Sampling and Approximation in the Context of RNA Secondary Structure Prediction

Algorithms and Studies Based on Stochastic Context-Free Modeling

Vom Fachbereich Informatik der
Technischen Universität Kaiserslautern
zur Verleihung des akademischen Grades
Doktor der Naturwissenschaften
(Dr. rer. nat.)
genehmigte

Dissertation

von

Anika Schulz

Dekan: Prof. Dr. Arnd Poetzsch-Heffter

Prüfungskommission:

Vorsitz:	Prof. Dr. Klaus Schneider
Erster Berichterstatter:	Prof. Dr. Markus Nebel
Zweiter Berichterstatter:	Prof. Dr. Rolf Backofen

Datum der wissenschaftlichen Aussprache: 12. Oktober 2012

D 386

Für Frederick

Zusammenfassung

Die Entwicklung effizienter Algorithmen zur Vorhersage der Sekundärstruktur von RNA-Molekülen ist eine der grundlegenden Problemstellungen aus dem Bereich der strukturellen Biologie und stellt eine anspruchsvolle Aufgabe dar. Aktuell existierende Vorhersage-Algorithmen nutzen die Methode der dynamischen Programmierung und basieren entweder auf einem allgemein gültigen thermodynamischen Modell oder auf einem speziellen probabilistischen Modell. Letzteres wird traditionellerweise über eine stochastische kontext-freie Grammatik realisiert. Bisher wurden zu diesem Zweck meist relativ einfache und übersichtliche Grammatiken betrachtet. Zudem wurde trotz der über die vergangenen Jahren stetig gewachsenen Akzeptanz und Wertschätzung für statistische Ansätze bislang noch kein Sampling-Algorithmus basierend auf einem stochastischen Modell der RNA-Struktur formuliert. Des Weiteren besitzen alle bekannten Vorhersage-Werkzeuge für eine beliebige Eingabe-Sequenz der Länge n die gleichen Zeit- und Speicher-Komplexitäten von $\mathcal{O}(n^3)$ und $\mathcal{O}(n^2)$ im Worst-Case, was deren Anwendung in der Praxis aufgrund der oftmals großen Länge der zu betrachtenden RNA-Moleküle in einem gewissen Maße einschränkt. Entsprechend steht für Biologen eine geringere Wartezeit für die Berechnung der Ergebnisse im Vordergrund, wobei aber auch die Qualität nicht allzu sehr leiden sollte.

All diese Aspekte werden hier adressiert, und zwar durch die Beschreibung von Algorithmen und die Durchführung umfangreicher Studien basierend auf ausgeklügelten stochastisch kontext-freien Grammatiken deren Komplexität in etwa der von thermodynamischen Ansätzen entspricht. Dabei wird in allen Fällen das Konzept des Sampling angewandt. Zusätzlich kommt die aus der Theoretischen Informatik bekannte Technik der Approximation zum Einsatz, um eine Beschleunigung der Worst-Case-Laufzeit von RNA-Strukturvorhersagen zu erreichen.

Zu Beginn wird ein Sequenz-unabhängiger Random Sampler für beliebige RNA-Klassen basierend auf (gewichtetem) Unranking entwickelt. Der resultierende Algorithmus benötigt zum Erzeugen jeder beliebigen möglichen Sekundärstruktur einer gegebenen festen Größe n lediglich $\mathcal{O}(n \cdot \log(n))$ Zeit. Die beobachteten Ergebnisse sind von hoher Qualität, was die praktische Anwendbarkeit bestätigt.

Hinsichtlich der Vorhersage der RNA-Struktur wird zunächst ein neuer probabilistischer Sampling-Algorithmus präsentiert, der statistisch repräsentative und reproduzierbare Samples der Menge aller zulässigen Sekundärstrukturen konform zu einer gegebenen Eingabe-Sequenz erzeugt. Diese Methode zieht die möglichen Faltungen anhand einer durch eine geeignete Grammatik induzierten Verteilung. Insbesondere werden auch einige (neuartige) Verfahren zur Berechnung von Vorhersagen aus zuvor erzeugten Samples vorgestellt. Beide Varianten benötigen $\mathcal{O}(n^3)$ Zeit und $\mathcal{O}(n^2)$ Speicher für Sequenz-Länge n und können qualitativ hochwertige (Vorhersage-)Ergebnisse liefern. Dies wird belegt durch umfangreiche Evaluationen.

Zur Reduzierung der Laufzeit von Algorithmen zur Vorhersage der RNA-Struktur ohne übermäßigen Qualitätsverlust wird eine innovative heuristische Methode für statistisches Sampling entwickelt, welche lediglich $\mathcal{O}(n^2)$ Zeit zum Sampeln einer festen Anzahl möglicher Strukturen für eine gegebene Sequenz der Länge n benötigt. Da eine sinnvolle Vorhersage effizient aus einer beliebigen Sample-Menge berechnet werden kann, wird so im Vergleich zu allen existierenden präzisen Methoden die Worst-Case-Laufzeit um einen linearen Faktor reduziert. Insbesondere wird auch eine (heuristische) Alternative zu der üblicherweise für statistisches Sampling verwendeten Sampling-Strategie eingeführt. Eine Validierung des entwickelten heuristischen Sampling-Ansatzes durch Vergleich mit mehreren gängigen Vorhersage-Werkzeugen deutet darauf hin, dass die Vorhersagen konkurrenzfähig sind, aber die Betrachtung größerer Sample-Mengen nötig sein kann.

Abstract

Predicting secondary structures of RNA molecules is one of the fundamental problems of and thus a challenging task in computational structural biology. Existing prediction methods basically use the dynamic programming principle and are either based on a general thermodynamic model or on a specific probabilistic model, traditionally realized by a stochastic context-free grammar. To date, the applied grammars were rather simple and small and despite the fact that statistical approaches have become increasingly appreciated over the past years, a corresponding sampling algorithm based on a stochastic RNA structure model has not yet been devised. In addition, basically all popular state-of-the-art tools for computational structure prediction have the same worst-case time and space requirements of $\mathcal{O}(n^3)$ and $\mathcal{O}(n^2)$ for sequence length n, limiting their applicability for practical purposes due to the often quite large sizes of native RNA molecules. Accordingly, the prime demand imposed by biologists on computational prediction procedures is to reach a reduced waiting time for results that are not significantly less accurate.

We here deal with all of these issues, by describing algorithms and performing comprehensive studies that are based on sophisticated stochastic context-free grammars of similar complexity as those underlying thermodynamic prediction approaches, where all of our methods indeed make use of the concept of sampling. We also employ the approximation technique known from theoretical computer science in order to reach a heuristic worst-case speedup for RNA folding.

Particularly, we start by describing a way for deriving a sequence-independent random sampler for an arbitrary class of RNAs by means of (weighted) unranking. The resulting algorithm may generate any secondary structure of a given fixed size n in only $\mathcal{O}(n \cdot \log(n))$ time, where the results are observed to be accurate, validating its practical applicability.

With respect to RNA folding, we present a novel probabilistic sampling algorithm that generates statistically representative and reproducible samples of the entire ensemble of feasible structures for a particular input sequence. This method actually samples the possible foldings from a distribution implied by a suitable (traditional or length-dependent) grammar. Notably, we also propose several (new) ways for obtaining predictions from generated samples. Both variants have the same worst-case time and space complexities of $\mathcal{O}(n^3)$ and $\mathcal{O}(n^2)$ for sequence length n. Nevertheless, evaluations of our sampling methods show that they are actually capable of producing accurate (prediction) results.

In an attempt to resolve the long-standing problem of reducing the time complexity of RNA folding algorithms without sacrificing much of the accuracy of the results, we invented an innovative heuristic statistical sampling method that can be implemented to require only $\mathcal{O}(n^2)$ time for generating a fixed-size sample of candidate structures for a given sequence of length n. Since a reasonable prediction can still efficiently be obtained from the generated sample set, this approach finally reduces the worst-case time complexity by a liner factor compared to all existing precise methods. Notably, we also propose a novel (heuristic) sampling strategy as opposed to the common one typically applied for statistical sampling, which may produce more accurate results for particular settings. A validation of our heuristic sampling approach by comparison to several leading RNA secondary structure prediction tools indicates that it is capable of producing competitive predictions, but may require the consideration of large sample sizes.

Acknowledgements

Throughout my studies, the preparation and compilation of this work, I was supported by several people and institutions in many different ways and without this support, the present thesis would not have been possible.

First of all, I am very grateful to my parents, Petra and Dieter, for enabling my studies in the first place and for their never-ending support during all of my life and my studies. I also want to thank the rest of my family: my brother Stefan, my aunt Karin and especially all of my grandparents, for their continued support in countless ways since I was a little child, for helping me in countless occasions, and for always believing in me.

Furthermore, I am indebted to my advisor, Markus Nebel for the time and knowledge he devoted to support my research for so many years. He always managed to provide me with useful advise and feedback, as well as with valuable freedom. I am also very grateful that he encouraged me to carry on when problems seemed unsolvable, and never questioned the successful completion of my PhD studies.

I would like to thank Rolf Backofen for acting as examiner.

Moreover, I wish to express my gratitude to the Carl Zeiss Foundation for promoting my research over the past years.

I also want to thank my current and former colleagues for helping me in miscellaneous occasions, as well as my friends and my parents-in-law for their mental support, especially during the final stages of my dissertation.

Last but not least, my greatest thanks belong to my husband Frederick, for bearing with me during times of (really) long work and when I was absent-minded, and for patiently waiting until these times were over or else, stopping me whenever I was too stressed out for my own good. In addition, without his consistent encouragement to carry on and his never-ending believe in me, I would not have finished this thesis. I am truly grateful for all these things and for everything else he has done for me.

Contents

Chapter 1

Introduction

This chapter briefly motivates the objectives of the present thesis and provides a short overview of the research plan to be pursued with respect to our global goal.

1.1. Background and Motivation

Computational prediction of RNA secondary structures from a single sequence has been an extensively studied research topic over the past decades and since become of great relevance for practical applications in structural biology. The main reason for this lies in the fact that it is well-known that the biological function of an RNA molecule is heavily dependent on its structure [GJMM+05]. However, the experimental determination of RNA structure via wet-lab techniques (see, for example [Fel07]) is usually time-consuming and expensive, motivating the development of computer based approaches for facilitating RNA structure analysis. Furthermore, RNA folding is hierarchical [BW97, TB99]. In fact, much of the final three-dimensional tertiary structure of functional RNAs is determined by the two-dimensional secondary structure of the molecule [TB99]; functional RNAs include messenger RNA (encoding proteins) and other non-protein coding RNA (see, for instance [Edd01]). Unlike proteins, the structure of RNA molecules generally partitions quite cleanly between secondary and tertiary hierarchical levels [BW97, Woo10, Woo11]. Therefore, prediction of RNA secondary structure is generally possible without knowledge of tertiary structure and can be seen as a first step in RNA modeling and folding algorithms.

Basically, two main situations need to be considered: In the first case, several sequences of homologous RNAs are known. Then, comparative sequence analysis can be applied (see, for example [PTW99]). This approach assumes that structures that have been conserved by evolution are far more likely to be the functional form and thus derives a consensus secondary structure by an alignment of RNA sequences, that is a secondary structure common to the considered set of sequences. In the second case, only one sequence is known. This case is known as the basic RNA folding problem, that is predicting the secondary structure of a single RNA strand which is given as an input (see, for instance [NJ80, ZS81]). In this thesis, we will exclusively deal with the latter case.

Anyway, accurate prediction of RNA secondary structure from a single sequence is an unsolved computational challenge. This is due to the fact that any RNA molecule can be folded in many different ways, resulting in an vast number of possible secondary structures for molecule lengths typically dealt with in practice. Actually, the number of possible foldings grows exponentially with the length of the RNA sequence [Wat78, SW78]. Thus, taking all these foldings into consideration is out of reach. Therefore, the main idea of computational prediction methods is to compute those foldings that are most realistic or most stable. For this reason, the up-to-date most successful approaches towards RNA structure prediction traditionally make use of the dynamic programming principle in order to compute a set of candidate structures that are considered the "best" for a given RNA sequence, that is the most optimal ones according to a particular objective function (see, for example [Edd04]).

Over the last decades, free energy minimization has become the most common technique to predict the secondary structure of an RNA molecule. This means the corresponding dynamic programming methods use free energy as their metric and produce a set of structures whose energy lies within a particular increment of the global minimum that can be reached according to the base sequence (see, for example [SHF+84, Zuk89b]). The reason why this approach has become prevalent is due to the fact that RNA molecules in solution arrange themselves so as to minimize the free energy of the entire system, such that it seems adequate to assume that the correct structure is the one with the lowest free energy.

An alternative methodology is based on principles of probabilistic modeling and is typically realized by utilizing stochastic context-free grammars or generalizations like conditional log-linear models as basis for deriving a corresponding dynamic programming method (see, for instance [KH99, DE04, DWB06]). Notably, in this thesis we call an approach probabilistic if and only if it abstracts from general thermodynamic models and instead tries to learn about the structural behavior of the molecules by training (a manageable number of) probabilistic

parameters from trusted RNA structure databases. The optimal solution is then traditionally defined as the most probable folding among all possible secondary structures for the considered sequence, according to the probability distribution induced by the learned parameters.

However, independent on the objective function used, a major problem of all these dynamic programming algorithms is that they inherently require cubic time and quadratic storage in the worst-case, limiting their applicability for long input sequences. Additionally, due to the strict deterministic optimization applied for determining (sub)optimal structures, it may happen that many foldings are proposed that share rather similar conformations and are thus redundant in some sense, since only structure with fundamental differences are of interest for RNA structure analysis. Notably, this motivated the use of shape based approaches, where classes of similar candidate foldings are represented by a corresponding shape abstraction in order to reduce the search space [GVR04, JRG08]. Moreover, due to the difficulties of both energy-based and probabilistic approaches to capture some specific molecular folding mechanisms or structural features and their inability to perfectly model all relevant aspects, only such foldings might potentially be predicted that are actually impossible to show up in nature.

In order to address these long-standing problems, several statistical sampling methods have been invented over the past years, such as for example [DL03, Pon08], which randomly sample secondary structures from the Boltzmann low-energy ensemble. Note that prediction methods based on random sampling represent a non-deterministic counterpart to the popular energy minimizing dynamic programming algorithms. Random sampling also differs from the probabilistic RNA structure prediction approach based on context-free modeling. Anyway, statistical sampling approaches like [DL03] have become increasingly appreciated, as they not only enable the generation of more diverse sets of candidate structures, but can also be used for characterizations of the complete ensemble of feasible foldings. Besides, several ways have been presented for deriving predictions from the generated statistically representative and reproducible Boltzmann samples, such as calculation of ensemble or cluster centroids [DCL05]. Notably, these sampling approaches yield the same high overall complexities (cubic time and quadratic storage in the worst-case) as the standard dynamic programming approaches. Furthermore, statistical sampling methods as implemented in the `Sfold` software [DCL04] are purely physics-based, employing a comprehensive set of energy parameters. Corresponding methods implementing a purely probabilistic model have not been studied so far. In fact, stochastic context-free grammars of comparable complexity have so far not even been applied for RNA folding in general.

