning algorithm. Our experiments indicate that our prediction method not only goes beyond what is possible by alignment alone, but the grammar that was acquired by our learning algorithm captures the type of long distance dependencies that could not be succinctly expressed by an HMM. We also stress that our method can predict the* structure *as well as the location of $\beta$-sheet regions, which was not possible by previous inverse protein folding methods.*

[1] 馬見塚 拓、安倍 直樹：RWCP、理論 NEC 研究室、c/o NEC C&C 研究所，〒 216 川崎市宮前区宮崎 4-1-1。
[2] Real World Computing Partnership.

# 1 Introduction

The problem of predicting protein structures from their amino acid sequences is probably the single most important problem in genetic information processing with immense scientific significance and broad engineering applications. The *secondary* structure prediction problem, namely the problem of determining which regions in a given amino acid sequence correspond to each of the three categories, $\alpha$-helix, $\beta$-sheet, and others, is considered to be an important step towards this goal, and has been attempted by many researchers (e.g. [14]). No method to date, however, has achieved a prediction accuracy much higher than 70 per cent, casting serious doubt as to whether a significantly better performance is achievable by any approach along this line.

Motivated largely by the apparent limitation of residue-wise secondary structure prediction methods, more 'knowledge intensive' approaches to the problem of protein structure prediction have been proposed and investigated, including the homology-based approach [4] and the 'inverse protein folding' approach [5]. More recently it has been proposed [6] that all protein structures (foldings) found in today's living organisms can be classified into a relatively small number (less than a thousand or so) of types, in confirmation of such knowledge intensive approaches. In a knowledge-based approach, the prediction method keeps effectively a catalogue of patterns of amino acid sequences corresponding to existing types of protein structures, and prediction on a new sequence is done by simply finding those patterns that match parts of the input sequence. The central issue here then is how to represent these patterns with a sufficient and appropriate level of generalization. Abe and Mamitsuka have recently proposed to use a certain class of stochastic tree grammars called the Stochastic Ranked Node Rewriting Grammars (SRNRG) as representational scheme for sequence patterns of protein secondary structures, especially those of $\beta$-sheets [2]. The primary goal of the present paper is to demonstrate its effectiveness by further experimental results.

The problem of predicting $\beta$-sheet regions has been considered difficult because $\beta$-sheets typically range over several discontinuous sections in an amino acid sequence, and their sequences exhibit long distance dependency. The family of stochastic tree grammars we use in the present paper (SRNRG) is suitable for expressing the kind of long-distance dependencies exhibited by the sequences of $\beta$-sheet regions, such as the 'parallel' and 'anti-parallel' dependencies and their combinations. RNRG was originally introduced in the context of computationally efficient learnability of grammars in [1], and its discovery was inspired by the pioneering work of Joshi et al. [11, 18] on a formalism for natural language called 'Tree Adjoining Grammars' (TAG).[1] In particular, SRNRG has expressive power exceeding those of both Hidden Markov Models (HMMs) and stochastic context free grammars (SCFGs), and yet allows the existence of polynomial time parsing and local optimization algorithm for the maximum likelihood settings of probability parameters.[2]

We designed and implemented a method for predicting $\beta$-sheet regions using SRNRG as the representational language. Our prediction method receives as input amino acid sequences with the location of $\beta$-sheet regions marked, and trains the probability parameters of an SRNRG,

[1] Searls claimed that the language of $\beta$-sheets is beyond context free and suggested that they are indexed languages [17]. Indexed languages are not recognizable in polynomial time, however, and hence indexed grammars are not useful for our purpose. RNRG falls *between* them and appears to be just what we need.

[2] Both HMM and SCFG have recently been used in the context of genetic information processing [12, 3, 15, 8].

so that its distribution best approximates the patterns of the input sample. Some of the rules in the grammar are intended *a priori* for generating $\beta$-sheet regions and others for non-$\beta$-sheets. After training, the method is given a sequence of amino acids with *unknown* secondary structure, and predicts according to which regions are generated by the $\beta$-sheet rules, in the *most likely* parse for the input sequence.

