

A grammatical theory for the conformational changes of simple helix bundles

David Chiang^{*†}

Information Sciences Institute
University of Southern California
4676 Admiralty Way
Marina del Rey, CA 90292

Aravind K. Joshi

Department of Computer and Information Science
University of Pennsylvania
Philadelphia, PA 19104

Ken A. Dill

Department of Pharmaceutical Chemistry
University of California
San Francisco, CA 94143

October 12, 2005

^{*}To whom correspondence should be addressed.

[†]This research was carried out while the first author was at the University of Pennsylvania, Department of Computer and Information Science.

Abstract

Polymers, including biomolecules such as proteins, have two particularly important types of single-molecule transitions: a helix-coil transition, driven by interactions that are local in the chain, and a collapse transition, driven by nonlocal interactions. A long-standing challenge of polymer statistical mechanics has been to deal with both types of transition in a single theoretical framework. The simplest paradigmatic problem would be a theory of helix-bundle folding. Here, we show how the machinery of formal grammars, originally developed in the context of linguistic analysis and programming-language compilation, provides a simple and general way to combine the Zimm-Bragg model of α -helices with the model of Chen and Dill for nonlocal interactions in antiparallel polymeric systems. We use a well-known construction in the theory of formal grammars to give the statistical mechanical partition function for two-helix bundles. Predictions are shown to be quite good in comparison to exact enumerations within a lattice model.

Keywords: partition function, helix bundles, formal grammars, finite-state automata, HP lattice model

1 Introduction

It is of interest to better understand conformational transitions in polymers, such as in the folding of proteins and RNA molecules, conformational switching in RNA, and helix-coil and collapse processes in synthetic and biological polymers. To predict these processes from first principles requires knowledge of a statistical mechanical partition function, the Boltzmann-weighted sum over all the conformational states. Historically, this has posed an important general problem. The most practical and useful statistical mechanical theories are those for which the partition function can be factorized into component factors. Helix-coil processes in polymers can be readily treated because their partition functions factorize by virtue of the dominance of “local” interactions in the chain. For example, in theories such as those of Schellman (1958), Zimm and Bragg (1959), or Lifson and Roig (1961), each helical turn is treated as being approximately dependent only on the neighboring helical turn. On the other hand, collapse processes, where nonlocal interactions are dominant, can also be readily treated through a rather different type of factorization of the partition function, into pairwise bead interactions in “mean-field” and lattice theories.

A major challenge in the statistical physics of polymers and biomolecules is to be able to treat both the local and nonlocal interactions within a single framework. The simplest paradigmatic problem in this area is the folding of a 2-helix bundle: it involves both the local helical interactions, and also the nonlocal interactions that bring the two helices together into a bundle. Here, we show that methods of computational linguistics have the power to do this, and to cross over into the arena of statistical physics, to provide good estimates of partition functions of such foldable and complex biomolecules.

We show how the machinery of formal grammars, originally developed in the context of linguistic analysis and programming-language compilation, provides a elegant reformulation of three models: the Zimm-Bragg model of α -helices and Chen and Dill’s models of double-stranded chain molecules (Chen and Dill, 1995, 1998). The algorithms used in these models are shown to be variants of the Forward algorithm for Markov models, and the Inside (CKY) algorithm for context-free grammars (Kasami, 1965; Younger, 1967). Our reformulation reveals some corrections and an improvement,

but the central result of the paper is the use of a well-known construction in the theory of formal grammars to combine the Zimm-Bragg model with Chen and Dill's second model, yielding a new model of two-helix bundles.

2 Background

The partition function of a molecule gives the relative accessibility of the possible conformations of the molecule. It is defined as

$$Q = \sum_j e^{-E_j/kT} \quad (1)$$

where T is the temperature, k is Boltzmann's constant, j ranges over the conformations of the molecule, and E_j is the energy of conformation j . When the terms of the series are divided by the sum Q , they form a probability distribution over conformations, the Boltzmann distribution. From a computational standpoint, the problem is not simply to find the sum Q , but various functions defined over the terms of the series: for example, to find the maximum term (corresponding to the most likely conformation), or to compute the expected value of some random variable (e.g. the number of contacts).

A formal grammar is a symbolic system that specifies a set of allowable structures of sequences. It was Searls (1988) who first applied formal grammars to model nucleic acid stem-and-loop secondary structure. Consider the following toy context-free grammar (CFG) for RNA sequences:

$$\begin{aligned} S &\rightarrow X \\ X &\rightarrow \text{aXu} \mid \text{uXa} \mid \text{cXg} \mid \text{gXc} \\ X &\rightarrow \text{XX} \mid \text{XXX} \\ X &\rightarrow \epsilon \end{aligned} \quad (2)$$

where the symbol ϵ stands for the empty string. The operation of the grammar can be thought of in several ways. First, it can be thought of as a set of rules for building a string by rewriting. We start with a single S , and at each step we choose an uppercase symbol, called a nonterminal symbol, and a grammar rule whose left-hand side matches the symbol, and we replace the symbol with the

right-hand side of the rule:

$$\begin{aligned}
 S &\Rightarrow X \\
 &\Rightarrow aXu \\
 &\Rightarrow acXgu \\
 &\Rightarrow acXXXgu \\
 &\Rightarrow acgXcXXgu
 \end{aligned}
 \tag{3}$$

Notice how the links in the rules are preserved in the growing string. These indicate base pairings in the RNA secondary structure.

Another way to think about the grammar is as a specification of tree structures: for example, the rule $X \rightarrow XX$ indicates that it is allowable for an X node in a tree to be the parent of two other X nodes. Figure 1 shows an example tree that is allowed by our example grammar. The advantage of the tree representation is that the shape of the tree mirrors the shape of the secondary structure, so that the two can be overlaid, as shown in the figure.

Figure 1

So far we have spoken of how to use a grammar to generate possible sequences; in practice we more commonly are given a sequence and wish to find possible structures for just that sequence. This can be done in polynomial time, and there is a large family of parsing algorithms for this purpose, one of which we describe later.

It is also common to attach probabilities (more generally, weights) to the rules of a grammar. This induces a probability (weight) distribution over all possible sequences and structures allowed by the grammar: the probability (weight) of a structure is the product of all the rule probabilities (weights) used to build the structure. In this paper, we will use this device to calculate the partition function, which is our primary object.

3 The Zimm-Bragg model

The Zimm-Bragg model (Zimm and Bragg, 1959) of the helix-coil transition represents a conformation simply as a sequence of labels, 1 for a monomer hydrogen-bonded to a previous monomer, or 0 for an unbonded monomer.

In the simplest version of the model, each 1 preceded by another 1 gets the weight s , whereas each 1 preceded by a 0 gets an additional nucleation parameter σ , for a total weight of σs . Each 0 gets the weight 1. All these weights are multiplied together to give a weight for each conformation.

For an n -mer, the simplest version of the model¹ defines n vectors with coordinates corresponding to the labels 0 and 1:

$$\mathbf{M} = \begin{pmatrix} 1 & 1 \\ \sigma s & s \end{pmatrix} \quad (4)$$

$$\mathbf{a}_1 = \begin{pmatrix} 1 \\ 0 \end{pmatrix} \quad (5)$$

$$\mathbf{a}_i = \mathbf{M}\mathbf{a}_{i-1} \quad i > 1 \quad (6)$$

Then if $\mathbf{1} = \begin{pmatrix} 1 \\ 1 \end{pmatrix}$,

$$Q = \mathbf{1}^\top \mathbf{a}_{i-1} \quad (7)$$

Other quantities besides the total partition function are important to compute as well. For example, one might want to calculate the expected number of hydrogen bonds. Zimm and Bragg give closed-form approximations for this quantity, but it is not difficult to compute exact values by dynamic programming, by defining for each i a sequence of vectors $\mathbf{a}_{i,m}$, where m is the number of hydrogen

¹Zimm and Bragg require that the first three monomers all be unbonded. Here we drop this requirement to one for simplicity; it will be reinstated later.

bonds:

$$\mathbf{a}_{1,0} = \begin{pmatrix} 1 \\ 0 \end{pmatrix} \quad (8)$$

$$\mathbf{a}_{i,m} = \begin{pmatrix} \mathbf{M}_0 \mathbf{a}_{i-1,m} \\ \mathbf{M}_1 \mathbf{a}_{i-1,m-1} \end{pmatrix} \quad (9)$$

$$\langle n_h \rangle = \sum_m m \mathbf{1}^\top \mathbf{a}_{n,m} \quad (10)$$

The Zimm-Bragg model can also be written as a weighted finite-state automaton (similar to a Markov chain), shown in Figure 2. There are two vertices in the automaton, one for helix and one for coil, corresponding to the two coordinates in the matrix formulation, and every path of length $n + 1$ through the automaton starting at vertex q_0 corresponds to a state of a polymer of length n : the weights of the edges in the path, when multiplied together, give the statistical weight of the corresponding polymer state.

Figure 2

Moreover, we may formulate the Zimm-Bragg model as a grammar. We do not make use of this formulation below, but it serves as a simple example of the correspondence between a matrix formulation and a grammatical formulation. Here, and in the grammars below, the symbols are the covalent bonds between amino acids (represented as $\bullet \rightarrow$), and the strings are polypeptide chains. The use of strings over symbols representing peptide bonds as opposed to monomers is a departure from common practice (Searls, 1992) whose rationale will be made more clear in the next section, when we begin to model secondary interactions between bases. In order to ensure that our strings are self-consistent, we formalize them as follows: if Σ is an alphabet of monomers, we represent sequences as strings over $\{\bullet \rightarrow^a \bullet \mid a, b \in \Sigma\}$. We assume everywhere that if two symbols $\bullet \rightarrow^a \bullet$ and $\bullet \rightarrow^c \bullet$ are adjacent, then $b = c$; thus a string of n symbols represents an $(n - 1)$ -mer.

