Formal grammars for estimating partition functions of double-stranded chain molecules

David Chiang
Dept of Computer and Information Science
200 S 33rd St
Philadelphia PA 19104-6389
dchiang@cis.upenn.edu

Aravind K. Joshi
Dept of Computer and Information Science
200 S 33rd St
Philadelphia PA 19104-6389
joshi@cis.upenn.edu

ABSTRACT

This paper looks at an algorithm due to Chen and Dill for estimating the partition functions of a restricted subclass of double-stranded polymers, and shows how it can be translated into a context-free grammar, so that their algorithm reduces to a variant of the CKY parsing algorithm. Our formulation clarifies the structure of Chen and Dill's algorithm, leading to revised complexity analyses and an optimization in one case, and lays a formal foundation for generalizing this method to more complex cases by the use of grammars beyond context-free power such as tree adjoining grammars.

1. INTRODUCTION

An important problem in the statistical mechanics of biomolecules is the computation of the *partition function* of a molecule, which is used to predict various physical properties of the molecule, including its stable conformations and conformational changes. Chen and Dill [1, 2] propose a model for a restricted subclass of molecules which gives an efficient approximate solution to this problem.

The goal of this paper is to show how Chen and Dill's model can be translated into a context-free grammar (CFG), so that their algorithm reduces to a variant of the CKY parsing algorithm [7, 14]. This formulation clarifies the structure of the algorithm, leading to a revised complexity analysis and an optimization in one case, and lays a formal foundation for generalizing this method to more complex cases using, for example, tree adjoining grammars [4, 6].

Molecules like RNAs or proteins are *sequences* of building blocks—nucleotides for RNA or amino acids for proteins—but they do not lie in straight lines in space. Rather, bonds, which we will call *self-contacts*, may form between nonadjacent members of the sequence, folding it into a certain shape in space. A pattern of self-contacts may be represented by a *polymer graph* (see Figure 1) in which straight edges represent the linear structure of the sequence, and curved edges represent the self-contacts. A polymer graph constrains but does not completely determine the shape of a molecule in space. In short, one sequence may correspond to many different polymer graphs, and one polymer graph may correspond to many different shapes or *conformations*.

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

$$Q = \sum_j e^{-E_j/kT} \tag{1}$$

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. Since all the conformations corresponding to a single polymer graph have the same energy, we may rewrite this as:

$$Q = \sum_j \Omega_j e^{-E_j/kT} \tag{2}$$

where j ranges over polymer graphs, Ω_j is the number of conformations corresponding to polymer graph j, and E_j is the energy of those conformations.

From a computational standpoint, the problem is not simply to find the sum Q, but the contribution of various parts to the sum. For example, if we normalize (1) to sum to 1, the individual terms give us the probabilities of the conformations; likewise, normalizing (2) gives the probabilities of the polymer graphs. Chen and Dill wish to compute the contribution of each energy level to Q, that is, to compute each term of the outer summation in:

$$Q = \sum_E \sum_{\substack{j \\ E_j=E}} \Omega_j e^{-E/kT}. \tag{3}$$

This computation divides into two parts: for each energy level E, we must compute Ω_j, and we must perform the inner summation, which is over all polymer graphs at energy level E.

The key to Chen and Dill's model 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 1 is nested, but not linearly nested, because the white regions are disjoint.) RNA secondary structures are conformations of nested poly-

.

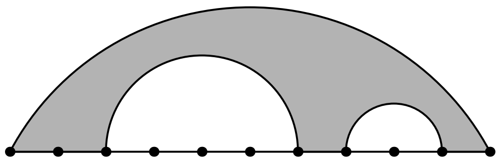

Figure 1: Nested polymer graph, with one face shaded. The straight edges represent covalent bonds, and the curved edges represent self-contacts.

mer 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 1).