The learning algorithm we use is an extension of the 'Inside-Outside' algorithm for the stochastic context free grammars. In order to reduce the rather high computational requirement of the learning and parsing algorithms, we have restricted the form of grammars to a certain subclass of RNRG which we call the 'linear RNRG,' and devised a simpler and faster learning algorithm for the subclass. We also employed a method of reducing the alphabet size[3] (i.e. the number of amino acids) by clustering them using MDL(Minimum Description Length) approximation and their physico-chemical properties, gradually through the iterations of the learning algorithm.[4]

We applied our method on real data obtained from the HSSP (Homology-derived Secondary Structures of Proteins Ver 1.0 [16]) database. The results obtained indicate that our method is able to capture and generalize the type of long-distance dependencies that characterize $\beta$-sheets. Using an SRNRG trained by data for a particular protein, our method was actually able to predict the location and structure of $\beta$-sheets in test sequences of a number of different proteins, which have similar structures but have less than 25 per cent pairwise homology to the training sequences.[5] We emphasize that, unlike previous secondary structure prediction methods, our method is able to predict the *structure* of the $\beta$-sheet, namely the locations of the hydrogen bonds. Furthermore, our experiments indicate that the grammar that was acquired by our learning algorithm captures the type of long distance dependencies that could not be succinctly expressed by an HMM.

# 2 Modeling Beta Sheet Structures with RNRG

We first briefly review the definition of the Ranked Node Rewriting Grammar (RNRG) and give some illustrative examples.[6] An RNRG is a tree generating system, and consists of a single tree structure called the *starting* tree, and a finite collection of rewriting rules which rewrite a node in a tree with an incomplete tree structure. The node to be rewritten needs to be labeled with a *non-terminal* symbol, and must have the same number of descendants (called the 'rank' of the node) as the number of 'empty nodes' in the incomplete tree structure. After rewriting, the descendants of the node are attached to these empty nodes in the same order as before rewriting. The string language of the grammar is the set of *yields* of the trees generated by the grammar, namely the strings that appear on the leaves of the trees. If we place an upper bound, say $k$, on the rank of a node that can be rewritten, we obtain families of grammars, RNRG($k$), each of which has varying expressive power. The string languages of RNRG(0), denoted RNRL(0),

[3] As is well known, there are twenty amino acids, and hence we are dealing with an alphabet of size 20.

[4] The physico-chemical properties we use are the molecular weight and the hydrophobicity, which were used in [13] in their method for predicting $\alpha$-helix regions.

[5] Prediction problems for which the training sequences and the test sequences are less than 25 per cent homologous are sometimes referred to in the literature as the 'Twilight Zone' [7], since alignment is not effective for such problems.

[6] We refer the interested reader to [1] for the detailed definition.

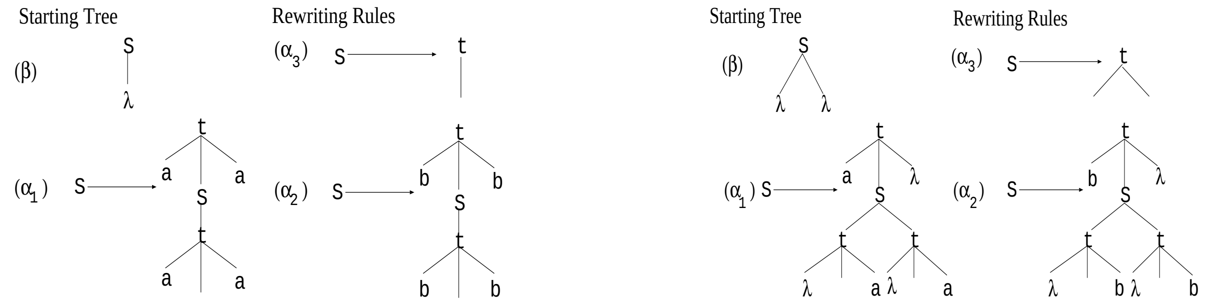

Figure 1: (a) RNRG(1) grammar $G_1$ and (b) RNRG(2) grammar $G_2$.