Finally, it should be mentioned that over the past years, motivated by new discoveries in the RNA domain and by the need to efficiently analyze the increasing amount of accumulated genome-wide data, some particular algorithmic approaches were applied for accelerating RNA folding, such as local prediction [LMM+12] or sparsification [DB12]. Briefly, sparsification uses the observation that the resulting dynamic programming matrices are sparse, which can lead to a significant reduction of time and space complexity. In fact, several practical speedups to current state-of-the-art RNA folding methods could recently be reached by applying this technique. Most importantly, the approach of [WZZU07] manages to speed up the standard prediction algorithms without sacrificing the optimality of the results, yielding an expected time complexity of $\mathcal{O}(n^2 \cdot \psi(n))$, where $\psi(n)$ is shown to be constant *on average* under standard polymer folding models. In [BTZZU11], it is shown how to reduce those average-case time and space complexities in the sparse case. Anyway, to date basically all computational approaches retain the cubic worst-case time complexity. A notable exception is the practical technique as suggested in [FG10a], which at least allows to obtain a slightly subcubic worst-case runtime. Nevertheless, there currently exists no competitive prediction method that indeed requires only quadratic time *in the worst-case*.

The main objectives of the present thesis are to address exactly the previously described issues. This means we want to design elaborate and very powerful stochastic context-free grammars for modeling secondary structure and apply them to the problem of RNA folding. However,

according to the previous discussion, we will not compute the most probable folding according to the induced probabilistic model as prediction, but we will devise a corresponding statistical sampling algorithm for generating sets of candidate foldings from which predictions can easily be derived. Principally, by utilizing the concept of sampling, we can hope for a higher structural variability in the set of proposed foldings, which might yield better accuracies of selected predictions. In addition to this, we want to make use of another algorithmic technique known from theoretical computer science, namely approximation, in an attempt to reduce the worst-case time complexity of our statistical sampling approach, but without sacrificing much of the quality of the derived results.

In order to evaluate our grammar designs as well as to validate the applicability of the devised algorithms, we want to perform fundamental empirical studies based on different sets of structural RNA data. Notably, these studies are not only interesting from the mere theoretical point of view, but are also of great practical concern. Actually, they are also highly motivated by the fact that especially biologists are consistently asking for more efficient computational prediction methods, where a (moderate) loss of accuracy would willingly be tolerated in favor of saving a significant amount of computation time. This, besides other, is due to the fact that the prediction results obtained by a particular tool generally need to be verified by human expertise and where necessary by additional information, like for instance alignments of homologous structures. Or, as in the case may be, the correct solution needs to be identified from the predicted set of suboptimal solutions or sampled structures. In this context, it is indeed obvious why they at rather high costs aim to reduce waiting times for the results to be considered.

1.2. Overview

The present thesis deals with the two basic concepts of sampling and approximation in connection with the development of efficient algorithms for predicting the secondary structure of RNA molecules. We here follow the probabilistic approach towards RNA folding, where the class of all RNA secondary structures is modeled by an appropriate context-free grammar.

Note that the body of this work is given by Chapters 2 to 10, which can basically be divided into two parts. The first part that consists of Chapters 2 and 3 provides some useful background information, introduces the needed basic definitions and prior results, and effectively summarizes the current state of research related to the area of this work. The second part, which includes Chapters 4 to 10, contains the results of this thesis. It starts with Chapter 4, which outlines the research plan that will be pursued throughout Chapters 5 to 9. In the following, we give a short overview over this plan and explicitly note in what form the respective scientific contributions were already published.

First, in Chapter 5, we present and study an efficient sequence-independent method for generating random samples of RNA secondary structures of a given fixed size n according to a (non-uniform) distribution derived from structural information obtained from trusted RNA secondary structure data. Independent on the considered set of RNA data, the method can be implemented to require only $\mathcal{O}(n \cdot \log(n))$ time for sampling a single structure of length n. In principle, this method builds on a weighted unranking algorithm constructed on the basis of a rather complex stochastic context-free grammar and completely abstracts from the base sequence. The content in this chapter was published as [NSW11].

Chapter 6 introduces and evaluates another sophisticated stochastic context-free grammar with probabilistic parameters basically corresponding to the free energy parameters applied in modern physics-based RNA structure prediction tools. This grammar actually serves as basis for a probabilistic statistical sampling algorithm for generating secondary structures conform with a given input sequence, where just like other state-of-the-art prediction methods, the

algorithm requires cubic time and quadratic space in the worst-case. Moreover, different ways for deriving meaningful structure predictions from generated sample sets are presented which do actually not increase the time complexity in the worst-case. The content of this chapter was published as [NS11b].

In Chapter 7, we describe how to modify the statistical sampling method devised in Chapter 6 to deal with length-dependent probabilistic parameters, that is with grammar parameters that depend on the lengths of generated substructures (see [WN11]). Note that this is actually possible without significant losses in performance. Additionally, this chapter includes a comparative study on how such more specialized parameters affect the reliability of the induced probabilistic model, as well as the accuracy of sampled secondary structures and predictions. This material has not been published so far; it is however available as [SN12b].

The aim of Chapter 8 is to prove or disprove the hypothesis that the concept of approximation could be applied for improving the worst-case complexity of statistical RNA secondary structure sampling. Therefore, we perform a comprehensive empirical study on the influence of different kinds and levels of disturbances in the sampling probabilities to the quality of generated sample sets. Throughout these examinations, some useful ideas for developing a corresponding time-reduced sampling method are presented and potential problems are discussed. Only a small percentage of this material, namely the very first intuitive results, were/will be published in [NS12, NS]. The complete content in this chapter was recently published as [SN12a].

Chapter 9 formally describes an innovative way for reaching a worst-case speedup of probabilistic statistical sampling by a linear factor, making use of the concept of approximation. In fact, several different preprocessing variants are presented that basically differ in the number of approximated values. Note that these methods are only heuristics rather than proper approximation algorithms, as they are not evidentially relative error bounded (no approximation ratio has been defined or proven). Anyway, this chapter also introduces a novel (heuristic) sampling strategy that was actually designed to fit especially well with the resulting "noisy" ensemble distributions. However, for the comprehensive evaluations and validations of our heuristic methods, the common strategy used in the preceding chapters is also considered. The content of this chapter was partially published in the conference paper [NS12] and will be published in the extended journal version [NS], where in both cases only the first intuitive results are included.

The thesis closes with Chapter 10, where we summarize the results of the previous chapters and point out some open questions and possible directions of future work.

Part I

Preliminaries

Chapter 2

The Structure of RNA

In this chapter, we want to provide some basic information that we consider useful for understanding the investigations that will be carried out in this work. We start with an introduction of RNA in Section 2.1. In Section 2.2, we will give all the needed definitions concerning RNA secondary structures that we will rely on in the sequel. Note that this chapter was written especially for those who are not familiar with this topic and may be over-read by others.

2.1. Ribonucleic Acids

In order to provide an all-embracing introduction to the principal topic of this thesis, we first want to present some background information on the molecular structure of RNA in general. Furthermore, we give a brief overview of the variety of different types of (non-coding) RNAs and their main functions in cellular processes. Note that we only provide some rather basic information; more detailed information can be found, for example, in [AJL+02] or [CB00]. Nevertheless, we additionally give a short summary on currently existing databases of diverse types of RNA molecules, including especially the ones that we will make use of in order to perform our examinations and evaluations in the subsequent chapters.

2.1.1. Molecular Structure

Ribonucleic acid (RNA) is a single-stranded *nucleotide polymer* (also called *oligonucleotide* or *polynucleotide*). Polymers are *marcromolecules* composed of a number of equal or similar smaller molecules, called *monomers*.

2.1.1.1. Nucleotides

The basis structural units (monomers) of RNA are *nucleotides*. In RNA, each nucleotide is a molecule consisting of a phosphate group, a sugar group (ribose) and one of the four bases adenine (a), cytosine (c), guanine (g) and uracil (u). The chemical structure of these four bases is illustrated in Figure 2.1.

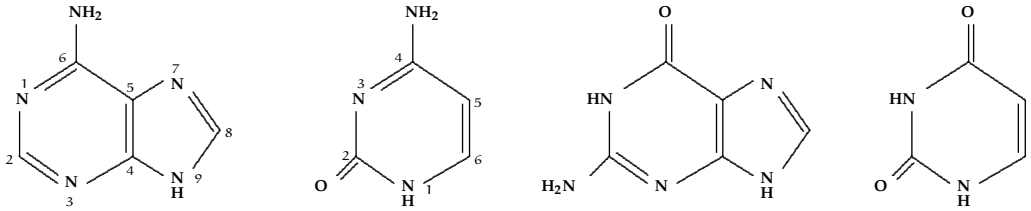


Figure 2.1.: **Chemical structures of the four different RNA bases.** Shown bases are adenine (a), cytosine (c), guanine (g) and uracil (u), from left to right. Figure taken from [Neb04b], with permission of the author.

Ribose is a 5-carbon sugar (pentose) existing in cyclic form, which is called the *ribose ring*. The five carbon atoms of the ribose ring are numbered in clockwise order (or from right to left), and the i^{th} carbon is called the i' carbon of the ribose ring. In each nucleotide, the base is bound to the $1'$ carbon of the ribose ring, as illustrated in Figure 2.2.

2.1.1.2. Single-Strands

An RNA single-strand is formed by linking together the nucleotide units. More precisely, the linear structure of the RNA molecule is formed by creating *phosphodiester bonds*. In such bonds, the phosphate group (at the $5'$ carbon of the ribose ring) of the first nucleotide is attached to the hydroxyl group (at the $3'$ carbon of the ribose ring) of the next nucleotide, as illustrated in Figure 2.3.

Consequently, in RNA strands, there is always a phosphate group at the $5'$ end of the first nucleotide and a hydroxyl group at the $3'$ end of the last nucleotide in the chain and hence this chain could be extended in both directions. In nature, RNA strands are always extended

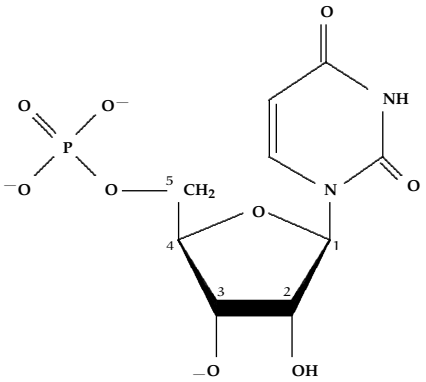


Figure 2.2.: **Chemical structure of one particular nucleotide.** The displayed nucleotide consists of a phosphate group, a ribose ring and the base uracil (u).

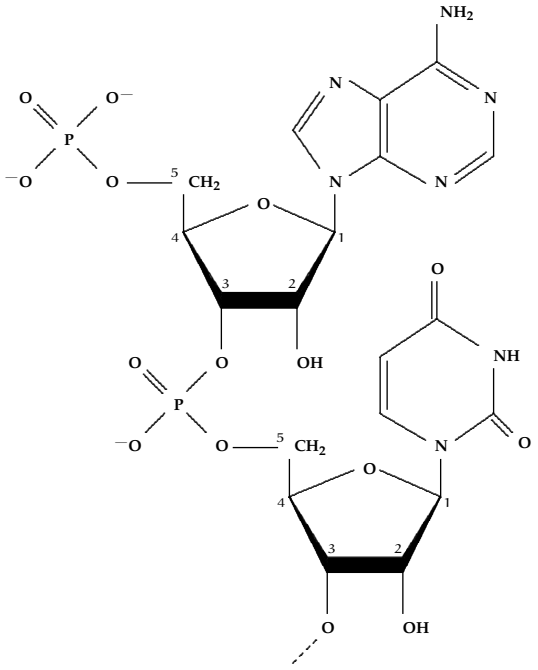


Figure 2.3.: **Two nucleotides chained by a link between their phosphate groups.** Figure taken from [Neb04b], with permission of the author.

at the 3′ end, which implies that they grow in the $5' \to 3'$ direction. Details can be found, for example, in [CB00].

2.1.1.3. Primary Structure

The specific sequence of bases along the RNA chain is called the *primary structure* of the molecule. The primary structure of an RNA molecule is essentially one-dimensional and is usually modeled as a string over the alphabet $\Sigma_r := \{a, c, g, u\}$[1]. That is, it is represented as a sequence of letters $r_1 r_2 \dots r_n$, where r_i is either a, c, g or u. By convention, strings representing the primary structure of RNA molecules are written in the $5' \to 3'$ direction, which means that they are written with the 5′ end at the left to the 3′ end at the right.

Note that the primary structure of a particular known RNA molecule can generally be found by utilizing GenBank [BKML+11], a comprehensive public database of nucleotide sequences which contains an annotated collection of all publicly available RNA sequences (supporting

[1]Each symbol in this alphabet Σ_r corresponds to one of the four bases adenine, cytosine, guanine and uracil.

both bibliographic and biological annotation). In fact, any sequence is given a corresponding accession number under which it can easily be found.

Example 2.1.1 *The primary structure of the well-known* Escherichia coli $tRNA^{Ala}$ *(or* E.coli $tRNA^{Ala}$*) molecule is given by the following sequence:*
ggggcuauagcucagcugggagagcgcuugcauggcaugcaagaggucagcgguucgaucccgcuuagcuccacca.
Its GenBank accession number is X66515.

2.1.1.4. Tertiary Structure

In vivo, single-stranded RNA chains bend and twine about themselves. The reason for this behavior is that, in addition to the phosphodiester bonds between neighbored bases in the RNA chain, two bases that are not neighbored may form other (weaker) chemical bonds, called *hydrogen bonds*. More precisely, the complementary bases a and u resp. c and g form stable base pairs with each other by creating hydrogen bonds. These base pairs are called *Watson-Crick pairs*, and they are illustrated in Figure 2.4.