The Zimm-Bragg model could be formulated as a grammar thus:

$$\begin{aligned} S &\rightarrow A_j & j \in \{0, 1\} \\ A_i &\xrightarrow{M_{ij}} A_j \bullet \rightarrow & i, j \in \{0, 1\} \\ A_0 &\rightarrow \epsilon \end{aligned} \quad (11)$$

where ϵ stands for the empty string. The grammar works as follows:

1. Begin by letting $w = S$ and $p = 1$. Eventually w will be the sequence and p will be the statistical weight of a structure of w .
2. Choose one uppercase symbol (called a nonterminal symbol) in w (call it X) and a rule with a matching left-hand side, $X \xrightarrow{q} Y$.
3. Replace the one occurrence of X in w with Y .
4. Set p to pq .
5. Go back to step 2 until there are no more nonterminal symbols.

For example:

w	p	Rule
S	1	$S \rightarrow A_0$
A_0	1	$A_0 \xrightarrow{1} A_1 \bullet$
$A_1 \bullet$	1	$A_1 \xrightarrow{s} A_1 \bullet \bullet$
$A_1 \bullet \bullet$	s	$A_1 \xrightarrow{\sigma s} A_0 \bullet \bullet$
$A_0 \bullet \bullet \bullet$	σs^2	$A_0 \xrightarrow{1} A_0 \bullet \bullet$
$A_0 \bullet \bullet \bullet \bullet$	σs^2	$A_0 \rightarrow \epsilon$
$\bullet \bullet \bullet \bullet$	σs^2	

Each time we run the grammar, we get a string (w) and a weight (p); to compute the partition function of an n -mer, we add up the weights of all the runs of the grammar that generate a chain of length n .

There may be many ways to generate a single string (polymer) with a grammar; the task of computing the best way, or summing over all possible ways, given an input string, is known as parsing, and can be done efficiently by dynamic programming just as above; we defer detailed discussion of parsing to the next section.

As above, we can also modify the grammar to calculate other quantities, like the expected number

of hydrogen bonds:

$$\begin{aligned}
 S^m &\rightarrow A_j^m & j \in \{0, 1\} \\
 A_0^m &\xrightarrow{M_{0j}} A_j^m \bullet\bullet & j \in \{0, 1\} \\
 A_1^m &\xrightarrow{M_{1j}} A_j^{m-1} \bullet\bullet & j \in \{0, 1\} \\
 A_0^0 &\rightarrow \epsilon
 \end{aligned} \tag{12}$$

Then

$$\langle n_h \rangle = \sum_m m w(S^m) \tag{13}$$

where $w(S^m)$ is the sum of the weights of all derivations starting with S^m and finishing with the input string.

4 The Chen-Dill model of hairpin conformations

4.1 The HP model

Figure 3

The HP model (Lau and Dill, 1989) represents conformations as self-avoiding walks on a lattice (Figure 3)—in the following, the two-dimensional square lattice. Each unit of the chain is classified as hydrophobic (h) or polar (p). Let ε_{hh} be the energy of forming a bond between two hydrophobic units, and $q_{hh} = e^{-\varepsilon_{hh}/kT}$; then the HP model assigns to every hh contact the weight q_{hh} , and to every other contact the weight 1. The weight of a conformation is the product of the weights of each of its contacts.

Enumerating all possible conformations of an n -mer is an NP-hard problem, however; even the particular problem of finding a minimum-energy conformation is NP-hard (Crescenzi *et al.*, 1998). The firehose model of Chen and Dill (Chen and Dill, 1995, 1998) uses a matrix method similar to the Zimm-Bragg model to approximate the partition function in polynomial time.

4.2 The firehose model

Figure 4

The firehose model factors the computation of the partition function into two parts using the notion of a polymer graph. A polymer graph is a pattern of self-contacts; every conformation corresponds to a polymer graph, but more than one conformation may correspond to the same polymer graph. For example, the three groups of conformations shown in Figure 3 correspond to the three polymer graphs shown in Figure 4.

The firehose model works by first enumerating all possible polymer graphs, and then estimating the number of conformations for each polymer graph. The key to its efficiency is that it considers restricted subclasses of polymer graphs. We say that a polymer graph is nested if no two curved edges cross each other. For any curved edge e , call the region bounded by e and one or more straight edges the interior of e . We say that a nested polymer graph is linearly nested if no two curved edges have disjoint interiors. (The polymer graph in Figure 5 is nested, but not linearly nested, because the white regions are disjoint.) RNA secondary structures are conformations of nested polymer graphs; Chen

and Dill call the conformations of linearly nested polymer graphs “hairpin conformations.”

The restriction to nested polymer graphs allows Chen and Dill’s model to decompose a polymer graph into elementary components called faces and assume that each folds independently of the others. For nested polymer graphs, a face is the interior of an edge minus the interiors of the edges contained within (e.g., the shaded region in Figure 5).

Then the number of conformations of the polymer graph, can be estimated as the product of the number of conformations of each of its faces, and E_j , the energy of the polymer graph, can be estimated as the sum of the energy increments of each of its faces. Because the model largely ignores excluded volume between different faces, dynamic programming can be used to efficiently sum over all possible combinations with a particular energy.

In this section, we present Chen and Dill’s model of hairpin conformations (that is, linearly nested polymer graphs), and our reformulation of the model as a grammar.

Figure 5

4.3 Matrix formulation

We present here a simplified version of Chen and Dill’s models. Define the matrices \mathbf{f} and \mathbf{g} as follows:

$$\mathbf{f}_{ij} = \frac{1}{4}U(j - i + 1)q_{a_i a_j} \quad (14)$$

$$+ \frac{1}{4}U(j - j' + i - i' + 2)q_{a_i a_j} \mathbf{f}_{i' j'} \quad (15)$$

$$\mathbf{g}_{ij} = C(j - j' + i - i' + 1) \mathbf{f}_{i' j'} \quad (16)$$

where $U(l)$ is the number of neighbor-avoiding loops of length l on the two-dimensional lattice (l even). Then the partition function of the molecule $a_0 \cdots a_n$ is g_{0n} . Chen and Dill provide equations, analogous to (12), to calculate the contribution of each energy level to the partition function.

4.4 As a grammar

In recasting Chen and Dill’s model as a CFG, we again represent sequences not as strings over an alphabet of monomers, but as strings over an alphabet of covalent bonds. The primary reason for this is that we also want to allow self-contacts to be drawn between monomers (in derived strings as

well as in the CFG rules), and using an alphabet of covalent bonds will allow the grammar to create multiple self-contacts on a single monomer. Sometimes a lone monomer $\overset{a}{\bullet}$ will appear in a grammar rule (below, in the case $l = 0$ or $m = 0$). This is not a symbol generated by the grammar, but simply a placeholder for specifying self-contacts.

If we assign the weight $\omega e^{-\Delta E/kT}$ to each production of the grammar, where ω is the conformation count of the corresponding face, and ΔE is the energy increment of its outer link, then the weight of the derivation corresponding to polymer graph j will be $\Omega_j e^{-E_j/kT}$, and the weight of the string will be Q .

$$\begin{aligned}
& S \rightarrow G \\
& G \xrightarrow{C(l+m-1)} \overset{\bullet}{a_1} \overset{\bullet}{a_2} \dots \overset{\bullet}{a_{l-1}} \overset{\bullet}{a_l} \overset{\bullet}{F} \overset{\bullet}{b_1} \overset{\bullet}{b_2} \dots \overset{\bullet}{b_{m-1}} \overset{\bullet}{b_m} \\
& F \xrightarrow{\frac{1}{4}U(l+m)q_{a_1 b_m}} \overset{\bullet}{a_1} \overset{\bullet}{a_2} \dots \overset{\bullet}{a_{l-1}} \overset{\bullet}{a_l} \overset{\bullet}{F} \overset{\bullet}{b_1} \overset{\bullet}{b_2} \dots \overset{\bullet}{b_{m-1}} \overset{\bullet}{b_m} \\
& F \xrightarrow{\frac{1}{4}U(l)q_{a_1 a_l}} \overset{\bullet}{a_1} \overset{\bullet}{a_2} \dots \overset{\bullet}{a_{l-1}} \overset{\bullet}{a_l}
\end{aligned} \tag{17}$$

where $q_{ab} = e^{\Delta E(a,b)/kT}$, and in the HP model, ΔE is defined as:

$$\Delta E(a, b) = \begin{cases} \varepsilon & \text{if } a = b = h \\ 0 & \text{otherwise} \end{cases} \tag{18}$$

$U(l)$ is again the number of neighbor-avoiding loops of length l on the two-dimensional lattice (l even), and $C(l)$, the number of neighbor-avoiding walks of length l . For large l we use approximations of the form $A\mu^l l^{\gamma-1}$ (Madras and Slade, 1993). For $U(l)$, we use $A = 1.3$, $\gamma = \frac{11}{32}$; for $C(l)$ we use $A = 0.034$, $\gamma = 0.5$; for both formulas we use $\mu = 2.3$. These values were chosen to approximate the results of exact enumeration; more precise or more theoretically-motivated values would be desirable.

4.5 Parsing algorithm

The grammar developed above is a linear CFG: no rule has more than one nonterminal symbol on its right-hand side. The algorithm to find the total weight of all derivations of a string using a linear CFG is shown below:

```

1: given words  $w_1, \dots, w_n$ , grammar  $G$ 
2: for  $l = 1$  to  $n$  do
3:   for  $i = 1$  to  $n - l + 1$  do
4:      $k = i + l$ 
5:     for all  $A \xrightarrow{p} w_i \cdots w_{k-1} \in G$  do
6:        $c[A, i, k] = c[A, i, k] + p$ 
7:     end for
8:     for all  $A \xrightarrow{p} w_i \cdots w_{j_1} B w_{j_2} \cdots w_k \in G$  do
9:        $c[A, i, k] = c[A, i, k] + c[B, j_1, j_2] \cdot p$ 
10:    end for
11:  end for
12: end for
13: return  $c[S, 1, n + 1]$ 

```

This algorithm computes the total weight Q for a given sequence. However, we may want to further group structures into bins in various ways and compute the total weight of each bin. For example, we might want to group structures according to their energy level and ask what the total contribution of each energy level is. Or we might group structures according to how many self-contacts are present or not present in the native structure (Chen and Dill, 2000).