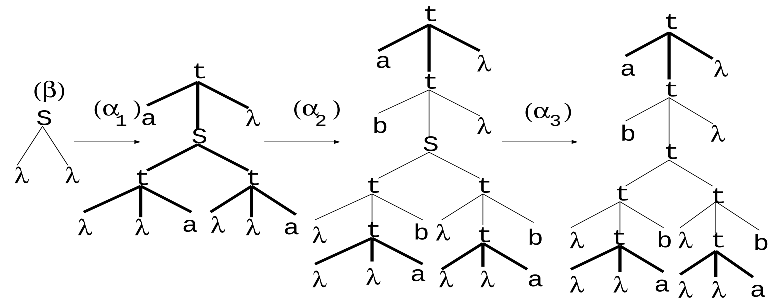

Figure 2: Derivation of 'ababab' by an RNRG-2 grammar

equal the context free languages (CFL), those of RNRG(1) equal the tree adjoining languages (TAL), and for any $k \geq 2$, RNRL($k$) properly contains RNRL($k-1$). We now give some examples of RNRG grammars. The language $L_1 = \{ww^Rww^R | w \in \{a,b\}\}$ is generated by the RNRG(1) grammar $G_1$ shown[7] in Figure 1(a). The '3 copy' language $L_2 = \{www \mid w \in \{a,b\}^*\}$ can be generated by the RNRG(2) grammar $G_2$ shown in Figure 1(b). Note that $L_1$ can be generated by a tree adjoining grammar, but not $L_2$. The way the derivation in RNRG takes place is illustrated in Figure 2, which shows the derivation of the string 'ababab' by $G_2$. Each of the trees shown in Figure 2 is called a 'partially derived tree.' Note that the tree structure introduced by a particular rule may be split into several pieces in the final derived tree, unlike usual parse trees in CFG. (In the figure, the part of the derived tree introduced by $(\alpha_1)$ is indicated in a thick line.) Given the definition of RNRG, the *stochastic* RNRG is defined analogously to the way stochastic CFG is defined from CFG. That is, associated with each rewriting rule in a stochastic RNRG is its *rule application probability*, which is constrained so that for each non-terminal, the sum total of rule application probabilities of all the rewriting rules for that non-terminal equals unity.

Next some typical $\beta$-sheet structures are illustrated schematically in Figure 3. The arrows indicate the $\beta$-sheet strands, and the line going through them the amino acid sequence. The $\beta$-sheet structure is retained by hydrogen bonds (H-bonds) between the corresponding amino acids in neighboring strands, so it is reasonable to suspect that there are correlations between the amino acids in those positions. The structure exhibited in Figure 3 (a) is known as the 'anti-parallel' $\beta$-sheet, as the dependency follows the pattern *abc..cba..abc...cba*, where the use of a same letter indicates that those positions are connected by H-bonds and believed to be

[7] Note that '$\lambda$' indicates the empty string, and an edge leading to no letter leads to an empty node.

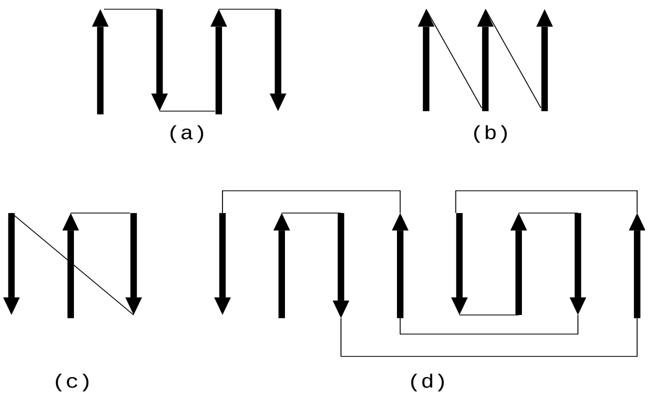

Figure 3: Some typical $\beta$-sheet structures

correlated. In contrast, the structure exhibited in Figure 3 (b) is known as the 'parallel' $\beta$-sheet, since the dependency here is of the pattern *abc..abc*... Both of these types of dependency can be captured by RNRG, in particular, $G_1$ and $G_2$ in Figure 1, respectively. These structures can be combined to obtain larger $\beta$-sheets, as is shown in Figure 3(d) and can result in a high degree of complexity, but they can be handled by an RNRG of a higher rank.