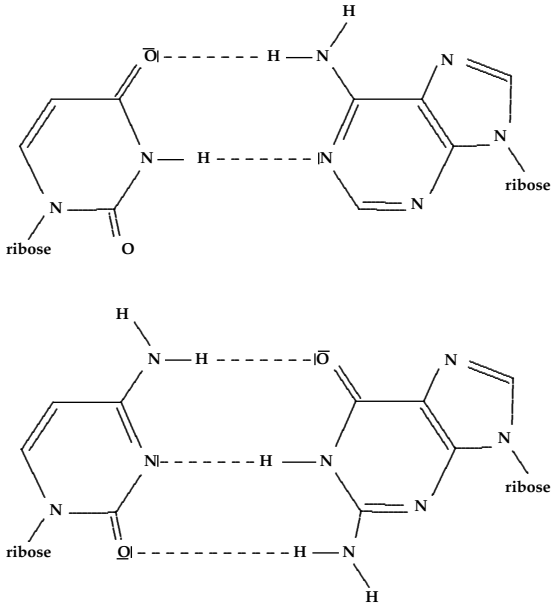
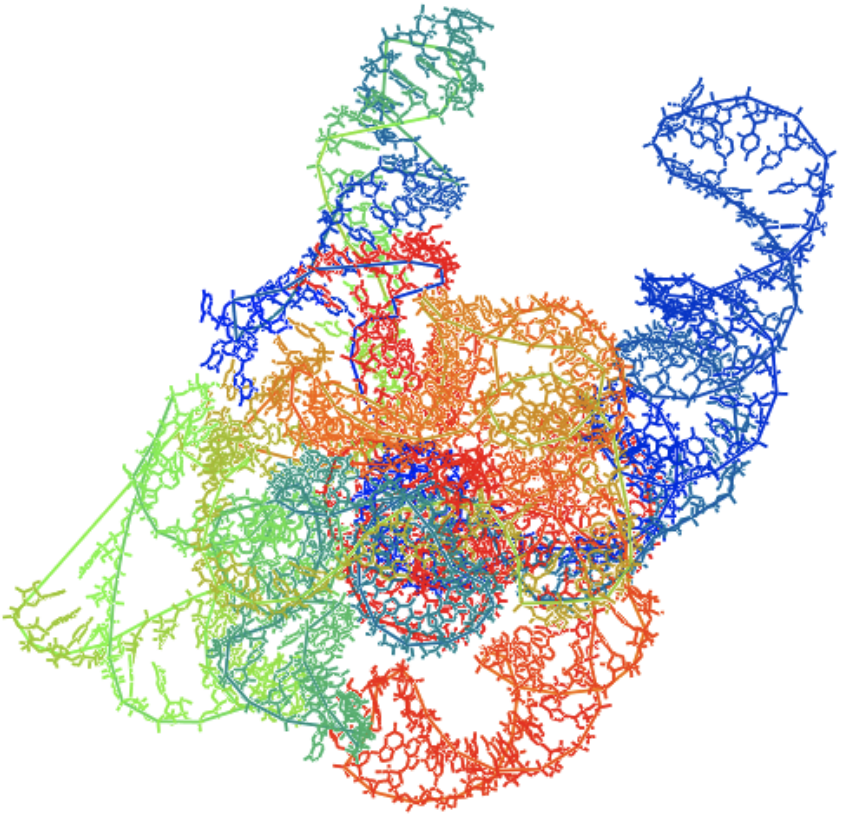


Figure 2.4.: **Watson-Crick base pairings.** The hydrogen bonds between the complementary bases a and u (top) resp. c and g (bottom). Figure taken from [Neb04b], with permission of the author.

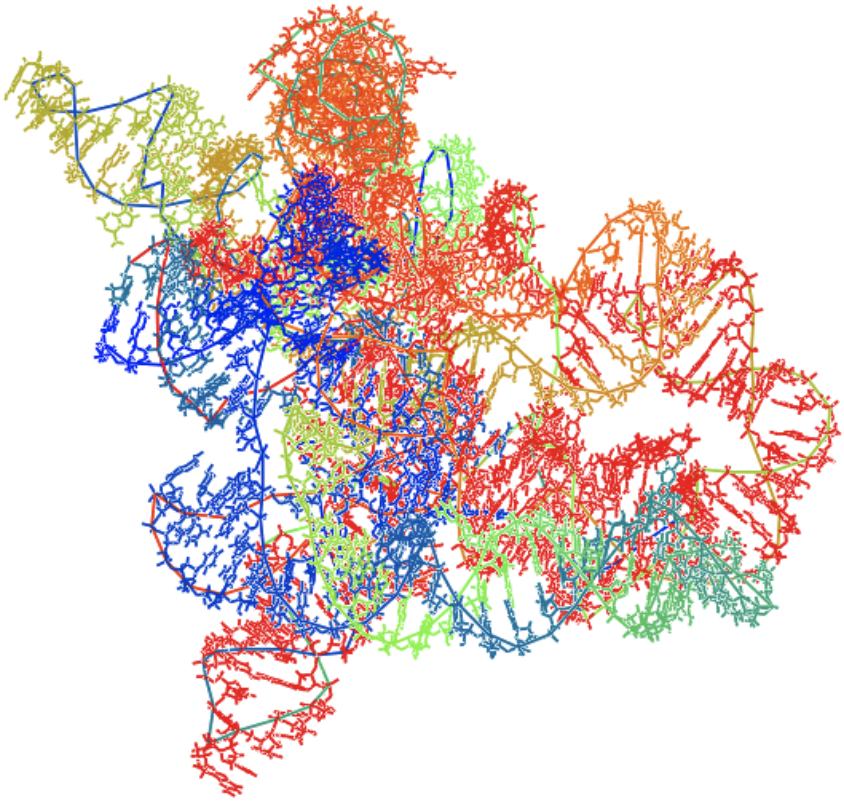
In addition to these stable Watson-Crick base pairs, there may occur weaker base pairs, called GU *wobble pairs*, which are formed by the non-complementary bases g and u. All these pairs (Watson-Crick and GU wobble pairs) will be called *canonical* base pairs in the sequel, as they are most common. Other pairs, which will be named *non-canonical* base pairs in the sequel, may also occur, but they are not as stable as the canonical ones. For more information on geometric nomenclature and classification of RNA base pairs, we refer to [LW01].

Since base pairs may be formed between any two non-neighbored nucleotides[2], the linear RNA chain is folded into a three-dimensional (3D) conformation, called the *tertiary structure* of the RNA molecule, which determines the biochemical activity of the molecule. Visualizations of two exemplary RNA tertiary structures are illustrated in Figure 2.5.

[2]In nature, base pairs between two nucleotides having distance less than three in the single-stranded RNA chain are impossible and do not form.



(a) *E.coli RNase P RNA*, listed as: An updated set of models, all-atom, from [CNHP98].



(b) *Escherichia coli RNase P RNA:pre-tRNA*, listed as: A completely reworked model, from [MJW98].

Figure 2.5.: **3D structures of RNAs coming from biological data bases.** Figures show visualizations of RNA tertiary structures (with pseudoknots) taken from a biological database, more specifically the Ribonucleic P database [Bro99]. Pictures were created with the molecular visualizer iMol [Rot07].

2.1.1.5. Secondary Structure

Although the function of non-protein coding RNA is often designated by its overall tertiary structure, energetically most of the 3D structure is dominated by the intramolecular base pairings of the molecule [WMJ11]. In fact, substantial tertiary interactions generally contribute only minimally to the stability of the native state [BW97, TB99]. Hence, the intramolecular base pairs provide a convenient and computationally tractable approximation to RNA structure [BTM+06]. Moreover, the experimental determination of the complete 3D structure of an RNA molecule is usually time-consuming and expensive.

Therefore, it has become customary in science to simplify the study of the tertiary structure of an RNA molecule by allowing only intramolecular base pairing interactions, in many cases even only base pairings *in the plane*, so-called *non-crossing* base pairs. That is, only such base pairs are then allowed that are either *nested* or *adjacent* with respect to the other pairings in the formed structure, such that the corresponding folding remains planar. Accordingly, this restriction yields a two-dimensional (2D) conformation, called the *secondary structure* of the molecule. By investigating secondary structures of RNA instead of the corresponding tertiary structures, the focus of attention is hence set only on what base pairs are involved, and not on the 3D conformation of the RNA chain.

Figure 2.6 shows a hand drawn RNA secondary structure made by a biologist. Note that in this drawing, the canonical (Watson-Crick and Wobble GU) pairs have been designated following the convention of [DG94] for annotating RNA secondary structure. That is, by using the symbol $-$ for both $\mathtt{au}$ and $\mathtt{cg}$ pairs and $\bullet$ for $\mathtt{gu}$ pairings, respectively. However, it is obvious that this structure is essentially planar if the red and blue lines are ignored. In fact, any of them represents a so-called *pseudoknot*, that is a number of consecutive base pairs, where any of these pairs is actually *crossing* with respect to the other base pairings in the plane (which are represented by symbols $-$ and $\bullet$); details will follow in Section 2.2.1. Notably, RNA secondary structure has to be understood in a wider sense by allowing pseudoknotted structures, and therefore those are usually considered as belonging to the tertiary structure.

Nevertheless, it should be mentioned that since the first discovery of RNA pseudoknots [PRB85], a large number of different classes of pseudoknots have been recorded and it has been recognized that if present, they are in many cases crucial for the function of the corresponding molecule. However, the vast majority of RNA molecules folds into secondary structures without crossing base pairs, such that their structure is essentially pseudoknot-free. In all other cases, the number of occurrences of crossing base pairs is generally rather low (compared to the number of intramolecular pairings in the plane). For a demonstration of this behavior, see for example Figure 2.6. Consequently, pseudoknots are often ignored in connection with studies related to the secondary structure of RNA in order to reduce problem or model complexity; details will follow in Section 3.4.1.

2.1.2. Types and Functions

Just like DNA and proteins (the two other major macromolecules involved in cellular processes), RNA is essential for all known forms of life. Synthesis of RNA within living cells is usually catalyzed by an enzyme (RNA polymerase), using DNA as a template. This process is known as *transcription* or RNA *replication*, which occurs in the $5' \to 3'$ direction. Notably, RNAs are often modified by enzymes after transcription.

The primary role of RNA is to convert the information stored in DNA into proteins during protein synthesis. Therefore, the three major categories of RNA are needed, which can briefly be described as follows:

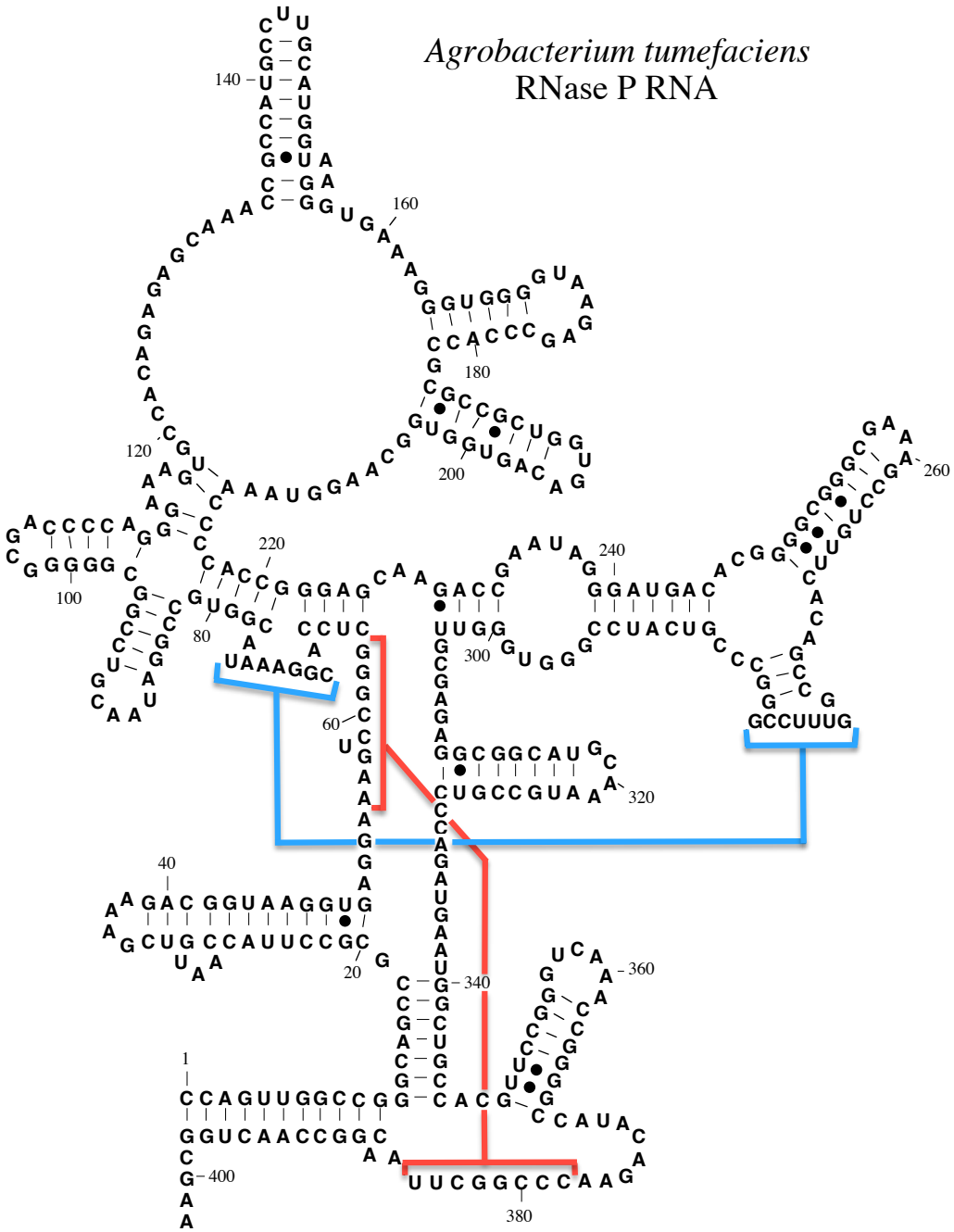


Figure 2.6.: **2D drawing of a pseudoknotted RNA coming from a biological database.** Figure shows an RNA secondary structure with two pseudoknots (highlighted with red and blue colors, respectively) based on the model of [HHW+01]. It has been taken from a biological database, more specifically the Ribonucleic P database [Bro99].

1. Messenger RNA (mRNA) has an information storage function, like DNA, since all cellular organisms use it to carry the genetic information (encoded by the sequence of nucleotides) that directs the synthesis of proteins. In fact, mRNA presents some sort of copy of the DNA base sequence of a gene after processing. Note that a number of viruses also use RNA instead of DNA as their genetic material.

2. Transfer RNA (tRNA) is used in biology to bridge the four-letter genetic code in mRNA with the twenty-letter code of amino acids in proteins. During protein synthesis, tRNA molecules are delivered to the ribosome by proteins, which aid in decoding the mRNA codon sequence. In fact, tRNA is somehow "charged" with an amino acid and used to recognize the code in the mRNA and "translate" it into the amino acid it is carrying. There are specific tRNA molecules for each amino acid, typically consisting of 73 to 93 nucleotides.

3. Ribosomal RNA (rRNA) is the RNA component of the ribosome, the protein manufacturing machinery of all living cells. Like proteins, rRNA has catalytic functions, that is it plays an active role in cells by catalyzing biological reactions. Notably, rRNA molecules provide a mechanism for decoding mRNA into amino acids and interact with tRNAs during translation. In fact, rRNA forms two subunits, the large subunit (LSU) and small subunit (SSU), and mRNA is sandwiched between those subunits.

However, biologically active types of RNAs produced in cells do not only include mRNA, tRNA, and rRNA, but also small nuclear RNA (snRNA), small nucleolar RNA (snoRNA), and other non-coding RNAs. For information on their respective function, see for example, [AJL+02].

For the sake of completeness, note that ribonuclease P (RNase P) – of which two known 3D structures are visualized in Figure 2.5 – is a ribozyme (a RNA that acts as a catalyst in the same way that a protein based enzyme would) that cleaves RNA. More specifically, it can cleave tRNA precursor molecules in buffers. Notably, RNase P is one of two known multiple turnover ribozymes in nature (the other being the ribosome). Furthermore, it has been shown that human nuclear RNase P is required for the normal and efficient transcription of various small non-coding RNA genes, including tRNA and 5S rRNA.

2.1.3. Structural Databases

In order to provide information on nucleotide sequences and conformations (secondary and/or tertiary) of RNA molecules, a number of corresponding databases have been made available to the public, which are specialized in different directions, following different goals.

For example, there exist several comprehensive collections of structural information for particular types of RNA molecules, including tRNAs [SHB+98, SV05, JMH+09], 5S rRNAs [SBEB02], SSU rRNAs [WdPWW02] and LSU rRNAs [WRdP+01]. Other structural RNA databases are specialized to RNase P molecules [Bro99], signal recognition particle RNAs (SRP RNAs) [LCZ91], signal recognition particles (SRPs) [RGK+03, ARL+06] and transfer-messenger RNAs (tmRNAs) [ZGK+03].