To do this, we incorporate the bins into the nonterminal alphabet. For example, to group the derivations of grammar (17) by the number of self-contacts, we would write:

$$\begin{aligned}
& S^n \rightarrow G^n \\
& G^n \xrightarrow{C(l+m-1)} \overset{a_1}{\bullet} \overset{a_2}{\bullet} \cdots \overset{a_{l-1}}{\bullet} \overset{a_l}{\bullet} \overset{b_1}{\bullet} \overset{b_2}{\bullet} \cdots \overset{b_{m-1}}{\bullet} \overset{b_m}{\bullet} \\
& F^{n+1} \xrightarrow{\frac{1}{4} U(l+m) q_{a_1 b_m}} \overset{a_1}{\bullet} \overset{a_2}{\bullet} \cdots \overset{a_{l-1}}{\bullet} \overset{a_l}{\bullet} \overset{b_1}{\bullet} \overset{b_2}{\bullet} \cdots \overset{b_{m-1}}{\bullet} \overset{b_m}{\bullet} \\
& F^1 \xrightarrow{\frac{1}{4} U(l) q_{a_1 a_l}} \overset{a_1}{\bullet} \overset{a_2}{\bullet} \cdots \overset{a_{l-1}}{\bullet} \overset{a_l}{\bullet}
\end{aligned} \tag{19}$$

This grammar has multiple start symbols S^n , one for each bin. The total weight of S^n is the total weight of states with n self-contacts; it is easy to modify the above algorithm to calculate the total

weight of each. With this information, we can also calculate the mean and variance of the number of self-contacts, or the most likely structure for each number of self-contacts.

This algorithm has a running time of $O(n^2|G|)$, and $|G|$, in turn, is dependent on n and the number of bins. Assume there exists an upper bound B on the number of bins for a given string. If B does not exist, our algorithm may not terminate. Typically, however, B is polynomial in n . If the bins are energy levels, they are all linear combinations (with integer coefficients) of the ΔE 's, which are drawn from a fixed, finite set. If there is a number x such that each ΔE can be expressed as an integer multiple of x (it suffices for the ΔE 's to be all rational), then B will be linear in the number of self-contacts. If the number of self-contacts a single terminal can participate in is bounded (in this case, it is bounded to one), then $B \in O(n)$. Similar reasoning fixes asymptotic upper bounds on other binning schemes as well.

Then the grammar effectively has $O(n^2B)$ productions, because the third production can be instantiated in $O(n^2)$ ways for a given span of the input string. (The second production can as well, but because this production is used relatively infrequently, it is not the critical factor.) This would give a running time of $O(n^4B)$, whereas Chen and Dill give (we believe incorrectly) a complexity of $O(n^4)$.

The algorithm can be improved by rewriting the third production with the following:

$$\begin{aligned}
 & \mathbf{F} \xrightarrow{\frac{1}{4}U(l)q_{ab}} \mathbf{F}_{l,a} \\
 & \mathbf{F}_{l+1,a} \rightarrow \mathbf{F}_{l,a} \overset{b'}{\bullet} \overset{b}{\bullet} \\
 & \mathbf{F}_{l,a_1} \rightarrow \overset{a_1}{\bullet} \overset{a_2}{\bullet} \dots \overset{a_{l-1}}{\bullet} \overset{a_l}{\bullet} \mathbf{F}
 \end{aligned} \tag{20}$$

This reduces the effective grammar size to $O(n)$. When $B \in O(n)$, this gives a running time of $O(n^4)$, agreeing (we believe coincidentally) with Chen and Dill's analysis.

In practice it is more efficient to calculate the partition function (including energies, lengths, counts, and bins) offline, after discarding chart items which are not part of a complete derivation. Since the nonterminal indices for lengths and bins do not affect grammaticality, we can first parse using a grammar without lengths or bins, and then repars only the resulting forest with the full grammar.

5 The Chen-Dill RNA secondary structure model

Next, we present Chen and Dill's model for RNA secondary structures, that is, nested polymer graphs; we then present our reformulation of this model as a grammar.

5.1 Matrix formulation

Again we present a simplified version. Define the matrices \mathbf{f} and \mathbf{g} as follows:

$$\mathbf{f}_{ij} = \frac{1}{4} U(l) q_{a_i a_j} \mathbf{g}_{ij}^l \quad (21)$$

$$\mathbf{g}_{ij}^{l+1} = \mathbf{g}_{i,j-1}^l + \mathbf{g}_{i,j'}^l \otimes \mathbf{f}_{j'j} \quad l > 2 \quad (22)$$

$$\mathbf{g}_{ij}^2 = 1 + \mathbf{f}_{ij} \quad (23)$$

where \otimes is elementwise multiplication.

Then the partition function of the molecule $a_0 \cdots a_n$ is $\sum_l C(l) g_{0n}^l$.

5.2 As a grammar

$$\begin{aligned} S &\xrightarrow{C(l-1)} G^l \\ G^{l+1} &\xrightarrow{1} G^l \overset{a}{\bullet} \overset{b}{\bullet} \\ G^2 &\xrightarrow{1} \overset{a}{\bullet} \overset{b}{\bullet} \\ G^{l+1} &\xrightarrow{1} G^l F \\ G^2 &\xrightarrow{1} F \\ F &\xrightarrow{\frac{1}{4} U(l) q_{ab}} \overset{a}{\bullet} \overset{b}{\bullet} \quad l > 2 \end{aligned} \quad (24)$$

where the superscripts are counters used for measuring lengths.

Again we may add bins to the nonterminal alphabet, for example, counting self-contacts:

$$\begin{aligned}
S^n &\xrightarrow{C^{(l-1)}} G^{l,n} \\
G^{l+1,n} &\xrightarrow{1} G^{l,n} \overset{a}{\bullet} \overset{b}{\bullet} \\
G^{2,0} &\xrightarrow{1} \overset{a}{\bullet} \overset{b}{\bullet} \\
G^{l+1,m+n} &\xrightarrow{1} G^{l,m} F^n \\
G^{2,n} &\xrightarrow{1} F^n \\
F^{n+1} &\xrightarrow{\frac{1}{4}U^{(l)}q_{ab}} \overset{a}{\bullet} G^{l,n} \overset{b}{\bullet} \quad l > 2
\end{aligned} \tag{25}$$

The CFG versions of the full versions of Chen and Dill's two models are given in Appendix A. They develop the above grammar further by enforcing a constraint on the maximum number of self-contacts a single monomer can have, and local constraints on what kinds of faces can be adjacent to each other. In our experiments below, however, we have used the above grammar.²

5.3 Parsing algorithm

Below is the CKY algorithm for parsing a more general class of CFGs; indeed, any CFG can be converted into an equivalent CFG that belongs to this class:

- 1: given words w_1, \dots, w_n , grammar G
- 2: **for** $l = 1$ to n **do**
- 3: **for** $i = 1$ to $n - l + 1$ **do**
- 4: $k = i + l$
- 5: **for all** $A \xrightarrow{p} w_i \cdots w_{k-1} \in G$ **do**
- 6: $c[A, i, k] = c[A, i, k] + p$
- 7: **end for**
- 8: **for** $j = i + 1$ to $k - 1$ **do**
- 9: **for all** $A \xrightarrow{p} BC \in G$ **do**
- 10: $c[A, i, k] = c[A, i, k] + c[B, i, j] \cdot c[C, j, k] \cdot p$

²To be more precise, our implementation uses yet another grammar which is formally equivalent to this one.

```

11:     end for
12: end for
13: for all  $A \xrightarrow{p} B \in G$  in topological order do
14:      $c[A, i, k] = c[A, i, k] + c[B, i, k] \cdot p$ 
15: end for
16: end for
17: end for
18: return  $c[S, 1, n + 1]$ 

```

Two notes are in order. First, our grammar's second production is of the form $A \rightarrow Bc$, which our algorithm does not account for. This is easily remedied by either creating a dummy nonterminal C and replacing the rule with two rules $A \rightarrow BC$ and $C \rightarrow c$, or else by mixing the previous algorithm with this one. Second, line 13 iterates through rules $A \rightarrow B$ in "topological order"; by this we mean that if the grammar has rules $A \rightarrow B$ and $B \rightarrow C$, then the latter must be applied first. Such an ordering does exist for our grammar; if it did not, then our algorithm would not be applicable to it.

This algorithm has time complexity $O(|G|n^3)$. Here the effective size of the grammar is $O(nB^2)$, because of the fourth production, giving an overall time complexity of $O(n^4B^2)$. Chen and Dill analyze their algorithm (we believe incorrectly) as running in time $O(n^5B)$. When $B \in O(n)$, the two analyses happen to agree.

As above, it is faster in practice first to parse without the devices in the grammar that do not affect grammaticality, and then reparse only the paths that lead to successful derivations. In this case, the $l > 2$ check in the last production means that even in the first pass, we must keep track of l up to 3, but the values 3 and above can be conflated without affecting correctness.

6 Helix bundles

In Section 3 we described the Zimm-Bragg model of α -helices (Zimm and Bragg, 1959), which gives partition functions for conformations with local self-contacts. Chen and Dill’s model gives partition functions for conformations with nested nonlocal self-contacts, but thus far it has been impossible to compute partition functions for chain molecules having both local and nonlocal interactions, as in bundles of helices (Figure 6).

Figure 6

But the Zimm-Bragg model is formally a weighted finite-state automaton, as described above, and we have already shown that Chen and Dill’s model is equivalent to a weighted CFG. Therefore, we can use the machinery of formal grammars to combine these two models easily. The combined system would generate polymer graphs representing two-helix bundles, and the constructed weighted CFG would correctly calculate the partition function.

6.1 Adapting the Zimm-Bragg model to the HP square lattice model

Before integrating the Zimm-Bragg model into Chen and Dill’s model, we must first adapt it to the HP lattice model which underlies Chen and Dill’s model. This will apply both to our final grammar-based model as well as the exact enumeration we will evaluate it against.