Then Ω_j, 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. Moreover, since any restrictions on what types of faces can combine with each other are strictly local, Chen and Dill's algorithm can use dynamic programming to efficiently sum over all possible combinations with a particular energy.

2. DEFINITIONS

This hierarchical decomposition of polymer graphs suggests that nested polymer graphs might be modeled by a CFG (and linearly nested polymer graphs by a linear CFG), following Searls [11]. Under this interpretation, the nonterminal symbols correspond roughly to edges, and the productions correspond roughly to faces.

The computation of Q is accomplished by augmenting each production with a *weight* p, which we write as

$$A \xrightarrow{p} \alpha_1 \cdots \alpha_n$$

where the α_i are either terminals or nonterminals. The weight of a derivation is the product of the weights of the productions used in the derivation, and the weight of a string is the sum of the weights of all possible derivations of the string.

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.

All this can be computed by a straightforward extension of a standard parsing algorithm like CKY. However, if we want to group polymer graphs according to energy level and compute the contribution to Q from each energy level, we need some more machinery, borrowed from the attribute grammars of Knuth [8].

We attach to each nonterminal symbol a set of *attributes*, and, for each occurrence of a nonterminal A in a derivation, we give each attribute v a value, which we write $v(A)$. These

```
1: given words $w_1, \ldots, w_n$, 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:             $c[A, i, k, e(A)] = c[A, i, k, e(A)] + p$
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:             for all $E_1 \in \mathcal{V}$ do
10:                $E = e(A)$ computed for $e(B) = E_1$
11:                $c[A, i, k, E] = c[A, i, k, E] + c[B, j_1, j_2, E_1] \cdot p$
12:            end for
13:        end for
14:    end for
15: end for
16: return $\{\langle E, c[S, 1, n+1, E]\rangle \mid E \in \mathcal{V}\}$
```

Figure 2: Algorithm for linear CFG.

values are defined by *attribute equations* associated with each production $A \to \alpha_1 \cdots \alpha_n$, which define each attribute of A in terms of the attribute values of the α_i, if any.[1] Define the *v-value* of a derivation rooted by S to be $v(S)$.

Attributes provide a flexible way of computing various functions on derivations, including energies. For example, for every production $A \to \alpha_1 \cdots \alpha_n$, we can write the attribute equation

$$e(A) = \Delta E + \sum_{\alpha_i \in V} e(\alpha_i)$$

where ΔE is the energy increment due to the outer link of the corresponding face, and V is the set of nonterminals. Then the e-value of a derivation will be the total energy of the corresponding polymer graph.

Putting the weights and attributes together, then, we can obtain the terms of the summation (3) simply by computing, for each E, the total weight of those derivations with e-value E.

3. ALGORITHMS

To do this, we modify the CKY algorithm to use a separate chart for each energy level, so that it can simultaneously compute $g(E)e^{-E/kT}$ for all values of E. Let $\mathcal{V}$ be the set of possible energy levels of any polymer graph or subgraph, that is, the set of possible e-values of any derivation or subderivation. Note that $\mathcal{V}$ must be finite for a given string, or else the algorithm will not terminate. It is sufficient for the grammar to be finitely ambiguous, which is the case for the grammars induced from Chen and Dill's equations (see Appendix A).

The algorithm for linear CFG is shown in Figure 2. Its running time is $O(n^2|G||\mathcal{V}|)$, where n is the length of the input string and $|G|$ is the number of productions in the grammar. The algorithm for a general CFG in Chomsky normal form[2] is shown in Figure 3, whose running time is $O(n^3|G||\mathcal{V}|^2)$. Both of these algorithms are implicit in Chen and Dill's equations,

[1]Knuth calls attributes defined in this way *synthesized attributes*. He also discusses *inherited attributes*, which we do not consider here.