Moreover, the comparative RNA web site [CSS+02] specializes in both rRNA and intron RNA molecules and the Rfam database [GJBM+03, GJMM+05] contains a large collection of non-coding RNA families. Additionally, PseudoBase [vBGP+00, vBGP01] contains a number of short RNA fragments that have pseudoknots, and an extension called PseudoBase++ [TLA+08], can be used for easy searching, formatting and visualization of pseudoknots.

Anyway, despite the variety of databases that have been made available to search and download specific classes of RNA secondary structures, to the best of our knowledge, the RNA STRAND database [ABHC08] (which stands for RNA secondary STRucture and statistical ANalysis database) was the first to provide convenient access to a large set of RNA molecules of any type

and organism, along with known RNA secondary structures. In fact, RNA STRAND spanned a more comprehensive range of RNA secondary structures than did previous databases, where information came from other publicly available collections of RNA data. This structural database may thus be used for deriving improved RNA energy models, for evaluating computational prediction methods, and for a better understanding of RNA folding in general, for instance by deriving statistics of naturally occurring structural features, or searching RNA molecules with specific motifs.

Finally, note that there are also several databases that contain 3D structures of RNAs (for instance [BOB+92, MR03, THK+04]). However, they are not as comprehensive as the previously mentioned collections, since the number of known RNA tertiary structures is actually much smaller than the number of solved secondary structures.

2.2. RNA Secondary Structures

The purpose of this section is to introduce all the needed formal definitions and notations concerning RNA secondary structures that will be used in the sequel, as well as to provide information on some of the most common representations for secondary structures of RNA molecules.

2.2.1. Formal Definitions

First, we want to present the basic formal definitions and notations for all terms related to RNA sequences and their secondary structures, including characterizations of common types of substructures and structural motifs. Note that in this thesis, we constantly use r to denote RNA sequences and s to denote secondary structures.

2.2.1.1. Overall Structure

The bases of an RNA sequence are traditionally numbered from 1 (called the $5'$ terminus) to n (called the $3'$ terminus), such that any RNA sequence r of length n can be written as $r_1 \dots r_n$. In the sequel, given an RNA molecule consisting of n nucleotides, we denote the corresponding sequence fragment from position i to position j, $1 \leqslant i, j \leqslant n$, by $R_{i,j} := r_i r_{i+1} \dots r_{j-1} r_j$.

Secondary structures as combinatorial objects are typically defined independent on their corresponding primary structures. We start with the following (sequence-independent) definition of an RNA secondary structure of size n, parameterized by the variable $\min_{HL} \in \mathbb{N}$, which will be explained later on:

Definition 2.2.1 ([Zuk86]) *A* secondary structure s *of size* n *is a finite set (possibly empty) of ordered pairs of positive integers, written as* $i.j$*, satisfying the following:*

1. *If* $i.j$ *is in* s*, then* $1 \leqslant i, j \leqslant n$ *and* $|j - i - 1| \geqslant \min_{HL} \geqslant 1$.
2. *If* $i.j$ *and* $i'.j'$ *are in* s *(we can assume that* $i \leqslant i'$*), then either*
   a) $i = i'$ *and* $j = j'$ *(that is,* $i.j$ *and* $i'.j'$ *are identical),*
   b) $i < i' < j' < j$ *(that is,* $i.j$ *includes* $i'.j' \mathrel{\widehat{=}} i.j$ *and* $i'.j'$ *are nested) or*
   c) $i < j < i' < j'$ *(that is,* $i.j$ *precedes* $i'.j' \mathrel{\widehat{=}} i.j$ *and* $i'.j'$ *are adjacent).*

If $i.j$ *is in* s*,* i *and* j *are said to form a* base pair.

Note that the original definition from [Zuk86] does not consider the structural parameter $\min_{HL} \geqslant 1$. In fact, that definition invariably assumes $\min_{HL} = 1$. However, in the style of [ZMT99], Definition 2.2.1 can be rewritten as follows:

Definition 2.2.2 *A* secondary structure s *of size* n *is a finite set (possibly empty) of* base pairs*. A base pair between* i *and* j*,* $1 \leqslant i < j \leqslant n$*, is denoted by* $i.j$*. A few constraints are imposed:*

(1) Two base pairs, $i.j$ *and* $i'.j' \in s$ *are either identical, or else* $i \neq i'$ *and* $j \neq j'$*. Thus base triplets are deliberately excluded from the definition of secondary structure.*

(2) Pseudoknots are prohibited. That is, if $i.j$ *and* $i'.j' \in s$*, then, assuming* $i < i'$*, either* $i < i' < j' < j$ *or* $i < j < i' < j'$*.*

(3) Sharp U-turns are prohibited. A U-turn, called hairpin loop, must contain at least $\min_{HL}$ *bases. That is, if* $i.j \in s$*, then* $|j - i - 1| \geqslant \min_{HL}$*.*

According to constraint (1), each $i \in [1; n]$ occurs either in exactly one pair or in no pairs, and i is described as *paired* or *unpaired*, accordingly. Pseudoknots, formed by two base pairs $i.j$ and $i'.j'$ satisfying $i < i' < j < j'$, are not permitted in secondary structures according to Definition 2.2.2, due to constraint (2). In fact, in this thesis they are always considered as belonging to the tertiary structure, meaning for the understanding of this thesis, no further knowledge on pseudoknotted structures is required. For this reason, we simply refer to [PB89, AvdBvBP90, GW90, DPD92, Ple94] for more fundamental information on RNA pseudoknots.

Anyway, it should be noted that constraints (1) and (2) of Definition 2.2.2 limit the number of possible foldings of a given RNA molecule in a very significant way. For values of $\min_{HL} \in \{1, 2\}$, however, this definition of secondary structures still allows many biologically impossible structures according to constraint (3). Thus, to limit the number of possible secondary structures of size n even more and to exclude many biologically impossible structures for any size n, the stereochemical constraint that i and j cannot base pair if $|j - i - 1| < 3$ has to be included into the definition of a secondary structure of size n. Actually, $\min_{HL} = 3$ is a reasonable choice that is mostly considered in literature.

Furthermore, due to the abstraction from the actual primary structure, Definition 2.2.2 does not exclusively include secondary structures conform with a particular RNA sequence, since any two bases are principally allowed to pair. Hence, in connection with a particular sequence, the definition of secondary structures might generally be restricted to include only biochemically admissible structures by adding a fourth condition allowing only canonical base pairs to the three conditions already formulated in Definition 2.2.2.

Example 2.2.1 *The native secondary structure of the* E.coli $tRNA^{Ala}$ *molecule given in Example 2.1.1 can easily be described by the following set of base pairs:*

$$\{1.72, 2.71, 3.70, 4.69, 5.68, 6.67, 7.66,$$
$$10.25, 11.24, 12.23, 13.22,$$
$$27.43, 28.42, 29.41, 30.40, 31.39,$$
$$49.65, 50.64, 51.63, 52.62, 53.61\}.$$

Note that this secondary structure is actually conform with the sequence, that is it exclusively contains canonical base pairings.

In summary, a secondary structure s is basically a collection of base pairs subject to a few simple rules. Following the convention that RNA sequences are written in the $5' \to 3'$ direction (or else, in the usual reading order from left to right), for any base pair $i.j$ in s, we call i the $5'$ base and j the $3'$ base of this pair.

2.2.1.2. Paired and Unpaired Regions

To distinguish between paired and unpaired bases, or else, double-stranded and single-stranded regions in RNA secondary structures, we can easily make use of the following definition:

Definition 2.2.3 ([ZMT99]) *A group of two or more consecutive*[3] *base pairs is called a* helix *(alternatively,* helical region*,* ladder *or sometimes also* stem*). The first and last are the closing base pairs of the helix. They may be written as* $i.j$ *and* $i'.j'$*, where* $i < i' < j' < j$*. Then* $i.j$ *is called the* external closing base pair *and* $i'.j'$ *is called the* internal closing base pair*.*

In this context, it should be mentioned that in addition to $\min_{HL}$, some authors have considered a corresponding structural parameter $\min_{hel}$ denoting the minimum number of consecutive base pairs that must be contained in any helical region, such that helices of lengths less then $\min_{hel}$ are prohibited. Actually, $\min_{hel} \in \{1, 2\}$ is to date most often used in literature, where $\min_{hel} = 1$ is conform with the common definition of RNA secondary structures (see above) and much easier to handle in many applications. However, the consideration of $\min_{hel} = 2$ is motivated by the biochemical aspect that helices consisting of only one single pairing (so-called *isolated* base pairs) may not be observed in nature, due to low stability.

Nevertheless, any secondary structure s can obviously be decomposed into single-stranded regions and helices. Due to this fact, secondary structures are often alternatively described by a corresponding set of triplets defining the distinct helical regions, and if needed, an additional set of tuples defining the distinct single-strands. Formally, any secondary structure s of size n can be uniquely defined by a set

$$\begin{aligned} \mathtt{helices} := \big\{(i,j,k)\,|&1 \leqslant i < j \leqslant n \text{ and } k \geqslant \min_{hel} \text{ and} \\ &i.j, (i+1).(j-1), \ldots, (i+(k-1)).(j-(k-1)) \text{ are consecutive base pairs}\big\}, \end{aligned}$$

accompanied by an additional (in some cases useful) set

$$\begin{aligned} \mathtt{unpRegs} := \big\{(i,l)\,|&1 \leqslant i \leqslant n \text{ and } l \geqslant 1 \text{ and} \\ &i, (i+1), \ldots, (i+(l-1)) \text{ are subsequent unpaired bases}\big\}, \end{aligned}$$

such that for any feasible secondary structure s of n, we obviously have:

$$\begin{aligned} \{1,\ldots,n\} \;=\; &\left(\bigcup_{(i,j,l)\in \mathtt{helices}} \{i,(i+1),\ldots,(i+(l-1))\} \cup \{j,(j-1),\ldots,(j-(l-1))\} \right) \cup \\ &\left(\bigcup_{(i,l)\in \mathtt{unpRegs}} \{i,(i+1),\ldots,(i+(l-1))\} \right). \end{aligned}$$

Example 2.2.2 *The secondary structure of* E.coli $tRNA^{Ala}$ *as presented in Example 2.2.1 can alternatively be formally described by the following set defining the corresponding helices:*

$$\mathtt{helices} = \big\{(1,72,7), (10,25,4), (27,43,5), (49,65,5)\big\}.$$

The corresponding single-stranded regions are thus given by:

$$\mathtt{unpRegs} = \big\{(8,2), (14,8), (26,1), (32,7), (44,5), (54,7), (73,4)\big\}.$$

[3] A group of $k \geqslant 1$ consecutive base pairs means k base pairs $(i+1).(j-1), \ldots, (i+k).(j-k)$ such that neither the two bases $(i+k+1)$ and $(j-k-1)$ nor the two bases i and j (if existing) form together a base pair.

Note that this representation of RNA secondary structures requires less storage space than the typically large sets of all formed base pairs and is hence commonly used for practical applications. For instance in order to minimize the output of comprehensive sets of data (like for example the complete sample sets computed by the `Sfold` tool [DCL04] for predicting and studying RNA secondary structure). Anyway, in the sequel we will (except for in a few pseudocode fragments) consider secondary structures according to Definition 2.2.2.

2.2.1.3. Canonical Loop Types

For our investigations, we additionally need to distinguish between base pairs and single-stranded regions located in different parts of the overall structure. Therefore, we first introduce the following definition:

Definition 2.2.4 ([Zuk86]) *Any subset of a secondary structure s is also a secondary structure, and is called a* substructure. *The substructure $S_{i,j}$ for $1 \leqslant i < j \leqslant n$ is defined as*

$$S_{i,j} = \{i'.j' \in s : i \leqslant i' < j' \leqslant j\}.$$

We can decompose any given secondary structure s in a unique way into a number of substructures such that each nucleotide is contained in exactly one such substructure:

Definition 2.2.5 (k-loop decomposition [ZS84, Zuk86]) *If $i.j$ is a base pair in the secondary structure s and if $i < k < j$, we say that k is* accessible *from $i.j$ if there is no $i'.j'$ in s such that $i < i' < k < j' < j$. Similarly, if $k.l$ is also in s, we say that the base pair $k.l$ is* accessible *if both k and l are accessible. The set of $(k-1)$ base pairs and k' unpaired bases accessible from $i.j$ is called the* k-loop *(or* k-cycle*) closed by $i.j$. The (possibly empty) set of base pairs in a k-loop constitute the* interior *base pairs of the k-loop. The closing base pair is called the* exterior *base pair. k' is called the* size *of the k-loop. The collection of $(k-1)$ base pairs and k' unpaired bases which are accessible from no base pair (the* exterior *or* free *base pairs and bases) is called the* null k-loop *or* exterior loop*. It is easy to see that any secondary structure s decomposes the sequence $1, 2, \ldots, n$ uniquely into k-loops $s_0, s_1, s_2, \ldots, s_m$, where s_0 is the null k-loop and $m > 0$ iff s is nonempty* [4].
Biochemists have developed their own nomenclature for k-loops. The various cases and subcases are given as follows:

1. $k = 1$*: A 1-loop is called a* hairpin loop.
2. $k = 2$*: Let $i'.j'$ be the base pair accessible from $i.j$. Then the 2-loop is called*
   a) *a* stacked pair, *if $i' - i = 1$ and $j - j' = 1$,*
   b) *a* bulge (loop) *if $i' - i > 1$ or $j - j' > 1$, but not both, and*
   c) *an* interior loop[5] *if $i' - i > 1$ and $j - j' > 1$.*
3. $k \geqslant 3$*: These k-loops are called* multi-branched loops, multiple loops *or simply* multiloops.

In [ZMT99], the authors note that if we imagine adding a 0^{th} and an $(n+1)^{st}$ base to the RNA sequence, and a base pair $0.(n+1)$ to the corresponding secondary structure, then the exterior loop becomes the loop closed by this imaginary base pair. They call this the *universal closing base pair* of an RNA structure. Furthermore, they used the following notations, which we will also use in this work:

[4] Note that this decomposition was first introduced in [SKMC83] and was later redefined. In the original definition, the closing pair belongs to the k-loop, but in the redefinition given here, the closing base pair is no longer contained in the k-loop.

[5] In the sequel, such an interior loop will sometimes be called $(i' - i - 1) \times (j - j' - 1)$ interior loop to specify the number of unpaired bases between the paired bases i and i', as well as j and j', respectively.

- The loop closed by $i.j$ will be denoted by $\text{Loop}(i.j)$.
- The exterior loop will be denoted by Loop_e.
- If s is a secondary structure, then s' denotes the same secondary structure with the addition of the universal closing base pair.
- The terms $l_s(\text{Loop})$ and $l_d(\text{Loop})$ denote the number of single-stranded bases and base pairs accessible from the closing base pair of loop Loop, respectively.
- The size of a 1-loop or a 2-loop is defined as $l_s(\text{Loop})$.