In a real α -helix, the O atom of each amino acid is hydrogen-bonded to the NH of the fourth previous amino acid, making 3.6 residues per turn (see Figure 7); the bending of the chain occurs at the two bonds to the C^α ’s. The Zimm-Bragg model models the self-contacts by giving each an energy of ε_s , and it models the freezing by giving a conformation count of $r < 1$ to each bond angle, representing the relative lack of conformational freedom of a helix relative to random coil. Therefore the first turn of the helix should get a weight of $r^6 e^{-\varepsilon_s/kT}$. The r^6 factor is for the six bond angles frozen by the first hydrogen bond, and the $e^{-\varepsilon_s/kT}$ for the energy of the hydrogen bond. Then each subsequent monomer, because it freezes two more bond angles and adds one more hydrogen bond, gets a weight of $r^2 e^{-\varepsilon_s/kT}$. The simplest version of the model collapses the whole first turn into the first monomer; thus the first monomer gets a weight of σs , where $\sigma \approx r^4$ and $s \approx r^2 e^{-\varepsilon_s/kT}$, and

subsequent monomers get a weight of s .

Figure 7

Now on a square lattice, a helix is modeled as shown in Figure 8a. This is not completely accurate (cf. Figure 8b): only every other monomer creates a new self-contact, and the $(2i)$ th monomer is in contact with the $(2i - 3)$ rd monomer. Nevertheless, we still give every monomer an energy of ε_s , plus an energy of ε_{hh} for every hh contact.

Figure 8

Since we explicitly count conformations on a lattice, we are already counting random coil as having more conformations than helix—by a factor of approximately μ (the connective constant from above) per monomer. Ideally, then, the factor r would be superfluous. In that case we would simply give each helix unit a weight of $s = e^{-\varepsilon_s/kT}$. However, because the lattice model does not match reality very exactly, we keep a correction factor on s : $s = s_0 e^{-\varepsilon_s/kT}$, where $s_0 \approx \mu r^2$. Moreover, in the lattice model the first turn is no more difficult to form than subsequent turns. Therefore we must retain the σ parameter as well.

To summarize, the factors contributing the weight of a conformation are:

$$\begin{aligned}
 q_{hh} &= e^{-\varepsilon_{hh}/kT} && \text{for every hh contact} \\
 s &= s_0 e^{-\varepsilon_s/kT} && \text{for every non-initial monomer in a helix} \\
 \sigma s &&& \text{for the first monomer in a helix}
 \end{aligned}$$

6.2 Integrating helices into the Chen-Dill model

For the two-helix bundle problem, our grammar is the CFG (25), and our finite-state automaton is shown in Figure 9. It is more complicated than the Zimm-Bragg model because it tries to model the shape of a helix in a square lattice.

Figure 9

The state q_c is for coil; the states q_{ij} are for helices, where i cycles through four values corresponding to the periodicity of the shape of a helix in a square lattice (see Figure 8a), and j is used to ensure that all helices are at least six monomers long.

The procedure for intersecting a context-free grammar with a finite-state automaton is due to Bar-Hillel et al. (1964, pp. 149–150). Given a CFG $G = \langle V, \Sigma, S, P \rangle$ and a finite-state automaton (without ϵ -transitions) $M = \langle \Sigma, Q, Q_0, Q_f, \delta \rangle$, the new CFG G' has nonterminal alphabet $Q \times V \times Q \cup S'$, where

S' is a new start symbol not in V ; and its production set consists of all productions of the form

$$\langle q_0, X, q_n \rangle \rightarrow \langle q_0, \alpha_1, q_1 \rangle \langle q_1, \alpha_2, q_2 \rangle \cdots \langle q_{n-1}, \alpha_n, q_n \rangle$$

where $X \rightarrow \alpha_1 \cdots \alpha_n \in P$ ($n > 0$), and for each i , either $\alpha_i \in V$ or else $\alpha_i \in \Sigma$ and $\langle q_{i-1}, \alpha_i, q_i \rangle \in \delta$; and

$$\langle q, X, q \rangle \rightarrow \epsilon$$

for all $q \in Q$, where $X \rightarrow \epsilon \in P$; and, finally,

$$S' \rightarrow \langle q_0, S, q_f \rangle$$

for all $q_0 \in Q_0, q_f \in Q_f$. The resulting grammar generates the language $L(G) \cap L(M)$.

The difference between this construction and one like Brown and Wilson's for pseudoknots is that the two component grammars are fully integrated, so that we may let them control each other however we please. For example, when two helices come together to form a bundle, self-contacts should only be allowed between monomers on the sides of the helices facing each other. Our original CFG had the rule $X \rightarrow Z$ which generated a self-contact; in the intersected grammar its corresponding productions include those of the form $\langle q_{ij}, X, q_{i'j'} \rangle \rightarrow \langle q_{ij}, Z, q_{i'j'} \rangle$. We may now stipulate that $i, i' \in \{0, 2\}$ for such productions, which ensures that only one side of a helix may participate in nonlocal contacts.

6.3 Computing the partition function

We compute the partition function offline as described above, with some modifications. First, we incorporate the weights σ and s from the Zimm-Bragg model, and an additional factor q_{hh} for every helical hh contact.

Second, previously we estimated the number of conformations of a loop of length l as $\frac{1}{4}U(l)$ and the number of conformations of the tails with combined length l as $C(l - 1)$. Now that these loops and tails may include helices, which are rigid, we must adjust these estimates. Our current approach is simply to count each helix as a single step in a neighbor-avoiding walk, without trying to take into account the length of the helix.

Finally, since the grammar is most error-prone with closed conformations, we use a special set of rules for loops of length eight or less, shown below (weights are conformation counts only):

$$\begin{aligned}
F &\xrightarrow{1} \text{HHHX'HHH} \\
F &\xrightarrow{1} \text{UX'HHH} \\
F &\xrightarrow{1} \text{HHHX'U} \\
F &\xrightarrow{1} \text{HFH} \\
F &\xrightarrow{1} \text{X'X'X'} \\
F &\xrightarrow{4} \text{CCCCCC} \\
X' &\rightarrow F \mid C \\
U &\rightarrow H \mid C \\
H &\rightarrow \langle q, a, q' \rangle \quad \langle q, a, q' \rangle \in \delta_H \\
C &\rightarrow \langle q, a, q' \rangle \quad \langle q, a, q' \rangle \in \delta_C
\end{aligned} \tag{26}$$

where $\delta_H \cup \delta_C$ comprises those triples $\langle q, a, q' \rangle$ such that there is an arrow from q to q' in Figure 9: δ_H contains the arrows with weight s or σs^2 , and δ_C contains the arrow with weight equal to one. These rules do not exhaustively cover all possible loops of length eight or less; a number of possibilities were left out somewhat arbitrarily to limit overcounting. Chen and Dill's steric compatibility matrices might be a more principled solution.

6.4 Results

We compared our parser against exact enumeration in various ways. First, we tried the sequence $w_1 = \text{hphhphhphhphhphhphh}$, which has minimum-energy structures high in both helix units and hh contacts (Figure 10a). In this experiment and those below, $\sigma = 0.01$. Figure 11 shows the average number of helix units as a function of the parameters s and q_{hh} , and Figure 12 shows superimposed cross-sections of these functions; the output of the parser qualitatively agrees with that of the exact enumerator. Figures 13 and 14 compare the average number of hh contacts; again there is qualitative agreement, except for the region $s < 2$ of Figure 14b, which is not very important because such low

Figure 10

Figure 11

Figure 12

Figure 13

Figure 14

values for s make helix units unfavorable relative to random coil, which is unrealistic.

Figure 15

We next tried the sequence $w_2 = \text{hppphhppphhppphhpph}$, which has structures with many hh contacts and structures with many helix units, but not both at the same time (Figure 10b). Figures 15 and 16 compare the average number of helix units; as with the first sequence, the parser's output qualitatively agrees with the exact enumerator's.

Figure 16

Figure 17

Figures 17 and 18 compare the average number of hh contacts. The agreement is not as good as before for high q_{hh} due to both overcounting of some structures and undercounting of others. For example, the grammar is able to generate small spirals, but does not have the "memory" needed to keep the spiral from colliding with itself. Figure 19a shows an unviable conformation generated by the grammar. Such structures are causing the parser to overestimate the average number of hh contacts for high q_{hh} and low s (again, such low values are unrealistic). On the other hand, the grammar does not have a rule that would let helices contact each other at right angles. Figure 19b shows a viable conformation with three helices that the grammar does not generate. Missing this structure is causing the parser to underestimate the average number of hh contacts for high q_{hh} and s . Nevertheless, the parser agrees with the enumerator in predicting that the number of hh contacts should decrease with s , in contrast to the first sequence.

Figure 18

Figure 19

The grammar could be modified to try to improve agreement, and this deserves further work. It is possible that in a three-dimensional lattice, the problem of collisions will be less severe because a greater proportion of structures will have viable conformations.

7 Conclusion

We have shown how the machinery of formal language theory can be used to combine two existing polymer models into a novel model of simple helix bundles. The new model is not more powerful than the Chen-Dill model in a computational sense; that is, since it is again a context-free grammar, it could in principle be translated back into the matrix formulation, although it would be quite complicated.

To generalize to more complicated conformations than RNA secondary structures, like pseudoknots, we must move beyond context-free grammars to more expressive formalisms, for example, tree adjoining grammars (Joshi *et al.*, 1975; Joshi and Schabes, 1997). The principles set forth here generalize easily to such formalisms. The main challenge lies in identifying the elementary components of these more complicated structures (in other words, what are the “faces” of a non-nested polymer graph?), and finding their energy increments and conformation counts.

Two related approaches may prove useful in this regard. Uemura *et al.* (1999) generate RNA pseudoknots using a tree-adjoining grammar. A standard parsing algorithm for tree adjoining grammars can be modified as we have done for CKY, giving a time complexity of $O(n^6|G|^2B^2)$, where $|G|$ is the number of elementary trees. Rivas and Eddy (2000) use a formalism called crossed-interaction grammar, which, when formalized, is equivalent to set-local multi-component TAG (Weir, 1988), an extension of TAG. The actual grammar they use for generating RNA pseudoknots is intermediate in formal power between single-component TAG and set-local two-component TAG.³ Again, their dynamic-programming algorithm (Rivas and Eddy, 1999) can be modified as we have done for CKY, giving the same time complexity as TAG: $O(n^6|G|B^2)$, where $|G|$ is the number of productions.