[2]In Chomsky normal form, every production is either of the form $A \to BC$ or $A \to a$. Every CFG (which does not generate ϵ) can be converted into this form. Alternatively, we could have started with a more general algorithm instead of CKY.

```
1: given words $w_1, \ldots, w_n$, 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:       $c[A, i, k, \Delta E] = c[A, i, k, \Delta E] + p$
7:     end for
8:     for $j = i + 1$ to $k - 1$ do
9:       for all $A \xrightarrow{p} BC \in G$ do
10:        for all $E_1, E_2 \in \mathcal{V}$ do
11:          $E = e(A)$ computed for $e(B) = E_1$, $e(C) = E_2$
12:          $c[A, i, k, E] = c[A, i, k, E]+$
                 $c[B, i, j, E_1] \cdot c[C, j, k, E_2] \cdot p$
13:        end for
14:      end for
15:    end for
16:  end for
17: end for
18: return $\{\langle E, c[S, 1, n + 1, E]\rangle \mid E \in \mathcal{V}\}$
```

Figure 3: Algorithm for general CFG in Chomsky normal form (modified CKY).

but here the use of dynamic programming is made explicit.

The grammars, which can be read more or less directly off of Chen and Dill's equations, are shown in Figures 4 and 5 and described in more detail in Appendix A. Our main concern here is in the size of the grammars, since the running time of the algorithms depends on it.

For hairpin conformations, the linear CFG has productions with arbitrarily long right-hand sides. Thus it would seem that the parser (Figure 2) must search through an infinite number of productions in lines 5 and 8. But in fact these productions are instantiations of rule schemata, and if the parser only instantiates each schema as necessary, line 5 only has to consider a bounded number of productions, and line 8 only has to consider $O(n^2)$ productions. This is how Chen and Dill's equations work as well. They analyze this algorithm as running in time $O(n^4)$, but our analysis, which is based on a more precise algorithmic description, gives a running time of $O(n^4|\mathcal{V}|)$.

However, the $O(n^2)$ effective grammar size is due to a single rule schema (for G^*_{Mc}, see Figure 4) which has the form

$$A \xrightarrow{p(l)} a_1 \cdots a_k A' a_{k+1} \cdots a_l.$$

If we replace this schema with the following:

$$A \xrightarrow{p(l)} A^l$$

$$A^i \xrightarrow{1} A^{i-1}a$$

$$A^i \xrightarrow{1} aA^{i-1}$$

$$A^0 \xrightarrow{1} A'$$

the effective $|G|$ becomes $O(n)$ (because the index l ranges from 0 to n). Compacting the grammar in this way improves the running time to $O(n^3|\mathcal{V}|)$.

For RNA secondary structures (general CFG), the grammar size is again $O(n)$, because it contains nonterminals of the form K^l_{tc} (see Figure 5), where l ranges between 2 and n. Chen and Dill analyze this algorithm as running in time $O(n^5|\mathcal{V}|)$, but our analysis gives a running time of $O(n^4|\mathcal{V}|^2)$.

How large is $|\mathcal{V}|$ for a given string? For computing energy levels, $|\mathcal{V}|$ consists of 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 (e. g., if the ΔE's are all rational), then $|\mathcal{V}|$ will be linear in the number of self-contacts. If the number of self-contacts a single terminal can participate in is bounded (in Chen and Dill's model, it is bounded to two), then $|\mathcal{V}| \in O(n)$, in agreement with Chen and Dill's analysis.

This gives us a running time of $O(n^4)$ for linear CFG (after compacting the grammar) and $O(n^6)$ for general CFG. By coincidence, seemingly, these bounds match Chen and Dill's analysis. However, their analysis underestimates the contribution of $|\mathcal{V}|$ to the overall complexity; moreover, compacting the linear CFG as described above is not merely a difference of analysis, but an optimization of Chen and Dill's original algorithm.

4. EXTENSIONS

4.1 Other groupings

There are other ways to group the terms of the partition function that are of interest, and attribute equations provide enough flexibility to specify a wide variety of such groupings. For example, Chen and Dill elsewhere [3] group conformations according to how many contacts belong to the native structure and how many do not. In this case (assuming as before that the number of self-contacts per terminal is bounded), $|\mathcal{V}| \in O(n^2)$.