Hence, if $\text{Loop}(i.j)$ is an interior loop with interior base pair $i'.j'$ which is accessible from the exterior base pair $i.j$ of the loop, then its size $l_s(\text{Loop})$ can be written as

$$l_s(\text{Loop}) = l_s^1(\text{Loop}) + l_s^2(\text{Loop}),$$

where $l_s^1(\text{Loop}) = i'-i-1$ and $l_s^2(\text{Loop}) = j-j'-1$. Due to this fact, there are some special types of interior loops, depending on the combination of the two sizes $l_s^1(\text{Loop})$ and $l_s^2(\text{Loop})$:

Definition 2.2.6 ([ZMT99]) *Let* $\text{Loop}(i.j)$ *be an interior loop of size* $l_s(\text{Loop}) = l_s^1(\text{Loop})+l_s^2(\text{Loop})$.

- *If* $l_s^1(\text{Loop}) = l_s^2(\text{Loop})$, *the loop is called* symmetric*; otherwise, it is* asymmetric, *or* lopsided.
- *The asymmetry of the interior loop* Loop, $a(\text{Loop})$ *is defined by:*

$$a(\text{Loop}) = |l_s^1(\text{Loop}) - l_s^2(\text{Loop})|.$$

- *If* $l_s^1(\text{Loop}) = 1$ *and* $l_s^2(\text{Loop}) = n$ *or* $l_s^1(\text{Loop}) = n$ *and* $l_s^2(\text{Loop}) = 1$, $n > 2$, *then the interior loop* Loop *is called a* "Grossly Asymmetric Interior Loop" (GAIL).

Note that in literature, (single-strands in) bulge and interior loops are sometimes described as interruptions of helical regions. This loose nomenclature formally contradicts Definition 2.2.3 (no interruptions by unpaired nucleotides are allowed), but somehow unites the different types of 2-loops according to Definition 2.2.5, which can simplify descriptions and formalizations of RNA secondary structures. Therefore, we will in certain cases also make use of this terminology.

Finally, it remains to mention that by including pseudoknots in RNA secondary structures, the k-loop decomposition breaks down.

2.2.2. Representations

As we already know, by definition, any secondary structure is a collection of base pairs subject to a few simple rules and could hence be represented by simply listing this set of base pairs (or alternatively, the corresponding set of resulting helices). However, such a formal description of an RNA secondary structure is not really intuitive and for this reason, several alternative representations have been devised over past decades that are commonly used in literature. In this section, we want to introduce two of the most popular and straightforward ways of representing RNA secondary structures and give a short overview on some other useful alternatives.

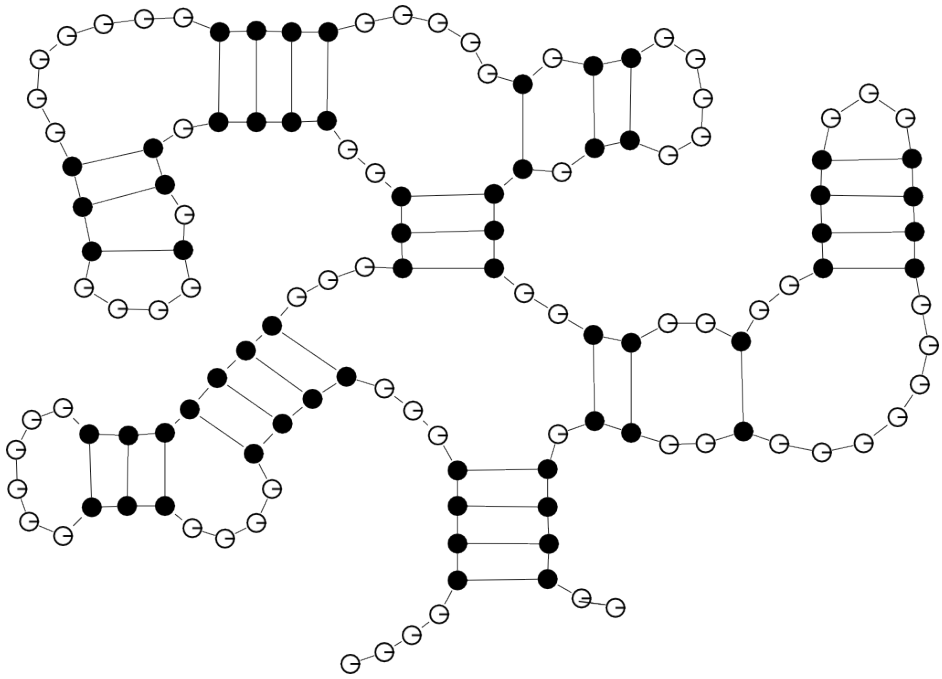


Figure 2.7.: **Planar graph representation of an RNA secondary structure.** Paired bases are colored black, whereas unpaired bases are colored white, respectively.

2.2.2.1. Planar Graph Representation

Since RNA secondary structures are essentially two-dimensional, they can readily be modeled as planar graphs. A formal definition is given as follows:

Definition 2.2.7 ([Wat78]) *A* secondary structure *of size* n *is a loop free graph on the set of* n *labeled points* $\{1, 2, \ldots, n\}$ *such that the adjacency matrix* $A = (a_{i,j})$ *(which is defined in the usual way by* $a_{i,j} = 1$ *if* i *and* j *are adjacent, and* $a_{i,j} = 0$ *otherwise, with* $a_{i,i} = 0$*) has the following three properties:*

(1) $a_{i,i+1} = 1$ *for* $1 \leqslant i \leqslant n-1$ *(≙ phosphodiester bonds).*

(2) For each fixed i, $1 \leqslant i \leqslant n$, *there is at most one* $a_{i,j} = 1$ *where* $j \neq i \pm 1$ *(≙ hydrogen bonds).*

(3) If $a_{i,j} = a_{k,l} = 1$, *where* $i < k < j$, *then* $i \leqslant l \leqslant j$.

Note that constraint (3) of Definition 2.2.7 ensures that these graph representations remain planar. In fact, according to this constraint, pseudoknots are prohibited. Furthermore, it should be clear that this planar graph model also abstracts from the actual RNA sequence, as only the number of base pairs and their positions are taken into account. An example for a secondary structure in planar graph representation is shown in Figure 2.7.

It is worth mentioning that representations of RNA secondary structures as planar graphs are used universally, as they pictorially represent the structure, such that the different substructures and loops can immediately be determined.

Example 2.2.3 *The colored planar graph shown in Figure 2.8 indeed perfectly illustrates the different structural components of RNA secondary structures.*
Obviously, the exterior loop of this secondary structure is composed of tree adjacent structural components: a single-strand (light pink), a helix and another single-strand (also light pink). The helix consists of a ladder (purple) and a multiloop, where this multiloop contains four unpaired regions (all colored white), as well as three helices radiating out from this loop:

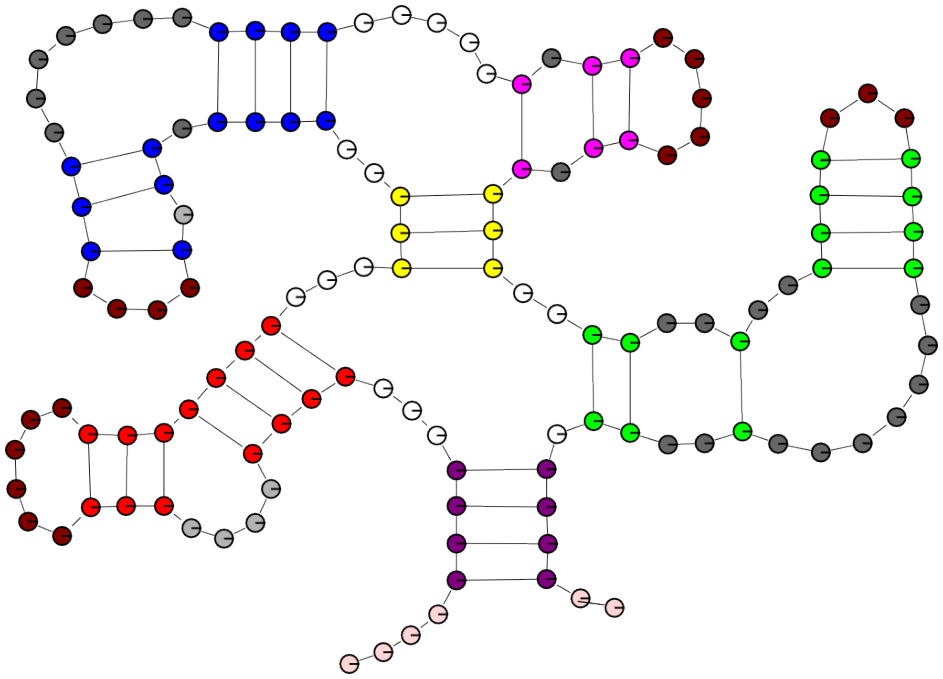


Figure 2.8.: **Planar graph representations of different substructures.** Figure shows a colored version of the secondary structure given in Figure 2.7. For each helical region, paired bases are displayed with matching colors. Additionally, hairpin loops are colored brown, single bulges interrupting ladders are colored light gray, interior loops are colored dark gray, and unpaired regions in multiloops and exterior loops are colored white and light pink, respectively.

- *The first helix of this multiloop can be decomposed into a hairpin loop (brown) and a ladder (red) which is interrupted by a single bulge (light gray).*
- *The third helix of this multiloop consists of a hairpin loop (brown) and a ladder (green) interrupted twice by interior loops (dark gray).*
- *Furthermore, the second helix radiating out from this multiloop can be decomposed into two different structural components, namely a ladder (yellow) and another multiloop. This other multiloop is formed by a single-strand (white) and two helices which are linked by another strand (also white).*
  The first helix of this multiloop consists of a hairpin loop (brown) and a ladder (blue) which is interrupted by an interior loop (dark gray) and a single bulge (light gray). Its second helix is composed of a hairpin loop (brown) and a ladder (pink) interrupted by two unpaired regions (dark gray) which together form an interior loop.

2.2.2.2. Dot-Bracket Representation

Besides the popular planar graph representation, many other ways of formalizing RNA folding have been described in literature. One well-established example is the so-called *dot-bracket representation*, where a secondary structure is modeled as a string over the alphabet $\Sigma_s := \{(,), \circ\}$, with a dot $\circ$ resp. a pair of corresponding brackets ( ) representing an unpaired nucleotide resp. two paired bases in the molecule. A formal definition of these dot-bracket representations of RNA secondary structures is given as follows[6]:

[6]Note that this definition actually describes a way for modeling RNA secondary structures by a corresponding *context-free language*. Details will follow in Section 3.3.4.

Definition 2.2.8 ([VC85]) *For $\Sigma_s := \{(,), \circ\}$ and $w \in \Sigma_s^*$, let $|w|_x$ for $x \in \Sigma_s$ denote the number of occurrences of symbol x in w. Then a word $w \in \Sigma_s^n$ represents a* secondary structure *of length n if w satisfies the following three conditions:*

1. *For every factorization $w = u \cdot v$, $|u|_(\geqslant |u|_)$.*
2. *$|w|_(= |w|_)$.*
3. *w has no factor ( ).*

It remains to mention that words over the alphabet $\{(,)\}$ which satisfy the first two conditions of Definition 2.2.8 are known as *semi-Dyck words*, whereas words over the alphabet Σ_s satisfying these first two conditions are known as *Motzkin words*. The third condition of Definition 2.2.8 ensures that a hairpin loop consists of at least one unpaired nucleotide, thus $\min_{HL} = 1$ is implicitly assumed in this original version.

However, the original version of the planar graph definition of secondary structures as given in Definition 2.2.7 also commits a minimum loop size of one for hairpin loops, and there obviously exists a one-to-one correspondence between the planar graph model for secondary structures and the model of bracketed string representations, as illustrated by the following example:

Example 2.2.4 *The secondary structure shown in Figure 2.8 has the following equivalent dot-bracket representation:*

$$\circ\circ\circ\circ(((\circ\circ\circ(((\circ\circ\circ\circ(((\circ\circ\circ\circ\circ\circ)))))\circ\circ\circ(((\circ\circ(((\circ(\circ(\circ\circ\circ\circ)))\circ\circ\circ\circ\circ\circ\circ)))\circ\circ\circ\circ\circ(\circ((\circ\circ\circ\circ\circ))\circ)))\circ\circ((\circ\circ(\circ\circ(((\circ\circ\circ)))\circ\circ\circ\circ\circ\circ\circ)\circ\circ))\circ))))\circ\circ.$$

It can also be decomposed into subwords corresponding to the basic structural motifs:

$$\overbrace{\circ\circ\circ\circ((((\underbrace{\circ\circ\circ\, hel_1 \,\circ\circ\circ\, hel_2 \,\circ\circ\, hel_3 \,\circ}_{\text{multiloop (of degree 3)}}))))\circ\circ}^{\text{exterior loop}}, \text{ where}$$

$$hel_1 = ((((\overbrace{\circ\circ\circ\circ(((\underbrace{\circ\circ\circ\circ\circ\circ}_{\text{hairpin}})))}^{\text{bulge left}})))),$$

$$hel_2 = (((\overbrace{\circ\circ((((\underbrace{\circ\, hel_{2,1} \,\circ\circ\circ\circ\circ\circ\circ}_{1\times 7 \text{ interior loop}}))))\circ\circ\circ\circ\circ\, hel_{2,2}}^{\text{multiloop (of degree 2)}})))), \text{ with}$$

$$hel_{2,1} = ((\overbrace{\circ(\underbrace{\circ\circ\circ\circ}_{\text{hairpin}})}^{\text{single bulge left}})), \quad hel_{2,2} = (\overbrace{\circ((\underbrace{\circ\circ\circ\circ\circ}_{\text{hairpin}}))\circ}^{1\times 1 \text{ interior loop}}), \text{ and}$$

$$hel_3 = ((\overbrace{\circ\circ(\underbrace{\circ\circ((((\circ\circ\circ))))\circ\circ\circ\circ\circ\circ\circ}_{2\times 7 \text{ interior loop}})\circ\circ}^{2\times 2 \text{ interior loop}})).$$

Note that the reading order of secondary structures in dot-bracket representation is from left to right, corresponding to the reading order of the primary structure, which again is due to the chemical structure of the RNA molecule.

It should be mentioned that the knowledge of this way of formalizing RNA secondary structure is essential for the understanding of the thesis at hand, as the presented approaches towards RNA secondary structure analysis and prediction basically rely on stochastic modeling of the *context-free language* of exactly these dot-bracket representations (details will follow in Section 3.3.4).

2.2.2.3. Representation as Ordered Trees

Besides the two popular variants discussed above, RNA secondary structures may also be modeled as rooted ordered trees, where each base pair is represented as a node. For example, secondary structures can be modeled as ordered unary-binary trees [Neb02a], where base pairs are represented as binary nodes and unpaired bases are represented as unary nodes. This effectively yields an obvious one-to-one correspondence to RNA secondary structures in dot-bracket representation.

Particularly, any (part of a) secondary structure represented by a word $(u)v$ corresponds to a (sub)tree, where the opening bracket of this word together with its corresponding closing bracket represent the root of this (sub)tree, whereas the subwords u and v represent the left and the right subtree of this root, respectively. The two subwords u and v are again (sub)words of a secondary structure in dot-bracket representation, such that this correspondence is continued recursively. Equally, a word $\circ u$ corresponds to a (sub)tree, with the symbol $\circ$ representing the root of this (sub)tree and the subword u representing the only subtree of this root.