Both of these approaches decompose conformations with crossing links into elementary structures and assign energy increments to each structure. However, these energy increments have not all been experimentally determined so far, so that both approaches must make approximations for pseudoknots. It also remains to be seen whether conformation counts can be assigned to such elementary

³These results are based on some preliminary work of Chiang and Joshi and several discussions at the Workshop on Language Modeling of Biological Data at the University of Pennsylvania in February 2001, and were presented by Joshi in an invited talk at the Pacific Symposium on Biocomputing, 2002.

structures that will give good estimates of overall conformation counts.

Acknowledgments

This work was partially supported by NSF-ITR grant EIA02-05456. We would like to thank Adam Lucas and Julia Hockenmaier for their valuable input.

A Full grammars for the Chen-Dill models

The grammars (that is, rule schemata for the grammars) for hairpin conformations and RNA secondary structures are shown in Figures 20 and 21, respectively. Note that where Chen and Dill use G^* , we use F , to reduce typographical clutter.

In both grammars, $S_t(l)_{c_1c_2}$, $(Y_{t_1t_2})_{c_1c_2}$, ω_c , α , and $\Delta E(x, y)$ are parameters which are either experimentally determined or determined by the underlying two-dimensional lattice model for counting conformations, as explained by Chen and Dill (1998).

The second grammar is like the grammar developed above, but duplicates G into two nonterminals, G for the tails and K for the faces. It adds two subscripts to the nonterminal alphabet: the first is called the graph type or face type and is used primarily to ensure that no monomer has more than two contacts; the second is called the inlet/outlet type and is used to account locally for excluded volume between two adjacent faces.

The productions we have given for $K_{LR,c}^l$ when $l > 2$ do not conform exactly to Chen and Dill's equations. Their equations are a simplification from a previous version of their model (Chen and Dill, 1995) (in that they drop a certain operator Q), and the productions we have given here are closer to the older equations. It is possible to reproduce the newer equations exactly, but for us it would be slightly more complicated.

The difficulty is that because a subgraph's weight vector is defined in terms of n subgraphs' weight vectors, we can no longer use a matrix to express this relationship. Chen and Dill's older solution is to give all the subgraphs but the rightmost the same type as the supergraph, and the rightmost is allowed to have a different type. The newer solution is to make the subgraphs dependent on each other in a nonlocal way. Our solution is to give all the subgraphs the same type, which can be different from the supergraph's.

The energy of the rule for F_{lc} is given as $\Delta E(a_1, a_l)$; this was not stated correctly in Chen and Dill's paper (Chen, p.c.). In the rule for F_{tc} ($t \neq l$), the condition $l > 2$ was omitted from the paper (Chen, p.c.); also, the right-hand side of their equation has a summation over t , which appears to be a

mistake (Song Cao, p.c.).

Finally, it would seem the following rules need to be added for completeness:

$$G_{0c} \rightarrow \overset{a}{\bullet} \text{---} \overset{b}{\bullet}$$

to allow left tails

$$G_{1c} \rightarrow F_{1c} \mid G_{0c}F_{1c} \mid G_{1c}F_{1c}$$

to allow a lone self-contact

Figure 20

Figure 21

References

- Bar-Hillel, Y., Perles, M., and Shamir, E., 1964. On formal properties of simple phrase structure grammars, 116–150. In Bar-Hillel, Y., ed., Language and Information: Selected Essays on their Theory and Application. Addison-Wesley, Reading, MA.
- Chen, S.-J. and Dill, K. A., 1995. Statistical thermodynamics of double-stranded polymer molecules. J. Chem. Phys. 103, 5802–5813.
- Chen, S.-J. and Dill, K. A., 1998. Theory for the conformational changes of double-stranded chain molecules. J. Chem. Phys. 109, 4602–4616.
- Chen, S.-J. and Dill, K. A., 2000. RNA folding energy landscapes. Proceedings of the National Academy of Sciences 97, 646–651.
- Crescenzi, P., Goldman, D., Papadimitriou, C., Piccolboni, A., and Yannakakis, M., 1998. On the complexity of protein folding. Journal of Computational Biology 5, 423–465.
- Joshi, A. K., Levy, L., and Takahashi, M., 1975. Tree adjunct grammars. Journal of Computer and System Sciences 10, 136–163.
- Joshi, A. K. and Schabes, Y., 1997. Tree-adjointing grammars, 69–124. In Rosenberg, G. and Salomaa, A., eds., Handbook of Formal Languages and Automata, volume 3. Springer-Verlag, Heidelberg.
- Kasami, T., 1965. An efficient recognition and syntax algorithm for context-free languages. Technical Report AFCRL-65-758, Air Force Cambridge Research Laboratory, Bedford, MA.
- Lau, K. F. and Dill, K. A., 1989. A lattice statistical mechanics model of the conformation and sequence spaces of proteins. Macromolecules 22, 3986–3997.
- Lifson, S. and Roig, A., 1961. On the theory of helix-coil transition in polypeptides. J. Chem. Phys. 34, 1963–1974.
- Madras, N. and Slade, G., 1993. The Self-Avoiding Walk. Birkhäuser, Boston.

- Rivas, E. and Eddy, S. R., 1999. A dynamic programming algorithm for RNA structure prediction including pseudoknots. Journal of Molecular Biology 285, 2053–2068.
- Rivas, E. and Eddy, S. R., 2000. The language of RNA: a formal grammar that includes pseudoknots. Bioinformatics 16, 334–340.
- Schellman, J. A., 1958. The factors affecting the stability of hydrogen-bonded polypeptide structures in solution. J. Phys. Chem. 62, 1485–1494.
- Searls, D. B., 1988. Representing genetic information with formal grammars, 386–391. In Proceedings of the Seventh National Conference on Artificial Intelligence (AAAI). AAAI Press/MIT Press.
- Searls, D. B., 1992. The linguistics of DNA. American Scientist 80, 579–591.
- Uemura, Y., Hasegawa, A., Kobayashi, S., and Yokomori, T., 1999. Tree adjoining grammars for RNA structure prediction. Theoretical Computer Science 210, 277–303.
- Weir, D. J., 1988. Characterizing Mildly Context-Sensitive Grammar Formalisms. Ph.D. thesis, University of Pennsylvania.
- Younger, D. H., 1967. Recognition and parsing of context-free languages in time n^3 . Information and Control 10, 189–208.
- Zimm, B. H. and Bragg, J. K., 1959. Theory of the phase transition between helix and random coil in polypeptide chains. Journal of Chemical Physics 31, 526–535.

List of Figures

1	Example CFG derivation of RNA secondary structure	33
2	The Zimm-Bragg model as a weighted finite-state automaton	35
3	All possible conformations of a 5-mer on a square lattice (modulo rotational and reflectional symmetry). For the groupings (a), (b), and (c), see Figure 4.	37
4	Example polymer graphs, corresponding to the conformations in Figure 3(a), (b), and (c)	39
5	Nested polymer graph, with one face shaded. The straight edges represent covalent bonds, and the curved edges represent self-contacts.	41
6	Two-helix bundle.	43
7	Two amino acids connected by a peptide bond.	45
8	α -helix: (a) on a square lattice; (b) somewhat more accurately.	47
9	Automaton for helices in a square lattice. Nodes with the same label represent the same state; the state is shown in multiple locations for visual clarity. The full automaton has the union of the transitions shown in all six diagrams. The initial states are $\{q_c, q_{00}, q_{20}\}$; the final states are $\{q_c, q_{12}, q_{32}\}$	49
10	Some favorable structures for example sequences: (a) Sequence w_1 ; (b) Sequence w_2 . Hydrophobic monomers (h) are indicated by black circles; polar monomers (p) by white circles.	51
11	Comparison against exact enumeration. Sequence w_1 ; helix units versus s and q_{hh} . (a) Exact enumeration. (b) Parser.	53
12	Comparison against exact enumeration. Sequence w_1 ; helix units versus s . (a) $q_{hh} = 1$. (b) $q_{hh} = 10$	55
13	Comparison against exact enumeration. Sequence w_1 ; hh contacts versus s and q_{hh} . (a) Exact enumeration. (b) Parser.	57

14	Comparison against exact enumeration. Sequence w_1 ; hh contacts versus s . (a) $q_{hh} = 1$. (b) $q_{hh} = 10$	59
15	Comparison against exact enumeration. Sequence w_2 ; helix units versus s and q_{hh} . (a) Exact enumeration. (b) Parser.	61
16	Comparison against exact enumeration. Sequence w_2 ; helix units versus s . (a) $q_{hh} = 1$. (b) $q_{hh} = 10$	63
17	Comparison against exact enumeration. Sequence w_2 ; hh contacts versus s and q_{hh} . (a) Exact enumeration. (b) Parser.	65
18	Comparison against exact enumeration. Sequence w_2 ; hh contacts versus s . (a) $q_{hh} = 1$. (b) $q_{hh} = 10$	67
19	Examples of grammar overcounting (a) and undercounting (b). Hydrophobic monomers (h) are indicated by black circles; polar monomers (p) by white circles.	69
20	Grammar for hairpin conformations.	71
21	Grammar for RNA secondary structures.	73

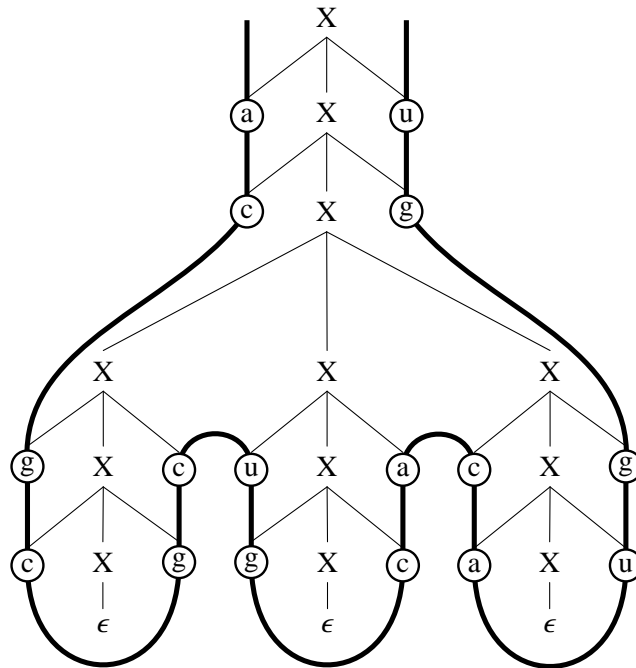


Figure 1: Example CFG derivation of RNA secondary structure, with superimposed primary structure.