4.2 More complex conformations

To generalize to more complicated conformations than RNA secondary structures, like pseudoknots, we must move to more expressive formalisms than CFG, for example, tree adjoining grammar (TAG). 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. [12] 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 $O(n^6|G|^2|\mathcal{V}|^2)$, where $|G|$ is the number of elementary trees. Rivas and Eddy [10] use a formalism called *crossed-interaction grammar*, which, when formalized, is equivalent to set-local multi-component TAG [13], 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 [9] can be modified as we have done for CKY, giving the same time

[3]These 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 [5].

complexity as TAG: $O(n^6|G||\mathcal{V}|^2)$, 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 structures that will give good estimates of overall conformation counts.

5. CONCLUSION

In summary, we have given a formally and computationally more precise account of Chen and Dill's method for computing partition functions of double-stranded chain molecules. From a computational standpoint, this account clarifies the structure of the algorithm, leading to a revised complexity analysis and an optimization for the linear case (hairpin conformations). From a formal standpoint, this account separates the principles of the model from the particular restrictions it chooses, laying a foundation for relaxing those restrictions to handle more complex conformations.

6. ACKNOWLEDGEMENTS

This work was partially supported by NSF grant SBR-89-20230. We would like to thank Shi-Jie Chen, Ken Dill, and the two anonymous reviewers, all of whom helped to improve the accuracy and presentation of our paper.

7. REFERENCES

[1] Shi-Jie Chen and Ken A. Dill. Statistical thermodynamics of double-stranded polymer molecules. *J. Chem. Phys.*, 103(13):5802–5813, 1995.

[2] Shi-Jie Chen and Ken A. Dill. Theory for the conformational changes of double-stranded chain molecules. *J. Chem. Phys.*, 109(11):4602–4616, 1998.

[3] Shi-Jie Chen and Ken A. Dill. RNA folding energy landscapes. *Proc. Natl. Acad. Sci.*, 97(2):646–651, 2000.

[4] A. K. Joshi, L.S. Levy, and M. Takahashi. Tree adjunct grammars. *J. Comp. Sys. Sci.*, 10(1):136–163, 1975.

[5] Aravind K. Joshi. Structural descriptions of formal systems and folded structures of biomolecules. In *Pacific Symposium on Bioinformatics*, Kauai, HI, January 2002. Invited talk.

[6] Aravind K. Joshi and Yves Schabes. Tree-adjoining grammars. In Grzegorz Rosenberg and Arto Salomaa, editors, *Handbook of Formal Languages and Automata*, volume 3, pages 69–124. Springer-Verlag, Heidelberg, 1997.

[7] Tadao Kasami. An efficient recognition and syntax algorithm for context-free languages. Technical Report AFCRL-65-758, Air Force Cambridge Research Laboratory, Bedford, MA, 1965.

[8] Donald E. Knuth. Semantics of context-free languages. *Mathematical Systems Theory*, 2(2):127–145, 1968. Corrections published in 5(1):95–96.

[9] Elena Rivas and Sean R. Eddy. A dynamic programming algorithm for RNA structure prediction including pseudoknots. *Journal of Molecular Biology*, 285:2053–2068, 1999.

[10] Elena Rivas and Sean R. Eddy. The language of RNA: a formal grammar that includes pseudoknots. *Bioinformatics*, 16(4):334–340, 2000.

[11] David B. Searls. The linguistics of DNA. *American Scientist*, 80:579–591, 1992.

[12] Yasuo Uemura, Aki Hasegawa, Satoshi Kobayashi, and Takashi Yokomori. Tree adjoining grammars for RNA structure prediction. *Theoretical Computer Science*, 210:277–303, 1999.