# 3 Learning and Parsing of The Linear Subclass

The 'linear' subclass of RNRG we use in this paper is the subclass satisfying the following two constraints: (i) Each rewriting rule contains at most one node labeled with a non-terminal symbol of rank greater than 0; (ii) Every other non-terminal (of rank 0) is a 'lexical' non-terminal, namely all rewriting rules for it are of the form $A \rightarrow a$ for some terminal symbol $a$. Examples of RNRG of rank 1 satisfying these constraints can be found, for example, in Figure 4(a). Note that each occurrence of a lexical non-terminal can be thought of as defining a distribution over the alphabet, and this is written in as part of the rule in the figure. With these constraints, the parsing and learning algorithms can be significantly simplified.

## 3.1 The Learning Algorithm

Our learning algorithm is an extension of the 'Inside-Outside' algorithm for SCFG [10] and it is a local optimization algorithm for the maximum likelihood settings of the rule application probabilities and the letter (amino acid) generation probabilities in the input grammar. The algorithm is an iterative procedure which re-estimates and updates its current settings of all of its probability parameters. The re-estimation procedure is guaranteed to increase the likelihood assigned to the sample, as is the case with the Baum-Welch re-estimation for HMM. Due to space limitations, we omit the details of our learning algorithm, which can be found in [2].

## 3.2 Reducing the Alphabet Size with MDL Approximation

After each iteration of the above learning algorithm at each lexical rule, we attempt to merge some of the amino acids, if the merge reduces the total description length (approximated using the probability parameters calculated up to that point). For this purpose we make use of the Euclidean distance between the 20 amino acids in the (normalized) 2-dimensional space defined by their molecular weight and hydrophobicity. At each iteration, we select the two among the clusters from the previous iteration, which are *closest* to each other in the above Euclidean space, and merge them to obtain a single new cluster, *provided* that the merge results in reducing the following approximation of 'description length,' where we let $c \in C$ be the clusters, $P(c)$ the sum total of generation probabilities of amino acids in the cluster $c$, and $m$ the *effective* sample size, namely the weighted frequency of the lexical rule in question in the parses of the input sample, weighted according to the current parameter settings.

# 4 Experimental Results

We applied our method on real data obtained from the HSSP database. In our first experiment, we picked three different proteins, 'Fasciculin' (1fas), 'Caldiotoxin' (1cdta) and 'Neurotoxin B' (1nxb), all of which are toxins. Although these three proteins do have relatively similar structures, their sequences are less than 25 per cent homologous to one another,[8] and hence alignment alone can hardly detect this similarity. We trained a stochastic RNRG with training data consisting effectively of *bracketed* sequences for one of the three proteins, say 1fas, and used the acquired grammar to predict the location of $\beta$-sheet regions in an amino acid sequence of another one of the three, either 1cdta or 1nxb. By *bracketing* the input sequences, we mean that we isolated out the (discontinuous) sub-strings of the training sequences that correspond to $\beta$-sheets from the rest, and trained the probability parameters of the '$\beta$-sheet rules' in the grammar with them.[9] The probability parameters of the non-$\beta$-rules were set to be uniform. We then used the acquired stochastic RNRG grammar to parse an amino acid sequence of either 1cdta or 1nxb, and predicted the location of $\beta$-sheet regions according to where the $\beta$-sheet rules are in the most likely parse. It was able to predict the location of all three $\beta$-strands contained in the test sequence almost exactly (missing only one or two residues which were absent in all of the training data) in both cases. We repeated the same experiment for all (six) possible combinations of the training data and a test sequence from the three proteins. Our method was able to predict all three of the $\beta$-strands in all cases, except in predicting the location of $\beta$-sheet in a test sequence for 1cdta from training data for 1nxb: It failed to identify one of the three $\beta$-strands correctly in this case. The sequences of these toxins were approximately 60 residue long, and the parsing of these sequences required more than an hour on a Silicon Graphics Indigo II graphic workstation.