Anyway, there also exist other elegant ways for representing RNA secondary structures as trees, for instance as ordered n-ary trees. More specifically, as ordered trees with an additional root node, where nodes may have arbitrary degrees. A corresponding definition of a tree T associated with an RNA secondary structure s might actually be given as follows:

- The leaves of T correspond to the unpaired bases in s and are colored white.
- The inner nodes of T correspond to the base pairs in s and are colored black. (This means that each inner node in T corresponds to two paired bases in s.)
- The root node of T is an additional (inner) node, which does not correspond to any part of the structure s. It should be seen as inner node corresponding to an imaginary base pair $0.(n+1)$ in s (that is, to two imaginary paired bases 0 and $n+1$ in s) if s is a secondary structure of size n. Therefore, it is represented by a squared box which is colored black.
- Each inner node (except the root node) of T is an ordered node and may have an arbitrary degree[7] $d \geqslant 2$ and thus an arbitrary number $d-1 \geqslant 1$ of children. The root node of T is also an ordered node and may have an arbitrary degree $d \geqslant 1$ and thus an arbitrary number d of children.
- The children of an inner node of T that corresponds to a base pair $i.j$ in s are exactly u leaves (white nodes) and p inner nodes (black nodes) corresponding to the u unpaired bases and p base pairs accessible from the pair $i.j$ in s. These children are ordered according to the increasing index of the bases in the structure s.

A corresponding tree is shown in Figure 2.9. It should be clear that just like the planar graph and the dot-bracket string models introduced above, such tree representations of secondary structures also abstract from the primary structure, considering only the numbers of base pairs and unpaired bases and their positions.

[7] Note that in this work, we will use the term degree in the same way as for undirected graphs. Thus, we say that a node has degree d if the corresponding vertex in the underlying undirected graph has degree d.

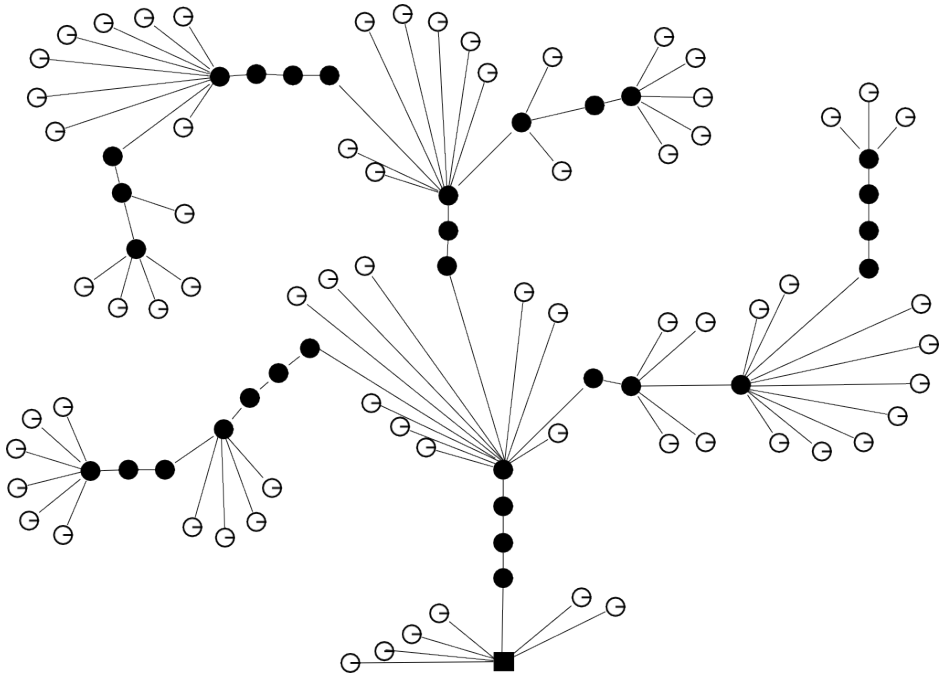


Figure 2.9.: **Representation of an RNA secondary structure as ordered tree.** The presented tree obviously corresponds to the planar graph representation shown in Figure 2.7.

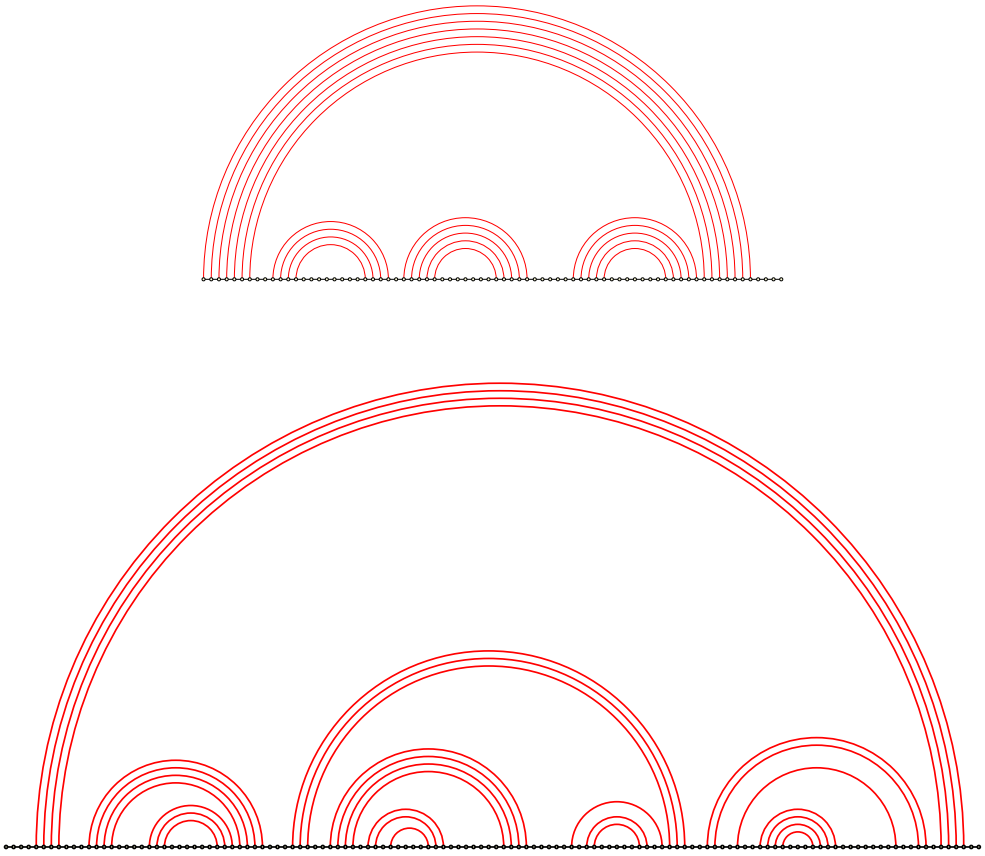


Figure 2.10.: **Arc diagrams of two RNA secondary structures.** Top (bottom) picture shows the secondary structure from Example 2.2.1 (Figure 2.7).

2.2.2.4. Other Representations

Another way of representing secondary structures on RNA sequences was introduced by Nussinov et al. [NPGK78]. There, the secondary structure of an RNA chain is represented by closing the RNA chain into a cyclic graph or planar loop, in which the vertices are identified with the bases in the chain. The base pairs in the secondary structure of the RNA chain are represented as edges in the interior of this cyclic graph, where each edge links two nonadjacent vertices. Hence, these representations of RNA secondary structures also implicitly assume a minimum size of $\min_{HL} = 1$ for hairpin loops. Furthermore, this cyclic graph has the property that the edges can all be drawn in the interior of the loop without crossing if and only if the secondary structure of the RNA chain contains no pseudoknots.

In the sequel, we will sometimes use a quite similar yet more convenient representation of RNA secondary structures for illustration purposes. This representation, usually named *arc diagram*, is also a widely known tool for modeling secondary structure. In principle, the sole difference to the one introduced in [NPGK78] is that the RNA chain is not closed into a planar loop. This actually means that a secondary structure of size n is represented by a horizontally drawn chain consisting of n vertices that represents the backbone (the sequential arrangement of its n bases), ordered from the $5'$ to the $3'$ end. Each base pairing $i.j$, $1 \leqslant i, j \leqslant n$, is represented as an arc (usually drawn above the straight base line) connecting the two vertices corresponding to i and j. Examples are given in Figure 2.10.

In addition to the discussed representations of RNA secondary structures, there are several other alternatives known from literature. But as we only intended to introduce some of the most common ones and especially the ones that will be considered in the subsequent chapters, we can finally finish this overview.

2.2.3. Shape Abstractions

Due to the diverse structural motifs that may occur even in the planar 2D conformation of RNA structures, the number of distinct secondary structures of a fixed size n is typically huge. In fact, it has been proven to grow exponentially in n (see, for instance [Wat78, SW78]). This behavior poses enormous problems in connection with both mathematical and algorithmic approaches related to the study of secondary structure, since the complete enumeration and exploration of all possible secondary structures is in general fundamentally inefficient and hence a practically insolvable task. However, (any subset of) the comprehensive set of all possible secondary structures for a given size n (or a given sequence of length n) usually contains lots of similar structures and in most cases, one is essentially only interested in structures with more fundamental differences, that is in structures that contain rather different motifs.

For this reason, the concept of *abstract shapes* was introduced by Giegerich et al. [GVR04]. Basically, abstract shapes are defined as homomorphic images of secondary structures, where each shape actually comprises a class of similar structures. Precisely, shape abstraction maps secondary structures to a tree-like domain of shapes, retaining adjacency and nesting of canonical structural features, but disregarding from helix lengths and lengths of single-stranded regions.

The idea of the so-called *abstract shapes approach* and abstract shape analysis in general is that instead of all possible secondary structures, only the corresponding shapes need to be considered. Although the number of distinct shapes corresponding to structures of size n (specifically, that are homomorphic images of secondary structures of size n) is also exponential in n [SN08, NS09], there indeed results a significant reduction of the search space when considering shape abstractions rather than complete secondary structure information, which obviously validates this approach. Notably, the maximum reduction of the search space is

reached under the assumption of the realistic structural parameter combination of $\min_{\text{hel}} = 2$ and $\min_{\text{HL}} = 3$, where the base of the exponential growth can be reduced from 1.84892 to 1.20259 (see [NS09]).

Since in the sequel, we will also utilize the concept of shape abstractions (mainly for evaluation and validation purposes), the aim of this section is to provide a short introduction to these combinatorial structures.

2.2.3.1. Hierarchy

There are five shape types for five different levels of abstraction. Two of them, namely type 1 and type 5 (also called π' and π shapes, respectively), were formally defined by a tree morphism in [GVR04]. All five different shape levels were first introduced and informally described in [SVR+06a]. However, since the different shape types were supposed to gradually increase abstraction and it was later observed that the shape definitions given in [SVR+06a] were not appropriate, the different abstraction levels were redefined (informally) in [JRG08].

In fact, analytically obtained enumeration results related to abstract shapes provided evidence that the original hierarchy of abstraction levels as introduced in [SVR+06a] was not properly ordered[8] (see [SN08]), but that the revised hierarchy proposed in [JRG08] actually satisfies the condition that higher levels of abstraction generally yield smaller numbers of shape representations and hence imply a greater reduction of the search space (see [NS09]). Due to these results, this thesis exclusively considers the renewed shape abstraction types as described in [JRG08], which will therefore now be introduced in detail.

2.2.3.2. Description of Different Levels

Common to all levels is their abstraction from loop and ladder lengths, while generally retaining nesting and adjacency of helices, but disregarding their size and concrete position in the primary structure. On the most accurate abstraction level (that is, in case of type 1 shapes), all structural components (except hairpin loops) contribute to the shape representation. The succeeding shape types gradually increase abstraction by disregarding certain unpaired regions or combining nested helices.

In general, helical regions are depicted by a pair of opening and closing squared brackets [resp.] and unpaired regions are represented by a single underscore _. Notably, this representation of abstract shapes as words over the alphabet $\{[,], \_\}$ has been originated in [GVR04] and is obviously closely connected to secondary structures in dot-bracket representation.

Therefore, we now want to consider secondary structures and shapes as strings over the respective alphabets in order to describe each of the five types (as defined in [JRG08]) separately, ordered by their degree of abstraction.

Type 1:

Most accurate – all loops and all unpaired

Accordingly, each helical region is depicted by a single pair of opening and closing squared brackets and all unpaired regions (except hairpin loops[9]) are represented as a single underscore. Thus, all structural components contribute to this shape representation, nesting and adjacency

[8] In accordance with observations independently made by R. Giegerich at about the same time (personal communication).

[9] According to the informal description of level 1 shapes given in [JRG08], it is not clear whether the (one and only but always existing) unpaired region in a hairpin must be recorded on this shape abstraction level or not. Here, we decided to follow the definition used by the RNAshapes tool, which is available at `http://bibiserv.techfak.uni-bielefeld.de/rnashapes/welcome.html`. This tool assumes that hairpin loops are not recorded.

Level 0 :	∘∘∘∘	((((	∘∘∘	$\mathtt{hel}_1$	∘∘∘	$\mathtt{hel}_2$	∘∘	$\mathtt{hel}_3$	∘	))))	∘∘
Level 1 :	_	[	_	$\mathtt{hel}_1$	_	$\mathtt{hel}_2$	_	$\mathtt{hel}_3$	_	]	_
Level 2 :		[		$\mathtt{hel}_1$		$\mathtt{hel}_2$		$\mathtt{hel}_3$		]	
Level 3 :		[		$\mathtt{hel}_1$		$\mathtt{hel}_2$		$\mathtt{hel}_3$		]	
Level 4 :		[		$\mathtt{hel}_1$		$\mathtt{hel}_2$		$\mathtt{hel}_3$		]	
Level 5 :		[		$\mathtt{hel}_1$		$\mathtt{hel}_2$		$\mathtt{hel}_3$		]	

(a) Overall structure.

Level 0 :	((((	∘∘∘∘	(((	∘∘∘∘∘∘	)))	))))
Level 1 :	[	_	[		]	]
Level 2 :	[	_	[		]	]
Level 3 :	[		[		]	]
Level 4 :	[					]
Level 5 :	[					]

(b) Substructure $\mathtt{hel}_1$.

Level 0 :	(((	∘∘	((((	∘	$\mathtt{hel}_{2,1}$	∘∘∘∘∘∘∘	))))	∘∘∘∘∘	$\mathtt{hel}_{2,2}$	)))
Level 1 :	[	_	[	_	$\mathtt{hel}_{2,1}$	_	]	_	$\mathtt{hel}_{2,2}$	]
Level 2 :	[		[	_	$\mathtt{hel}_{2,1}$	_	]		$\mathtt{hel}_{2,2}$	]
Level 3 :	[		[		$\mathtt{hel}_{2,1}$		]		$\mathtt{hel}_{2,2}$	]
Level 4 :	[		[		$\mathtt{hel}_{2,1}$		]		$\mathtt{hel}_{2,2}$	]
Level 5 :	[		[		$\mathtt{hel}_{2,1}$		]		$\mathtt{hel}_{2,2}$	]

(c) Substructure $\mathtt{hel}_2$.

Level 0 :	((	∘	(	∘∘∘∘	)	))
Level 1 :	[	_	[		]	]
Level 2 :	[	_	[		]	]
Level 3 :	[		[		]	]
Level 4 :	[					]
Level 5 :						

(d) Substructure $\mathtt{hel}_{2,1}$.

Level 0 :	(	∘	((	∘∘∘∘∘	))	∘	)
Level 1 :	[	_	[		]	_	]
Level 2 :	[	_	[		]	_	]
Level 3 :	[		[		]		]
Level 4 :	[		[		]		]
Level 5 :	[						]

(e) Substructure $\mathtt{hel}_{2,2}$.