[13] David J. Weir. *Characterizing Mildly Context-Sensitive Grammar Formalisms*. PhD thesis, Univ. of Pennsylvania, 1988.

[14] Daniel H. Younger. Recognition and parsing of context-free languages in time n^3. *Information and Control*, 10(2):189–208, 1967.

APPENDIX

A. GRAMMARS

The grammars (that is, rule schemata for the grammars) for hairpin conformations and RNA secondary structures are shown in Figures 4 and 5, respectively. The w_i are terminal symbols, and S is the start symbol. The G_{tc}, G^*_{tc}, and K^l_{tc} are nonterminals; these symbols and their subscripts are explained by Chen and Dill [2].

The weight of each production is $\omega e^{-\Delta E/kT}$, where ω and ΔE are as listed next to the grammars, and each production $A \to \alpha_1 \cdots \alpha_n$ has the attribute equations

$$E(A) = \Delta E + \sum_{\alpha_i \in V} E(\alpha_i)$$

$$x = \begin{cases} x(\alpha_1) & \text{if } \alpha_1 \in V \\ \alpha_1 & \text{otherwise} \end{cases}$$

$$y = \begin{cases} y(\alpha_n) & \text{if } \alpha_n \in V \\ \alpha_n & \text{otherwise} \end{cases}$$

where, again, ΔE is as listed next to the grammars. The purpose of the x and y attributes is to keep track of the leftmost and rightmost symbols in the yield of each nonterminal.

In both grammars, $\mathbf{S}_t(l)_{c_1c_2}$, $(\mathbf{Y}_{t_1t_2})_{c_1c_2}$, 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 [2]. Note that because the weight of a production depends on ΔE, and ΔE depends on the attributes x and y, the weights depend on the attributes, which is not strictly allowed under our definitions. Since x and y are drawn from a finite set, it would be possible to write these attributes into the grammar, but this would complicate the grammar unnecessarily. For the parsing algorithms given here, there is no problem with letting the weights depend on the attributes.

In the second grammar, the productions we have given for $\mathrm{K}^L_{\mathrm{LR},c}$ 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 [1], 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.

Production	ω	ΔE	
$\mathrm{S} \to \mathrm{G}_{tc}$	1	0	
$\mathrm{G}_{tc} \to \mathrm{G}^*_{tc}$	ω_c	0	
$\mathrm{G}^*_{\mathrm{L}c} \to \mathrm{G}^*_{t'c_2} w_{j'+1} \cdots w_j$	$\mathbf{S}_{\mathrm{L}}(j - j' + 2)_{cc_1}$	$\Delta E(x, y)$	$(\mathbf{Y}_{\mathrm{L}t'})_{c_1 c_2} = 1$
$\mathrm{G}^*_{\mathrm{M}c} \to w_i \cdots w_{i'-1} \mathrm{G}^*_{t'c_2} w_{j'+1} \cdots w_j$	$\mathbf{S}_{\mathrm{M}}(j - j' + i' - i + 2)_{cc_1}$	$\Delta E(x, y)$	$(\mathbf{Y}_{\mathrm{M}t'})_{c_1 c_2} = 1$
$\mathrm{G}^*_{\mathrm{R}c} \to w_i \cdots w_{i'-1} \mathrm{G}^*_{t'c_2}$	$\mathbf{S}_{\mathrm{R}}(i' - i + 2)_{cc_1}$	$\Delta E(x, y)$	$\mathbf{Y}_{\mathrm{R}t'})_{c_1 c_2} = 1$
$\mathrm{G}^*_{\mathrm{I}c} \to w_i \cdots w_j$	$\mathbf{S}_{\mathrm{I}}(j - i + 1)_c$	$\Delta E(x, y)$	

$$t, t' \in \{\mathrm{L}, \mathrm{M}, \mathrm{R}, \mathrm{I}\}$$
$$c, c_1, c_2 \in \{1, 2, 3, 4\}$$

Figure 4: Rule schemata for grammar for hairpin conformations.