Figure 4(a) shows the part of the stochastic RNRG(1) grammar obtained by our learning algorithm on the training set for 1fas that generates the $\beta$-sheet regions. Note that, in the figure, the amino acid generation probabilities at each position are written in a box. For example, the distribution at the right upper corner in ($\alpha_4$) gives probability 0.80 to the cluster $\{I, L, V\}$ and probability 0.10 to the single amino acid $Y$. The interpretation of the grammar is summarized schematically in Figure 4(b). It is easy to see that the grammar represents a class of $\beta$-sheets of type (c) in Figure 3. Each of the rules ($\alpha_1$), ($\alpha_2$), ($\alpha_3$), ($\alpha_4$), ($\alpha_6$) and ($\alpha_7$) generates part of the $\beta$-sheet region corresponding to a row of H-bonds, and ($\alpha_5$) inserts an 'extra' amino acid

[8] These were obtained using PDB_SELECT (25 %) developed by Hoboem et. al. [9].

[9] Bracketed input samples are often used in applications of SCFG in speech recognition.

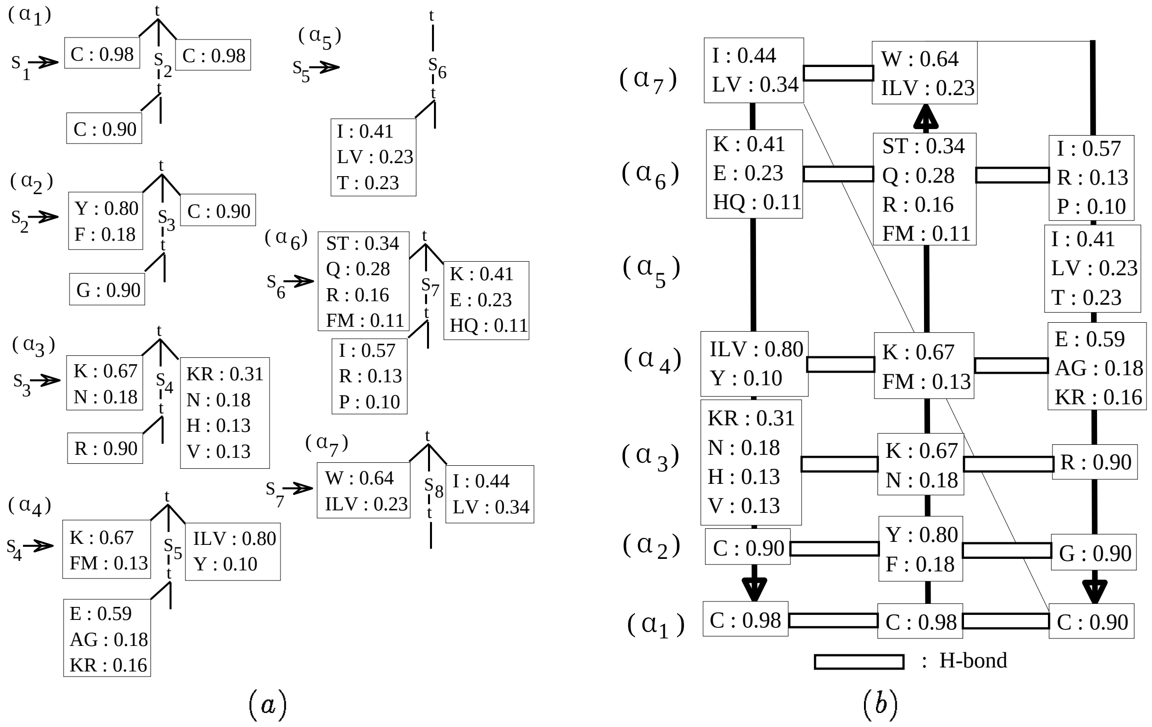

Figure 4: (a) A part of the acquired RNRG grammar and (b) its interpretation.

that does not take part in any H-bond. Rule $(\alpha_4)$ says that in the third (from the top) row of H-bonds, amino acids $I, L$ and $V$ are equally likely to occur in the leftmost strand, and it is very likely to be $K$ in the middle strand. Note that $I, L$, and $V$ have similar physico-chemical properties, and it is reasonable that they were merged to form a cluster.