Nonterminal symbols other than X are suppressed for clarity.



Chiang, Joshi, Dill

Figure 1 (of 21)

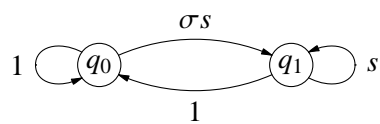


Figure 2: The Zimm-Bragg model as a weighted finite-state automaton.



Chiang, Joshi, Dill

Figure 2 (of 21)

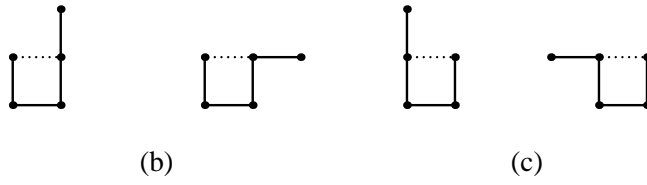
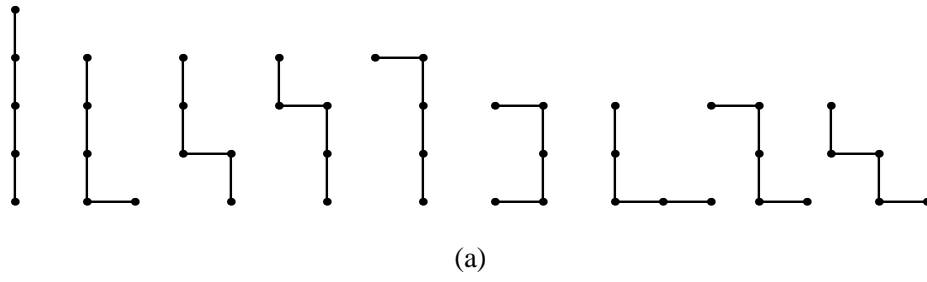


Figure 3: All possible conformations of a 5-mer on a square lattice (modulo rotational and reflectional symmetry). For the groupings (a), (b), and (c), see Figure 4.



Chiang, Joshi, Dill

Figure 3 (of 21)

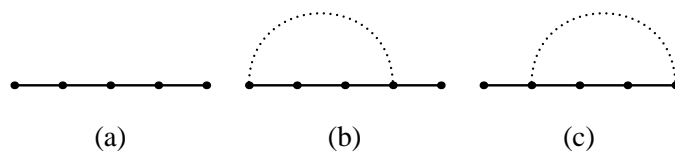


Figure 4: Example polymer graphs, corresponding to the conformations in Figure 3(a), (b), and (c)



Chiang, Joshi, Dill

Figure 4 (of 21)

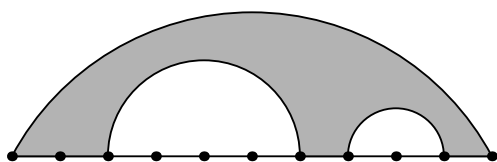


Figure 5: Nested polymer graph, with one face shaded. The straight edges represent covalent bonds, and the curved edges represent self-contacts.



Chiang, Joshi, Dill

Figure 5 (of 21)

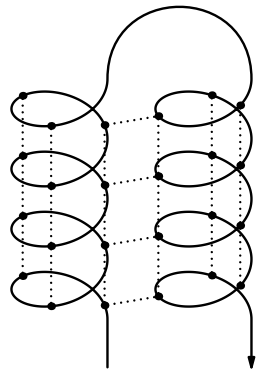


Figure 6: Two-helix bundle.



Chiang, Joshi, Dill

Figure 6 (of 21)

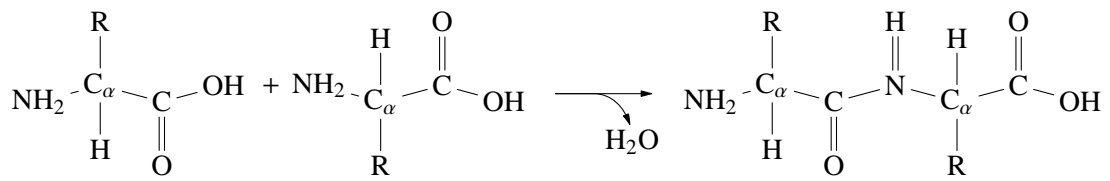


Figure 7: Two amino acids connected by a peptide bond.



Chiang, Joshi, Dill

Figure 7 (of 21)

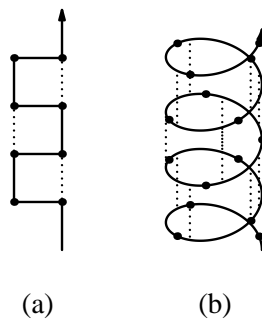


Figure 8: α -helix: (a) on a square lattice; (b) somewhat more accurately.



Chiang, Joshi, Dill

Figure 8 (of 21)

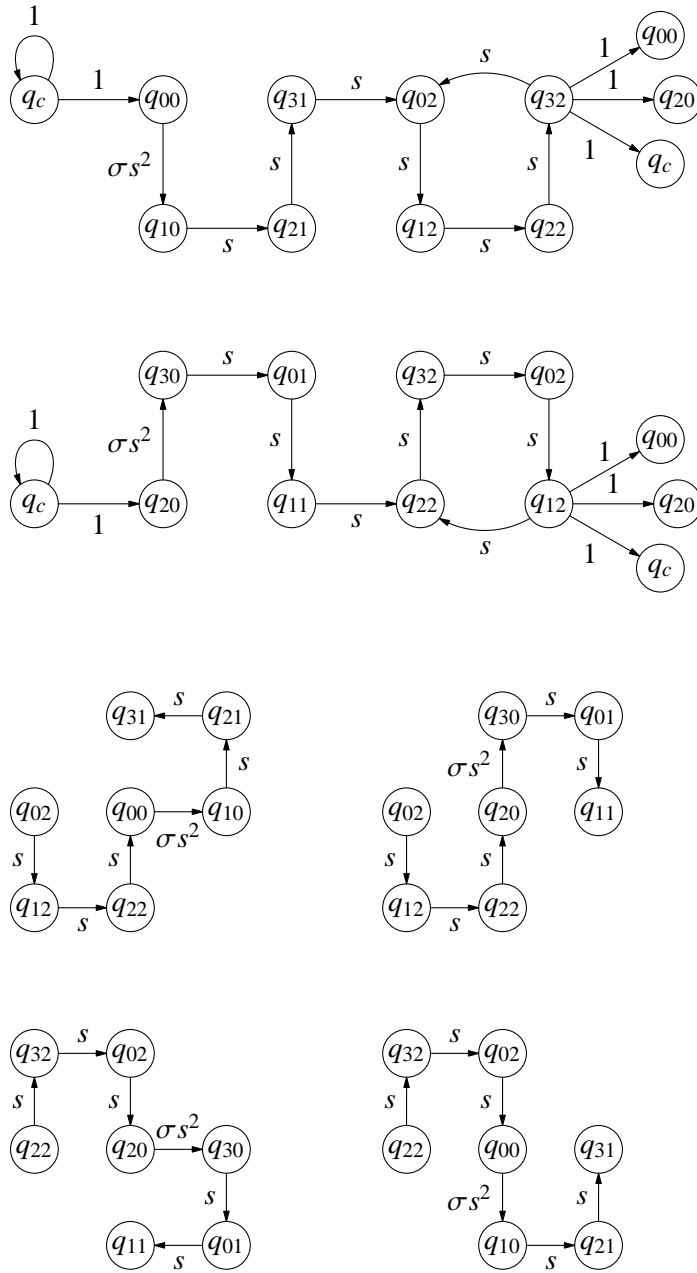


Figure 9: Automaton for helices in a square lattice. Nodes with the same label represent the same state; the state is shown in multiple locations for visual clarity. The full automaton has the union of the transitions shown in all six diagrams. The initial states are $\{q_c, q_{00}, q_{20}\}$; the final states are $\{q_c, q_{12}, q_{32}\}$.



Chiang, Joshi, Dill

Figure 9 (of 21)

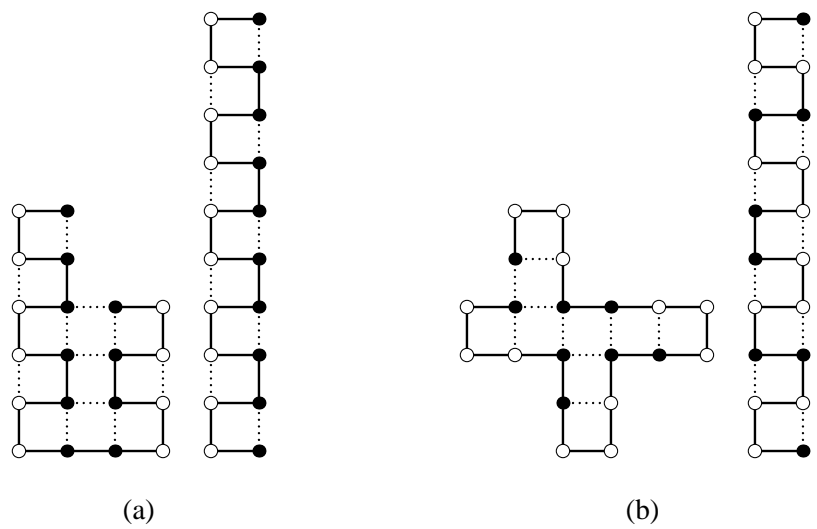
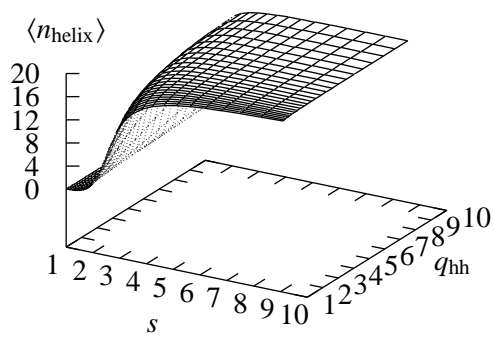


Figure 10: Some favorable structures for example sequences: (a) Sequence w_1 ; (b) Sequence w_2 .
 Hydrophobic monomers (h) are indicated by black circles; polar monomers (p) by white circles.