Level 0 :	((	∘∘	(	∘∘	((((	∘∘∘	))))	∘∘∘∘∘∘∘	)	∘∘	))
Level 1 :	[	_	[	_	[		]	_	]	_	]
Level 2 :	[	_	[	_	[		]	_	]	_	]
Level 3 :	[		[		[		]		]		]
Level 4 :	[		[		[		]		]		]
Level 5 :	[										]

(f) Substructure $\mathtt{hel}_3$.

Figure 2.11.: **Shape abstractions of an RNA secondary structure.** Figures show abstract shapes (for all levels) of particular parts of the secondary structure given in Figure 2.7, based on the decomposition into structural components as presented in Example 2.2.4.

of helices are retained. As a consequence, this shape type only abstracts from loop and ladder lengths.

Type 2:

Nesting pattern for all loop types and unpaired regions in ladder interrupting bulges and interior loops

Consequently, all helical regions (ladders) are depicted by a pair of opening and closing squared brackets and furthermore, single ladder interrupting bulges and unpaired regions in interior loops are represented as a single underscore, respectively. This means that in this shape representation, nesting and adjacency of helices is still retained, but in difference to type 1 shape representations, not all structural components contribute to this shape representation, since underscores representing single-stranded regions in exterior loops and multiloops are omitted.

Type 3:

Nesting pattern for all loop types, but no unpaired regions

Shape representations of type 3 thus also retain nesting and adjacency of helices, since all helical regions are depicted by a pair of opening and closing squared brackets. But in contrast to the previously introduced two types, no unpaired regions are considered.

Type 4:

No nesting pattern for ladder interruptions by single bulges, nesting pattern for all other loop types and no unpaired regions

Compared to type 3 shapes, the only difference is that nested helices which are only interrupted by a single bulge are combined and represented by one pair of squared brackets only.

Type 5:

Most abstract – helix nesting pattern and no unpaired regions

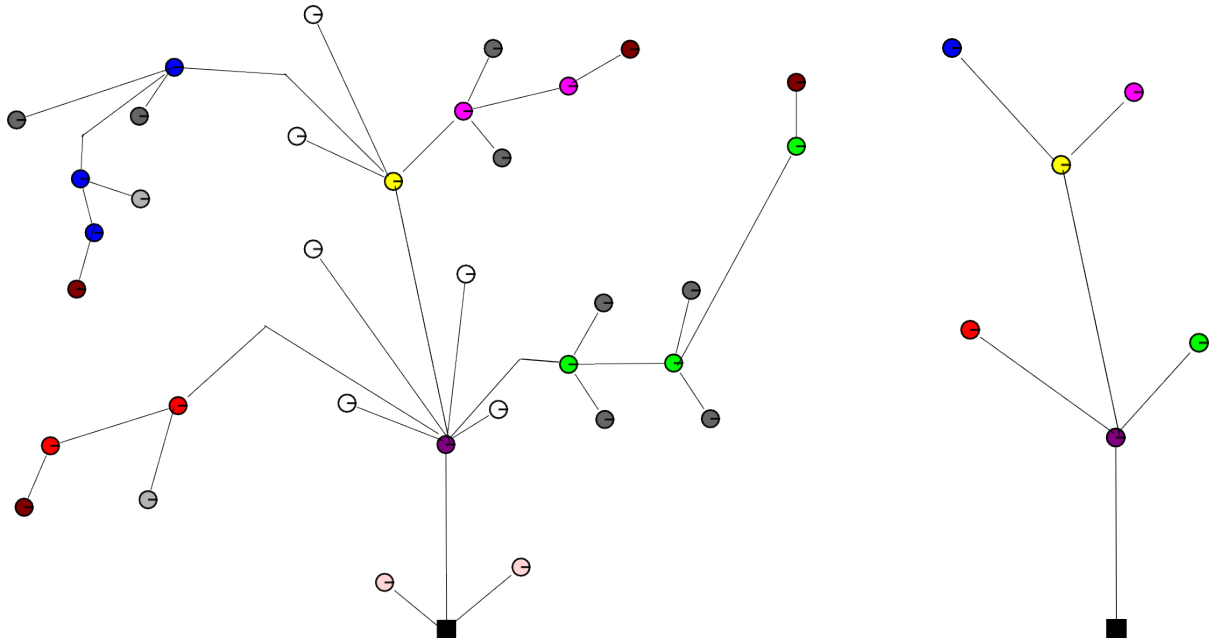
In this shape abstraction, we do not account for any helix interruptions (by single bulges or interior loops). This means that (interrupted) ladders are depicted by a pair of opening and closing squared brackets, since nested helices are now always combined.

Finally, note that secondary structures may accordingly be considered type 0 shapes, since they indeed represent the exact case, where no abstraction is used at all. The differences between the different abstraction levels are illustrated exemplarily in Figure 2.11.

2.2.3.3. Representation as Ordered Trees

The differences between the five shape levels might become more obvious when considering ordered trees rather than bracketed strings for representing secondary structures and shapes. In fact, according to the definition of a tree associated with a secondary structure (as given in Section 2.2.2.3), a corresponding tree T associated with a type 1 shape *sh* has the following properties:

- The leaves of T correspond to the unpaired regions in *sh* and are colored white.
- The inner nodes of T correspond to the paired regions in *sh* and are colored black. (This means that each inner node in T corresponds to an uninterrupted ladder in *sh*.)
- The root node of T is an additional (inner) node, which is represented by a squared box and colored black.
- Each inner node (except the root node) of T is an ordered node and may have an arbitrary degree $d \geqslant 2$ and thus an arbitrary number $d - 1 \geqslant 1$ of children. The root node of T is also an ordered node and may have an arbitrary degree $d \geqslant 1$ and thus an arbitrary number d of children.



(a) Type 1 shape. (b) Type 5 shape.

Figure 2.12.: **Tree representations of abstract shapes.** The presented trees correspond to shape abstractions of the secondary structure shown in Figure 2.8.

- The children of an inner node of T that corresponds to a paired region *reg* in *sh* are exactly u leaves (white nodes) and p inner nodes (black nodes) corresponding to the u unpaired regions and p paired regions accessible from the paired region *reg* in *sh*. These children are ordered according to the increasing indices of the bases that form these regions in the structure *sh*.

Anyway, a tree associated with a secondary structure can easily be transformed into a corresponding tree for shape abstraction level 1 in the following way:

1. Delete each inner node of degree 2 whose two neighbors are inner nodes (deleting such a node corresponds to merging the two incident edges into a single edge).
2. Contract each bunch of leaves (white nodes) that are incident to the same inner node (black node) into a single leaf (contracting such a bunch of leaves corresponds to deleting all but one of these leaves).

To illustrate the reduction from secondary structures to type 1 shapes, let us consider the tree in Figure 2.9. A colored version of the resulting type 1 tree is shown in Figure 2.12a. Comparing this colored tree to the colored secondary structure shown in Figure 2.8, it is easy to see that all leaves and inner nodes of certain degrees in this tree correspond to certain structural components in the secondary structure.

Similar reduction procedures for obtaining corresponding trees associated with any of the remaining four shape abstraction levels, respectively, can straightforwardly be derived. Basically, we simply have to delete all nodes having some specific properties, respectively. In fact, the four different types of reduction are very transparent considering tree structures, and will therefore not be described in detail.

However, consider Figure 2.12b, which shows a colored version of the tree associated with the type 5 shape of the secondary structure given in Figure 2.8. Comparing this compact tree to the original secondary structure, it is easy to see that on the highest abstraction level, only the number and adjacency of helical regions is considered.

Chapter 3

Computational RNA Structure Prediction

This chapter gives an overview on existing methods connected to RNA secondary structure analysis and prediction. It basically summarizes the current state of research in the area of this thesis. Formalities and details will only be given if they contribute to the understanding of subsequent matters or if they will be needed in the sequel.

3.1. Background

In bioinformatics, we aim for algorithms predicting the structure of functional RNA from its sequence, since in many cases there exists only knowledge on the linear primary structure but no information on corresponding higher-dimensional conformations of an RNA molecule. Due to an exponential growth of the number of possible conformations with respect to the size of a molecule, a brute-force attempt is out of reach, motivating the development of more sophisticated methods.

In fact, computational prediction of RNA secondary structures has become a research field of great importance over the past decades, for the following reasons: First, complete tertiary structure prediction is computationally complex. Particularly, by allowing pseudoknots (which obviously must be considered in tertiary studies), the problem of RNA structure prediction becomes $\mathcal{NP}$-complete [LP00]. This means under the assumption that $\mathcal{P} \neq \mathcal{NP}$[1], no procedure can solve it in less than exponential time for any given input. Forbidding pseudoknots, however, this task is computationally feasible. In fact, algorithms for predicting RNA secondary structure without pseudoknots typically run in $\mathcal{O}(n^3)$ time for n the length of a single input sequence. On the other hand, it has proven convenient to first search for the secondary structure of a molecule before determining its final 3D conformation, since most of the tertiary structure is determined by secondary interactions. Therefore, it is customary for prediction algorithms to allow only non-crossing (that is, hierarchically nested or adjacent) base pairs defined by the secondary structure in order to support practical applicability.

The first method that has been developed in order to predict RNA secondary structure was *comparative sequence analysis* [PSOJ89, GPH+92, PTW99]. Briefly, this approach relies on evolutionary information and is motivated by the fact that homologous RNAs tend to conserve a common base paired secondary structure called *consensus*. In fact, base pairs that have been conserved by evolution are extremely likely to be a part of the functional form. Base pairings of the predicted consensus RNA secondary structure are therefore inferred by determining canonical pairs that are common among multiple structurally homologous sequences. Note that if a sufficiently large number of homologous sequences are available, predictions obtained by comparative sequence analysis are extremely reliable (see, for example, [GLC02, MW90]). However, despite the strikingly high accuracy of produced structure predictions, the required large number of sequences (as well as required time and human expertise) are considered a major drawback in connection with this purely comparative approach.

Due to these facts, a number of automated approaches for RNA secondary structure prediction have been devised that use (a particular kind of) biophysical knowledge of what structures are most plausible (according to the considered biophysical information) in order to compute a corresponding prediction on the basis of a single input sequence. However, such single sequence approaches have a difficult job detecting (either one or a rather small number of) reasonable foldings from the typically tremendously large set of possible secondary structures for the considered input sequence.

In fact, any RNA molecule can be folded in many different ways, and the resulting number of distinguishable feasible secondary structures for a given input sequence grows exponentially in the corresponding sequence length. According to this behavior, the main idea of automated computational prediction methods is to calculate those foldings of a given input sequence that are considered the "best" ones among all feasible secondary structures for that sequence. These structures are generally identified according to a particular quality criterion which describes how good the respective foldings actually are (compared to all others). Hence, the problem of RNA secondary structure prediction can in many cases be described as an *optimization problem*,

[1] Traditionally, $\mathcal{P}$ represents the class of all problems that can be solved on a deterministic sequential machine in an amount of time that is polynomial in the size of the input, whereas $\mathcal{NP}$ denotes the class of all those problems that can be solved in polynomial time on a non-deterministic machine.

where the goal is to identify the *optimal* folding for a given input sequence. That is, a feasible secondary structure that minimizes or maximizes a particular (mathematically defensible and reasonable) *objective function* for rating the possible structures.

Notably, different kinds of objective functions can be considered in order to obtain accurate RNA structure predictions. Particularly, in most cases, the value of the objective function represents the free energy according to a general (incomplete) thermodynamic model, and the optimization problem is actually a minimization problem (details will follow in Section 3.2.2). Alternatively, the objective function often represents the structure probability according to a specific stochastic model for RNA secondary structure, derived on the basis of structural information obtained from reliable RNA databases. In this case, the RNA folding problem obviously results in a maximization problem (details will follow in Section 3.3.7).

3.2. Methods Based on Experimental Physical Information

Most of the published RNA secondary structure prediction algorithms build on a particular (more or less complex) model of the *Gibbs free energy* or *Gibbs energy* G describing the energetics of molecules in aqueous solution. More precisely, the Gibbs free energy G is a thermodynamic potential measuring the "useful" or process-initiating work obtainable from a thermodynamic process in which the temperature of the system stays constant.

Since in nature, molecules in solution arrange themselves so as to minimize the free energy of the entire system, the free energy of an RNA molecule is the potential of the molecule to release more energy by folding, that is by creating additional (hydrogen) bonds by intramolecular base pairing. Hence, the change ΔG of Gibbs free energy in a chemical process, such as RNA folding, determines the direction of the process:

- $\Delta G < 0$ indicates a *favorable* or *stabilizing* process,
- $\Delta G = 0$ indicates equilibrium and
- $\Delta G > 0$ indicates an *unfavorable* or *destabilizing* process.

Consequently, it can reliably be assumed that the native (2D or 3D) structure of a given molecule is equal to the one with the lowest free energy ΔG, as this corresponds to the most stable structure than can be reached according to the rules of base pairing.

The aim of this section is to provide a short review on thermodynamic models for RNA secondary structure and on different kinds of energy based computational methods that can readily be applied for structure prediction.

3.2.1. Thermodynamic Models

We start with a short overview of the development and complexity of state-of-the-art thermodynamic models for RNA secondary structures. We also discuss some major problems connected to such physics-based RNA models.

3.2.1.1. Background and Development

In the early 1970s, biochemists hypothesized that each base pair in a helix contributes to the stability of that helix and that the contribution of a base pair depends on its adjacent base pairs [GC73, BDTU74]. This yielded a new model in which the thermodynamic stability of a given base pair is dependent on the identity of its *nearest neighbor*, the so-called *individual nearest-neighbor (INN) model*. Later on, an expanded nearest-neighbor model for formation

of RNA helices with canonical base pairs was presented, which was termed the *individual nearest-neighbor hydrogen bond (INN-HB) model* [XSB+98, MSZT99].

Thermodynamics for RNA secondary structures have also been studied for all other common substructures. These studies led to a number of different thermodynamic parameters for certain (special) types of loops along with corresponding loop-dependent free energy rules. These results are summarized in [ST95] (for the INN-model), as well as in [MSZT99] and in [ZMT99] (for the INN-HB model).

Since the Gibbs free energy G depends on the temperature and the thermodynamic parameters for basic structural motifs utilized in state-of-the-art energy models are usually estimated or extrapolated from chemical melting experiments at 37°C (see, for example, [XSB+98, MSZT99, MDC+04]), many authors use $\Delta G^\circ_{37}(s)$ to denote the free energy of a secondary structure s at 37°C. Notably, in loop-dependent energy models, the overall energy $\Delta G^\circ_{37}(s)$ of a given secondary structure s is assumed to be given by the sum of the free energies of all its substructures, formally

$$\Delta G^\circ_{37}(s) = \Delta G^\circ_{37}(\mathrm{Loop}_e) + \sum_{i,j \in s} \Delta G^\circ_{37}(\mathrm{Loop}(i,j)).$$

3.2.1.2. The Turner Model

To date, most prediction algorithms use the INN-HB model with loop-dependent energy rules [XSB+98, MSZT99] to compute the free energy of RNA secondary structures[2]. This standard thermodynamic model is most commonly called Turner's energy model and therefore, we will use this name in the sequel, too.