Figure 5(a) shows the most likely parse (derived tree) obtained by the grammar on a test sequence of 1cdta. The shaded areas indicate the actual $\beta$-sheet regions, which are *all* correctly predicted. The seven types of thick lines correspond to the parts of the derived tree generated by the seven rules shown in Figure 4(a), respectively. The structural interpretation of this parse is indicated schematically in Figure 5(b), which is also exactly correct. Note that the distributions of amino acids are quite well spread over a large number of amino acids. For example, none of the amino acids in the third strand of the test sequence, except the last two $C$s, receives a dominantly high probability in the acquired grammar. The merging of $I, L$ and $V$ mentioned above, therefore, was crucial for the grammar to be able to predict the third strand of the $\beta$-sheet in the test sequence.

One apparent shortcoming of the experimental result we just described is that only one copy of each of the rules $(\alpha_1)$, ..., $(\alpha_7)$ was present in the trained grammar. As a result, each of the acquired rules was able to simply capture the distributions of amino acids at each residue position, and therefore was not able to truly capture the correlations that exist between residue positions, even if they are captured by a single rule. In another experiment we conducted using exactly the same data as in the above experiment, we used multiple copies (two in particular) of each of the $\beta$-sheet rules $(\alpha_1)$, ..., $(\alpha_7)$. (Randomly generated numbers were used for the initial values of their probability parameters.) In the acquired grammar, some rules were split into a pair of rules that significantly differ from each other, while others became basically two

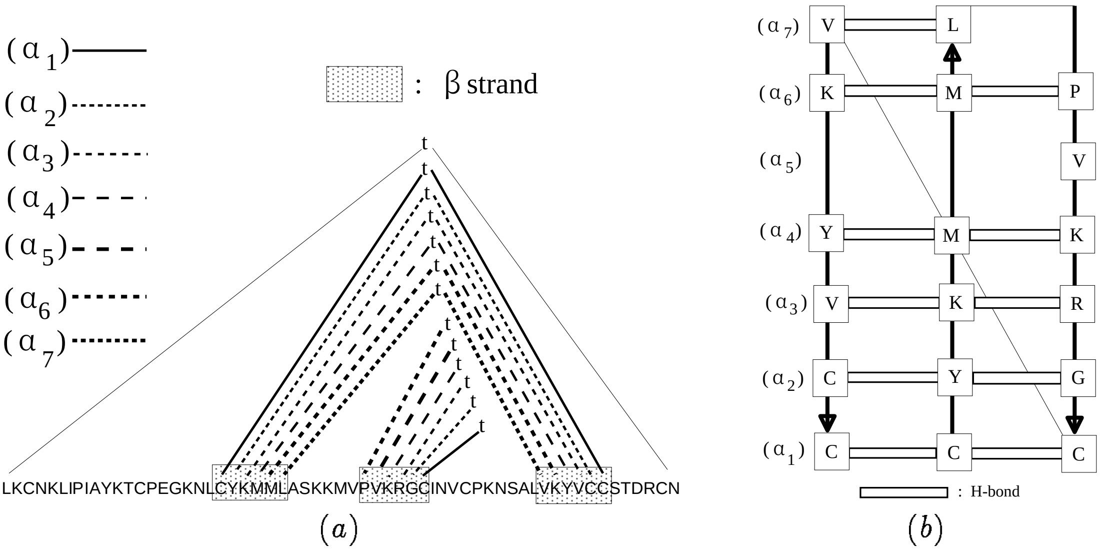

Figure 5: (a) The parse of the test sequence and (b) its interpretation.

copies of the same rule. An example of a rule that was split is ($\alpha_3$) in Figure 4(a), and the two rules it split into are shown in Figure 6(a). This split is meaningful, because in the new grammar, the joint distribution over the two nodes at the top are seen to be heavily concentrated on $(K, \{N, H\})$ and $(N, \{R, K\})$, which is finer than what we had in the previous grammar $(\{K, N\}, \{N, H, R, K\})$. This way, the grammar was able to capture the correlation between these residue positions, which are far from each other in the input sequence.