Production	ω	ΔE	
$\mathrm{S} \to \mathrm{G}_{0c} \mid \mathrm{G}_{1c} \mid \mathrm{G}_{2c}$	1	0	
$\mathrm{G}_{0c} \to \mathrm{G}_{0c} w_j \mid \mathrm{G}_{1c} w_j$	α	0	
$\mathrm{G}_{1c} \to \mathrm{G}^*_{\mathrm{L}c} \mid \mathrm{G}^*_{\mathrm{M}c} \mid \mathrm{G}_{0c}\mathrm{G}^*_{\mathrm{L}c} \mid \mathrm{G}_{0c}\mathrm{G}^*_{\mathrm{M}c} \mid \mathrm{G}_{1c}\mathrm{G}^*_{\mathrm{M}c}$	1	0	
$\mathrm{G}_{2c} \to \mathrm{G}^*_{\mathrm{R}c} \mid \mathrm{G}^*_{\mathrm{LR},c} \mid \mathrm{G}_{0c}\mathrm{G}^*_{\mathrm{R}c} \mid \mathrm{G}_{0c}\mathrm{G}^*_{\mathrm{LR},c} \mid \mathrm{G}_{1c}\mathrm{G}^*_{\mathrm{R}c}$	1	0	
$\mathrm{G}^*_{\mathrm{I}c} \to w_i \cdots w_j$	$\mathbf{S}_{\mathrm{I}}(j - i + 1)_c$	$\Delta E(x, y)$	
$\mathrm{G}^*_{tc} \to \mathrm{K}^l_{t'c_2}$	$\mathbf{S}_t(l)_{cc_1}$	$\Delta E(x, y)$	$l > 2, (\mathbf{Y}_{t\mathrm{M}})_{c_1 c_2} = 1, t \neq \mathrm{I}$
$\mathrm{K}^l_{\mathrm{I}c} \to w_i \cdots w_{i+l-1}$	1	0	
$\mathrm{K}^l_{\mathrm{L}c} \to \mathrm{K}^{l-1}_{\mathrm{L}c} w_j \mid \mathrm{K}^{l-1}_{\mathrm{LR},c} w_j$	1	0	
$\mathrm{K}^l_{\mathrm{M}c} \to \mathrm{K}^{l-1}_{\mathrm{M}c} w_j \mid \mathrm{K}^{l-1}_{\mathrm{R}c} w_j$	1	0	
$\mathrm{K}^l_{\mathrm{R}c} \to w_i \mathrm{K}^{l-1}_{\mathrm{R}c} \mid w_i \mathrm{K}^{l-1}_{\mathrm{LR},c}$	1	0	
$\mathrm{K}^l_{\mathrm{LR},c} \to \mathrm{K}^{l-1}_{\mathrm{LR},c_1} \mathrm{G}^*_{t'c_2}$	1	0	$l > 2, (\mathbf{Y}_{\mathrm{MM}})_{cc_2} = 1, t' \notin \{\mathrm{L}, \mathrm{LR}\}$
$\mathrm{K}^l_{\mathrm{LR},c} \to \mathrm{K}^{l-1}_{\mathrm{L}c_1} \mathrm{G}^*_{t'c_2}$	1	0	$l > 2, (\mathbf{Y}_{\mathrm{MM}})_{cc_2} = 1$
$\mathrm{K}^2_{\mathrm{LR},c} \to \mathrm{G}^*_{\mathrm{M}c} \mid \mathrm{G}^*_{\mathrm{I}c}$	1	0	

$$l > 0$$
$$t, t' \in \{\mathrm{L}, \mathrm{M}, \mathrm{R}, \mathrm{LR}, \mathrm{I}\}$$
$$c, c_1, c_2 \in \{1, 2, 3, 4\}$$

Figure 5: Rule schemata for grammar for RNA secondary structures.