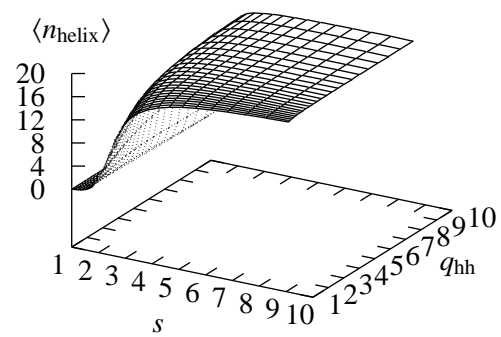


Chiang, Joshi, Dill

Figure 10 (of 21)



(a)



(b)

Figure 11: Comparison against exact enumeration. Sequence w_1 ; helix units versus s and q_{hh} . (a) Exact enumeration. (b) Parser.



Chiang, Joshi, Dill

Figure 11 (of 21)

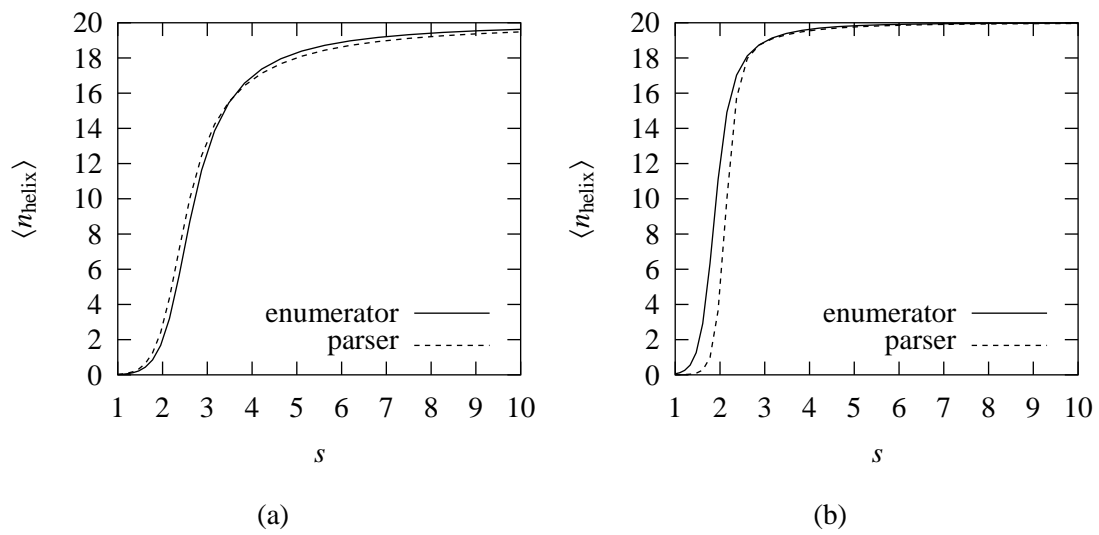


Figure 12: Comparison against exact enumeration. Sequence w_1 ; helix units versus s . (a) $q_{\text{hh}} = 1$. (b) $q_{\text{hh}} = 10$.



Chiang, Joshi, Dill

Figure 12 (of 21)

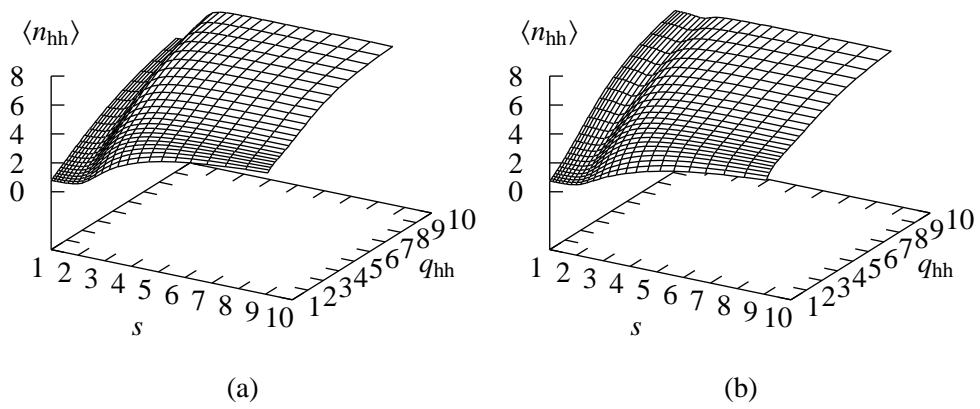
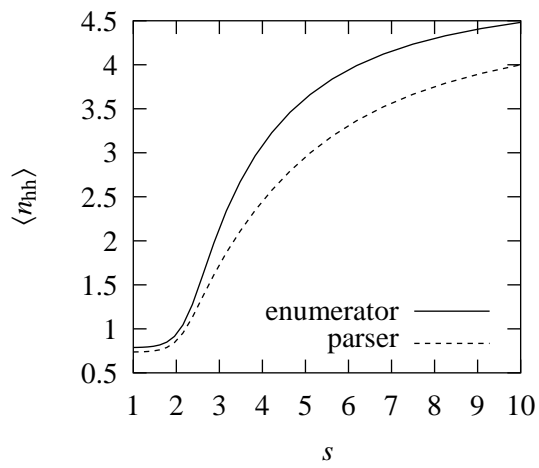


Figure 13: Comparison against exact enumeration. Sequence w_1 ; hh contacts versus s and q_{hh} . (a) Exact enumeration. (b) Parser.

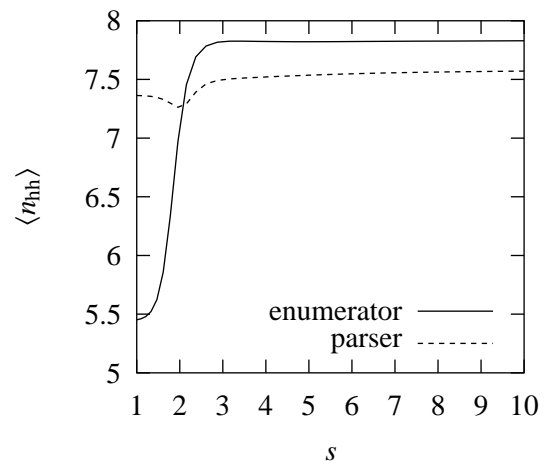


Chiang, Joshi, Dill

Figure 13 (of 21)



(a)



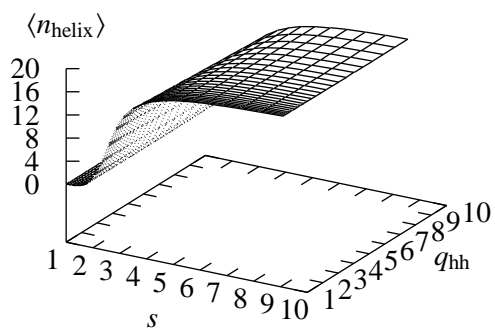
(b)

Figure 14: Comparison against exact enumeration. Sequence w_1 ; hh contacts versus s . (a) $q_{hh} = 1$.
 (b) $q_{hh} = 10$.

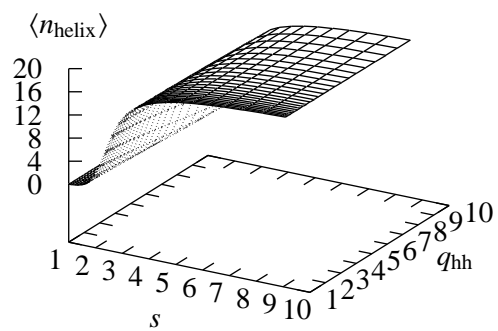


Chiang, Joshi, Dill

Figure 14 (of 21)



(a)



(b)

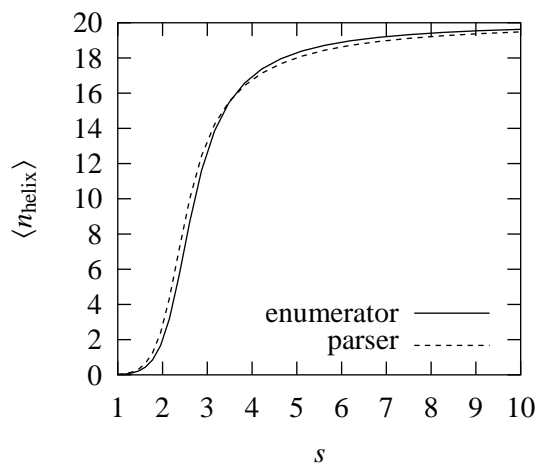
Figure 15: Comparison against exact enumeration. Sequence w_2 ; helix units versus s and q_{hh} . (a)

Exact enumeration. (b) Parser.