The grammar containing two copies each of the $\beta$-sheet rules obtained using training data for 1fas was used to predict a test sequence for both 1cdta and 1nxb. As before, the locations of all three $\beta$-strands were predicted exactly correctly. Interestingly, distinct copies of some of the split rules were used in the respective most likely parses for 1cdta and 1nxb. For example, rule ($\alpha_3$-1) was used in the most likely parse for the test sequence for 1cdta, ($\alpha_3$-2) for 1nxb. It seems to indicate that the training sequences for 1fas contained at least two dependency patterns for this bonding cite, as shown in Figure 6(b), and the corresponding bonding cite in 1cdta was of the first type and 1nxb of the second.

The point just illustrated is worth emphasizing. If one tried to capture this type of correlations that exist in bonding cites by a hidden Markov model (HMM), it would necessarily result in a much higher complexity. For example, suppose that eight bonding cites in a row (say each with just two residue positions for simplicity) are split into two distinct rules. Note that in an HMM, the eight rules would have to be realized by two copies of consecutive states - sixteen states in a chain. Since there are $2^8 = 256$ possible combinations of rules to use, the HMM would have to have 256 non-deterministic branches of state sequences, each corresponding to a possible combination of the eight options. In the case of stochastic tree grammar, we only needed to have $2 \times 8$ rules. Clearly this huge saving in complexity is made possible by the richer expressive power of stochastic tree grammars.

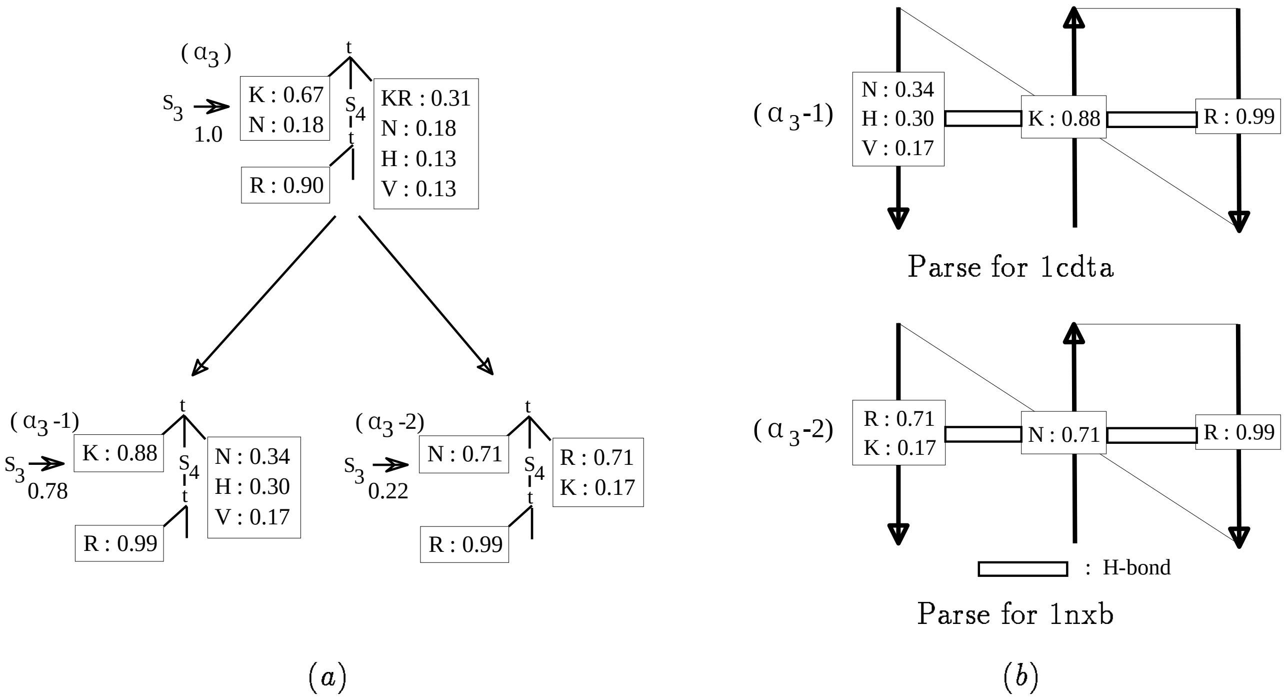

Figure 6: (a) Rules ($\alpha_3$) split into and (b) their interpretations.