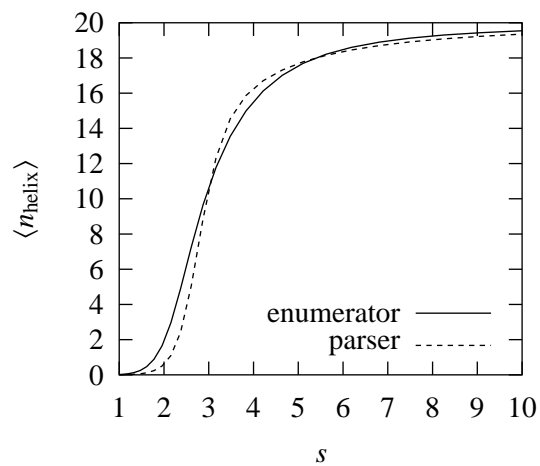


Chiang, Joshi, Dill

Figure 15 (of 21)



(a)



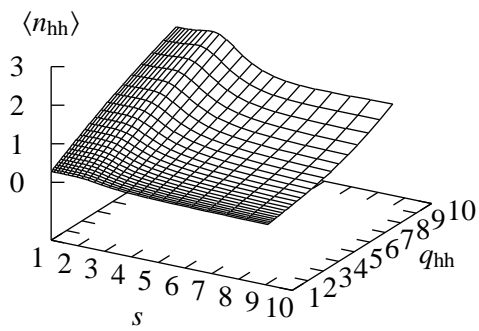
(b)

Figure 16: Comparison against exact enumeration. Sequence w_2 ; helix units versus s . (a) $q_{\text{hh}} = 1$. (b) $q_{\text{hh}} = 10$.

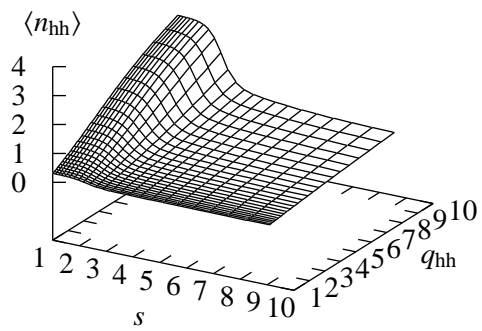


Chiang, Joshi, Dill

Figure 16 (of 21)



(a)



(b)

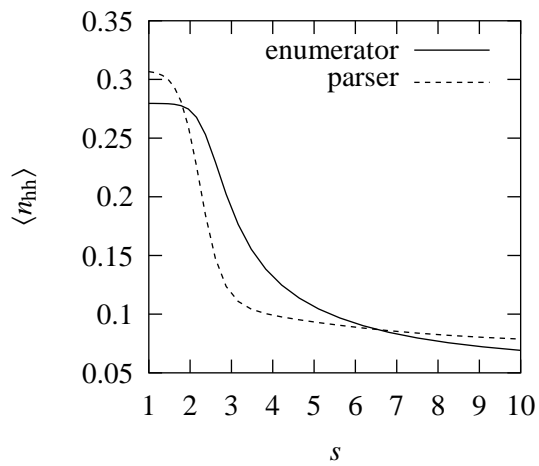
Figure 17: Comparison against exact enumeration. Sequence w_2 ; hh contacts versus s and q_{hh} . (a)

Exact enumeration. (b) Parser.

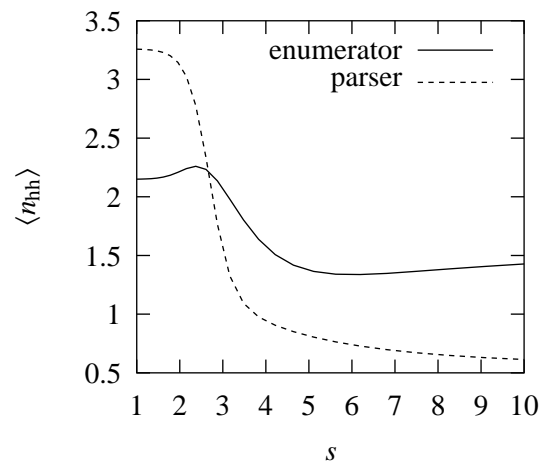


Chiang, Joshi, Dill

Figure 17 (of 21)



(a)



(b)

Figure 18: Comparison against exact enumeration. Sequence w_2 ; hh contacts versus s . (a) $q_{hh} = 1$.
 (b) $q_{hh} = 10$.



Chiang, Joshi, Dill

Figure 18 (of 21)

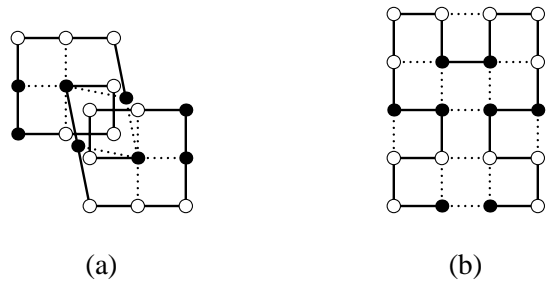


Figure 19: Examples of grammar overcounting (a) and undercounting (b). Hydrophobic monomers (h) are indicated by black circles; polar monomers (p) by white circles.



Chiang, Joshi, Dill

Figure 19 (of 21)

$$\begin{array}{lll}
\mathbf{S} \rightarrow \mathbf{G}_{tc} & 1 & \\
\mathbf{G}_{tc} \rightarrow \overset{a_1}{\bullet} \overset{a_2}{\bullet} \cdots \overset{a_{l-1}}{\bullet} \overset{a_l}{\bullet} \mathbf{F}_{tc} \overset{b_1}{\bullet} \overset{b_2}{\bullet} \cdots \overset{b_{m-1}}{\bullet} \overset{b_m}{\bullet} & \omega_c(l, m) & l, m \geq 0 \\
\mathbf{F}_{Lc} \rightarrow \overset{a_1}{\bullet} \mathbf{F}_{t'c'} \overset{b_1}{\bullet} \overset{b_2}{\bullet} \cdots \overset{b_{m-1}}{\bullet} \overset{b_m}{\bullet} & (\mathbf{S}_L(l+1)\mathbf{Y}_{L'})_{cc'} \exp(-\Delta E(a, b_m)/kT) & m > 0 \\
\mathbf{F}_{Mc} \rightarrow \overset{a_1}{\bullet} \overset{a_2}{\bullet} \cdots \overset{a_{l-1}}{\bullet} \overset{a_l}{\bullet} \mathbf{F}_{t'c'} \overset{b_1}{\bullet} \overset{b_2}{\bullet} \cdots \overset{b_{m-1}}{\bullet} \overset{b_m}{\bullet} & (\mathbf{S}_M(l+m)\mathbf{Y}_{Mt'})_{cc'} \exp(-\Delta E(a_1, b_m)/kT) & l, m > 0 \\
\mathbf{F}_{Rc} \rightarrow \overset{a_1}{\bullet} \overset{a_2}{\bullet} \cdots \overset{a_{l-1}}{\bullet} \overset{a_l}{\bullet} \mathbf{F}_{t'c'} \overset{b}{\bullet} & (\mathbf{S}_R(l+1)\mathbf{Y}_{R'})_{cc'} \exp(-\Delta E(a_1, b)/kT) & l > 0 \\
\mathbf{F}_{Ic} \rightarrow \overset{a_1}{\bullet} \overset{a_2}{\bullet} \cdots \overset{a_{l-1}}{\bullet} \overset{a_l}{\bullet} & \mathbf{S}_I(l)_{cc'} \exp(-\Delta E(a_1, a_l)/kT) & l > 0
\end{array}$$

$$t, t' \in \{\text{L, M, R, I}\}$$

$$c, c' \in \{1, 2, 3, 4\}$$

Figure 20: Grammar for hairpin conformations.



Chiang, Joshi, Dill

Figure 20 (of 21)

$S \rightarrow G_{0c} \mid G_{1c} \mid G_{2c}$	1	
$G_{0c} \rightarrow G_{0c} \overset{a}{\bullet} \overset{b}{\bullet} \mid G_{1c} \overset{a}{\bullet} \overset{b}{\bullet}$	α	
$G_{1c} \rightarrow F_{Lc} \mid F_{Mc}$	1	
$G_{1c} \rightarrow G_{0c}F_{Lc} \mid G_{0c}F_{Mc} \mid G_{1c}F_{Mc}$	1	
$G_{2c} \rightarrow F_{Rc} \mid F_{LR,c}$	1	
$G_{2c} \rightarrow G_{0c}F_{Rc} \mid G_{0c}F_{LR,c} \mid G_{1c}F_{Rc}$	1	
$F_{Ic} \rightarrow \overset{a_1}{\bullet} \overset{a_2}{\bullet} \cdots \overset{a_{l-1}}{\bullet} \overset{a_l}{\bullet}$	$S_I(l)_{cc'} \exp(-\Delta E(a_1, a_l)/kT)$	
$F_{tc} \rightarrow \overset{a}{\bullet} \overset{b}{\bullet} \overset{t}{\bullet} \overset{c'}{\bullet}$	$(S_t(l)Y_{tM})_{cc'} \exp(-\Delta E(a, b)/kT)$	$l > 2, t \neq I$
$K_{Ic}^l \rightarrow \overset{a_1}{\bullet} \overset{a_2}{\bullet} \cdots \overset{a_{l-1}}{\bullet} \overset{a_l}{\bullet}$	1	
$K_{Lc}^l \rightarrow K_{Lc}^{l-1} \overset{a}{\bullet} \overset{b}{\bullet} \mid K_{LR,c}^{l-1} \overset{a}{\bullet} \overset{b}{\bullet}$	1	
$K_{Mc}^l \rightarrow K_{Mc}^{l-1} \overset{a}{\bullet} \overset{b}{\bullet} \mid K_{Rc}^{l-1} \overset{a}{\bullet} \overset{b}{\bullet}$	1	
$K_{Rc}^l \rightarrow \overset{a}{\bullet} \overset{b}{\bullet} K_{Rc}^{l-1} \mid \overset{a}{\bullet} \overset{b}{\bullet} K_{LR,c}^{l-1}$	1	
$K_{LR,c}^l \rightarrow K_{LR,c}^{l-1} F_{t'c}$	1	$l > 2, t' \in \{M, R, I\}$
$K_{LR,c}^l \rightarrow K_{Lc}^{l-1} F_{t'c}$	1	$l > 2$
$K_{LR,c}^2 \rightarrow F_{Mc} \mid F_{Ic}$	1	

$$l > 0$$

$$t, t' \in \{L, M, R, LR, I\}$$

$$c, c' \in \{1, 2, 3, 4\}$$

Figure 21: Grammar for RNA secondary structures.



Chiang, Joshi, Dill

Figure 21 (of 21)