# 5 Concluding Remarks

We have described a method for predicting protein secondary structures, using a class of stochastic tree grammars as representational language for their amino acid sequence patterns. Our experimental results, admittedly, were of preliminary nature in which the test sequences were known to contain relatively similar structures to those of the training sequences. In the future, we hope to demonstrate that our method can achieve a high prediction accuracy on an *arbitrary* test sequence, when equipped with a large catalogue of generalized patterns, expressed as SRNRG rules.

# References

[1] N. Abe. Feasible learnability of formal grammars and the theory of natural language acquisition. In *Proceedings of COLING-88*, August 1988.

[2] N. Abe and H. Mamitsuka. A new method for predicting protein secondary structures based on stochastic tree grammars. In *Proceedings of the Eleventh International Conference on Machine Learning*, 1994.

[3] K. Asai, S. Hayamizu, and K. Onizuka. HMM with protein structure grammar. *Proceedings of the Hawaii International Conference on System Sciences*, pages 783–791, 1993.

[4] T. L. Blundell, B. L. Sibanda, M. J. E. Sternberg, and J. M. Thornton. Knowledge-based prediction of protein structures and the design of novel molecules. *Nature*, 326:347–352, March 1987.

[5] J. U. Bowie, R. Luthy, and D. Eisenberg. A method to identify protein sequences that fold into a known three-dimensional structure. *Science*, 253:164–170, July 1991.

[6] C. Chothia. One thousand families for the molecular biologist. *Nature*, 357:543–544, June 1992.

[7] R. F. Doolittle, D. F. Feng, M. S. Johnson, and M. A. McClure. Relationships of human protein sequences to those of other organisms. *Cold Spring Harbor Symp. Quant. Biol.*, 51:447–455, 1986.

[8] S. Eddy and R. Durbin. Rna sequence analysis using covariance models. *Nucleic Acids Research*, 22:2079–2088, 1994.

[9] U. Hoboem, M. Scharf, R. Schneider, and C. Sander. Selection of a representative set of structures from the Brookhaven protein data bank. *Protein Science*, 1:409–417, 1992.

[10] F. Jelinik, Lafferty, and R. Mercer. Basic methods of probabilistic context free grammars. *IBM Research Reports*, RC16374(#72684), 1990.

[11] Aravind K. Joshi, Leon Levy, and Masako Takahashi. Tree adjunct grammars. *Journal of Computer and System Sciences*, 10:136–163, 1975.

[12] A. Krogh, M. Brown, S. Mian, K. Sjolander, and D. Haussler. Hidden markov models in computational biology: Applications to protein modeling. *J. Mol. Biol.*, 235:1501–1531, 1994.

[13] H. Mamitsuka and K. Yamanishi. Protein secondary structure prediction based on stochastic-rule learning. In *Proceedings of the Third Workshop on Algorithmic Learning Theory*, pages 240–251, 1992.

[14] B. Rost and C. Sander. Prediction of protein secondary structure at better than 70 % accuracy. *J. Mol. Biol.*, 232:584–599, 1993.

[15] Y. Sakakibara, M. Brown, R. C. Underwood, I. S. Mian, and D. Haussler. Stochastic context-free grammars for modeling RNA. In *Proceedings of the 27th Hawaii International Conference on System Sciences*, volume V, pages 284–293, 1994.

[16] C. Sander and R. Schneider. Database of homology-derived structures and the structural meaning of sequence alignment. *Proteins: Struct. Funct. Genet.*, 9:56–68, 1991.

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

[18] K. Vijay-Shanker and A. K. Joshi. Some computational properties of tree adjoining grammars. In *23rd Meeting of A.C.L.*, 1985.