Notably, the Turner model basically relies on the k-loop decomposition of a secondary structure s according to Definition 2.2.5. Moreover, it additionally distinguishes between a number of special types of k-loops. More specifically, the following (special) loop types are considered:

- hairpin loops of size 3, called *triloops*,
- hairpin loops of size 4, called *tetraloops*,
- hairpin loops of size > 4,
- stacked pairs,
- bulge loops of size 1, called *single bulges*,
- bulge loops of size > 1,
- 1×1 interior loops, called *single mismatches*,
- 2×2 interior loops, called *tandem mismatches*,
- 1×2 (resp. 2×1) interior loops,
- non-grossly asymmetric interior loops of size > 4,
- grossly asymmetric interior loops (GAILs),
- multiloops and
- exterior loops.

In particular, for hairpin loops, the thermodynamic parameters and free energy rules include a length-dependent loop destabilizing free energy and a terminal mismatch stacking energy (for loops of size $\geqslant 4$) resp. the terminal AU/GU penalty (for loops of size 3). Additionally, a GGG loop bonus (applies only to gu closed hairpins in which a $5'$ closing g is preceded by two g residues) and a penalty term for poly-C hairpin loops (that is, for hairpin loops in which all unpaired nucleotides are c), as well as a tetraloop bonus (for hairpin loops of size 4) are included.

[2] Note that only Watson-Crick and wobble GU pairs are allowed in this INN-HB model, as nearest neighbor rules break down for non-canonical base pairs. This means that non-canonical base pairs in helices must instead be treated as mismatched pairs for the computation of free energies.

For bulge loops, a length-dependent loop destabilizing free energy, as well as the terminal AU/GU penalty for both the interior and exterior base pair (for loops of size > 1 only) are included in the model. For single bulges and for stacked pairs, a stacking energy for the stacking interaction of the interior and exterior base pair is added.

Small symmetric interior loops and almost symmetric interior loops, particularly 1×1, 2×2 and 1×2 interior loops, are treated in a special way, since for these loops, individual sets of free energy values are consulted that contain values for every possible sequence variation. For all other interior loops, the thermodynamic parameters include a length-dependent loop destabilizing free energy and a free energy contribution that penalizes asymmetry in the loop. Additionally, a terminal mismatch stacking energy (for loops of size > 4 that are no GAIL) resp. two free energies associated with the terminal base pairs of the two helices in which the loop ends (for GAILs) is added to the stability of the loop.

Finally, for multi- and exterior loops, the terminal AU/GU penalty and a free energy contribution for the stacking interaction of a base pair with (0, 1 or 2) single-stranded bases adjacent to that base pair are explicitly applied to all the terminal base pairs of the helices that are radiating out from this loop[3]. Additionally, for multiloops, a destabilizing initiation free energy is added, which depends on the number of single-stranded bases and on the number of base pairs accessible from the closing base pair of the loop.

Note that in this model, the terminal AU/GU penalty term for a terminal au or gu base pair at the end of a helix is added to the free energy of a given secondary structure s along with the free energy of the loop $\mathrm{Loop}(i.j)$ closed by a base pair $i.j \in s$ in which the helix terminates. This means that the terminal AU/GU penalty, if necessary, is formally assigned the loop $\mathrm{Loop}(i.j)$ closed by the pair $i.j \in s$, although it really belongs to the helix in which the loop ends.

For a much more detailed and formal description of the Turner energy rules, we refer to [Sch07].

3.2.1.3. Problems and Limitations

Although thermodynamic parameters and energy rules for particular secondary structure motifs have undergone a number of refinements in oder to provide a more accurate characterization of folding thermodynamics, there are still uncertainties in both the experimentally obtained estimates of the parameters as well as in some of the applied rules, especially for loops.

In fact, even the rather complex INN-HB model with loop-dependent energy rules is still incomplete. For instance, non-canonical base pair interactions are so far not taken into account. While such pairings poorly contribute energetically, it is known that they play a stabilizing role [LW06] which is neither fully understood nor modeled yet. Moreover, some known sequence effects on the stabilities of basic motifs (specifically, bulges and single non-canonical base pairs) that are not nearest-neighbor ones [LKT90, KBT99] are not included in the actual model. The same holds for many still to discover strong sequence dependencies on the stability of particular interior loops [SBT99, CZJT04].

Note that due to the absence of measured estimates, some of the thermodynamic parameters considered still need to be extrapolated to loop sequences and loop sizes, respectively, substantiating the suspicion that those are inevitably inaccurate. The resulting energy contributions for the corresponding loop types can thus only be approximations. Hence, the free energy

[3] Note that if $i.j$ and $(j+2).l$ are two base pairs, then r_{j+1} can interact with both of them. In this case, the stacking is assigned to only one of the two base pairs, whichever has a lower free energy (usually the 3′ stack). In fact, the sum of all the free energy contributions for stacking of single-stranded bases to the terminal base pairs has to be minimized.

computed for an overall secondary structure is an approximation as well, because of the assumption of free energy additivity.

There are also a number of other physical and biochemical limitations in connection with experimental parameter estimation, with respect to ion concentration, temperature and entropic effects (see, for instance, [MT06]). Moreover, it is practically impossible to incorporate information on folding kinetics, since certain important chemical aspects, including the influences of proteins/enzymes or of co-transcriptional folding [MM04] and mechanisms such as RNA thermometers and riboswitches [MRB04], can simply not be measured in terms of free energy. As a consequence, although the Turner model is considered valid for any type of RNA, it encounters specific problems for particular types of RNA. For instance for tRNAs, where it is well-known that modified nucleotides introduce problems for structure prediction [RCM99].

Due to these facts, an increasingly accepted alternative to considering the experimentally derived Turner parameters is to use statistical estimates of the different energy parameters, obtained on the basis of structural information from reliable RNA databases. Details will follow in Section 3.3.10.

3.2.2. Energy Minimization Paradigm

As already noted, the problem of RNA secondary structure prediction basically represents an optimization (minimization or maximization) problem, as the simplest way is to identify a single secondary structure that is optimal in some sense – that yields the minimum or maximum value of the considered objective function. In connection with thermodynamics of RNA structures, the objective function usually represents the free energy of the system being modeled. As in nature, every RNA molecule seeks to achieve a minimum of free energy by folding into a higher-dimensional conformation, it is assumed that the correct (optimal) structure is the one with the lowest free energy. For these reasons, many prediction methods use free energy as their metric and try to compute a conformation of minimum free energy (MFE).

3.2.2.1. Optimal Structure Prediction

The most successful and popular method for energy minimization is to use a dynamic programming (DP) algorithm. In the context of optimization problems (here minimizing the free energy), DP algorithms implicitly check the values of a particular objective function (here free energy according to a thermodynamic model) of all possible solutions (here feasible secondary structures) for a given problem instance (here RNA sequence) without explicitly generating the respective solutions, such that this method takes far less time than the naive one. Generally, DP simplifies a complicated problem (here MFE structure calculation) by breaking it down into simpler subproblems in a recursive manner.

This is accomplished in two steps: In the first step, the optimal objective function value (here MFE) is derived for all different parts of the problem, that is all subproblems (here all possible sequence fragments), starting with simplest ones (here the shortest fragments) and then continuing with more complex ones (here with longer fragments). Notably, previously determined solutions for simpler problems (here MFEs for shorter fragments) are used in order to speed up computations; therefore, the objective function values of the respective optimal solutions need to be stored in a corresponding matrix. This first step is finished once the most complex subproblem instances, that is the main problem instance (here the overall sequence) has been considered. Then, the objective function value of the optimal solution (here the MFE value) is known and the optimal solution (here MFE folding) has to be determined in a subsequent second step, called *traceback*, which makes use of the objective function values (here free energies) calculated during step one.

The principal recursion scheme for non-pseudoknotted RNA secondary structure prediction is given as follows [NPGK78, NJ80]:

$$
\begin{aligned}
&M_{i,i-1} = 0, \text{ for } 2 \leqslant i \leqslant n,\\
&\quad M_{i,i} = 0, \text{ for } 1 \leqslant i \leqslant n, \text{ and}\\
&\quad M_{i,j} = \min \begin{cases} eu_i + M_{i+1,j} \text{ (i becomes unpaired)},\\ M_{i,j-1} + eu_j \text{ (j becomes unpaired)},\\ ep_{i,j} + M_{i+1,j-1} \text{ (i basepairs with j)},\\ \min_{i<k<j-1} M_{i,k} + M_{k+1,j} \text{ (bifurcation)}, \end{cases} \qquad \text{for } 1 \leqslant i \leqslant j-1 \text{ and } 1 \leqslant j \leqslant n,
\end{aligned}
$$

where eu_i and $ep_{i,j}$ are scores (usually free energies) for an unpaired base i and for a pairing of two bases i and j, respectively, and n is the length of the input sequence. The complete DP algorithm for obtaining the optimal (usually MFE) folding of a molecule of length n according to the considered scores eu_i and $ep_{i,j}$ can be implemented to run in $\mathcal{O}(n^3)$ time, where $\mathcal{O}(n^2)$ space is required for storing the DP matrix M. For details, we refer to [Edd04].

Nevertheless, it should be mentioned that the pioneering work in this domain was published in 1978 by Nussinov et al. [NPGK78], where the authors introduced an efficient DP algorithm for maximizing the number of complementary base pairs. Effectively, the considered recursion scheme relies on a quite simple energy model. In particular, each complementary base pair $i.j$ in a given secondary structure s is assigned an energy $e(i.j)$ and the overall energy of the secondary structure s is given by

$$\Delta G(s) = \sum_{i.j \in s} e(i.j), \tag{3.1}$$

where the stability of gc pairs is considered to be equal to that of au pairs. Additionally, contributions due to stacking of base pairs and destabilizing effects for loop formation were ignored. Obviously, the simple free energy function ΔG is minimized when the secondary structure contains the maximum number of complementary base pairs. Hence, this particular algorithm from [NPGK78] basically assumed $eu_i = 0$ and

$$ep_{i,j} = \begin{cases} -1, & \text{if } i.j \text{ can become a complementary base pair},\\ 0, & \text{if } i \text{ and } j \text{ may not become a base pair}, \end{cases}$$

for any $i, j \in \{1, \ldots, n\}$, where n is the length of the input sequence. This actually simplified the problem of folding a nucleotide sequence into a structure with minimal free energy to the problem of finding a structure with maximum number of base pairs. In fact, the computed folding contained the maximum number of base pairs that could be found for the entire nucleotide sequence.

By utilizing a simple method for estimating the free energy of loops found in single-stranded RNA based on their sequence, which was derived earlier [TUL71], the folding rules of this DP algorithm for maximal matching were modified to allow an estimate of the free energy of loop structures based on sequence data [NJ80]. This means that values for base pairing energies were incorporated into the algorithm. More precisely, hydrogen bond potential energies $e(i.j)$ are computed for each base pair $i.j$, such that the algorithm computes one structure with the lowest free energy ΔG. But as these energy rules are only base pair-dependent, stacking and destabilizing energies were not incorporated into this algorithm.

In 1981, Zuker and Stiegler then presented a novel DP algorithm for folding an RNA molecule that finds a conformation of minimum free energy using thermodynamics and auxiliary information [ZS81]. This algorithm uses loop-dependent energy rules to compute the free

energy of each loop, such that the overall energy of a secondary structure s is given by

$$\Delta G(s) = \sum_{i,j \in s'} e(\text{Loop}(i,j)).$$

During the following years, this DP algorithm based on thermodynamic parameters has been improved several times (see, for instance [SKMC83, ZS84, Zuk89a]).

3.2.2.2. Suboptimal Structure Prediction

Anyway, although these early attempts obviously propose an elegant way for computing the optimal (MFE) structure, the predicted folding may however often be wrong, mainly for the following reasons:

1. Due to the facts discussed in Section 3.2.1.3, the thermodynamic parameters of a particular energy model underlying the DP recursions may still be lacking some accuracy. Moreover, it is known that slightly different physiological conditions may alter the values of the considered default parameters and even slight deviations in the free energy parameters can lead to substantial differences in the computed optimal folding (see [Zuk00]).
   On the other hand, for some effects that are known to have a relevant influence on the stability of the overall structure and that can indeed be modeled by corresponding energy rules, there still exists no efficient solution on how to include them into a corresponding DP algorithm. A well-known example in this context is the asymmetry of distribution of unpaired bases in multiloops (see, for instance [MT02b]).
   For these reasons, the true MFE structure of an RNA molecule that can be assumed to fold in nature may not be equal to the optimal one computed by DP. In fact, the native folding might be only a *suboptimal* structure with respect to the underlying energy model.

2. According to [Pon08], the ignorance of all tertiary interactions (including pseudoknots) may favor a conformation whose best planar structure is promising over another conformation with lower free energy, but whose best planar structure has higher free energy than that of the first.

3. Evidences have shown that some RNA molecules can adopt more than one alternative fully functional conformation, as in the case of natural riboswitches [BCW+04, TB05] that change structure in response to their environment. Actually, in biochemical systems, a number of suboptimal foldings might be found in addition to the actual MFE structure, and these suboptimal conformations may also play a functional role [INN+03].

To overcome these limitations of optimal MFE structure prediction, more elaborate DP algorithms have been invented that efficiently generate a set of *suboptimal solutions* (alternative low energy structures, called *candidate structures*) for a given input sequence in order to provide significantly more information, attempting for more accurate structure predictions. Notably, these algorithms fundamentally use the same DP recursions as presented above, but they require more terms and more matrices for enabling suboptimal structure generation.

For example, a DP algorithm for determining RNA secondary structures within any prescribed increment of the computed global MFE was introduced by Zuker [Zuk89b]. This algorithm was implemented in the `Mfold` software, which has become a widely used program to predict RNA secondary structures. The description and use of the `Mfold` package has appeared in a number of articles [JTZ89, JTZ90, Zuk94, ZMT99]. `Mfold` is also available as an online web server [Zuk03], which by the way is recorded as one of the oldest in computational molecular biology. Note that while the `Mfold` web server is still in use, a few years ago, the `Mfold` software package has been replaced by `UNAFold` [MZ08], a corresponding software package that is much easier to install and run and that offers many more types of computations. Other methods for computing suboptimal RNA structures can be found, for instance in [NYY95].

3.2.2.3. Exhaustive Suboptimal Structure Prediction

However, a limitation of the method proposed in [Zuk89b] is that it computes a constrained optimal folding for each admissible pair. This means the constraint is a prescribed, fixed base pair, respectively. If that pair is in the optimal folding, then the suboptimal algorithm will regenerate this optimal structure. As a consequence, if n_0 denotes the number of base pairs in the optimal folding, then at most $\frac{n\cdot(n-1)}{2} - n_0$ suboptimal foldings are examined by this algorithm for a sequence of length n. A structure that is not one of the constrained optimal foldings generated by its base pairs is in the complementary set of suboptimal foldings excluded by the suboptimal algorithm. This collection of missing suboptimal foldings includes for instance such structures that are specified by the removal of one or more base pairs from the optimal folding. For these reasons, the number of alternative structures in the computed set of suboptimal foldings is obviously limited.

Nevertheless, a DP method for exhaustive suboptimal structure determination was published by Wuchty et al. [WFHS99]. Representing a more analytical treatment than an earlier attempt in this area [WT86], the corresponding algorithm actually computes all suboptimal foldings within an energy increment above the MFE, that is it predicts all possible secondary structures whose free energy falls within a prescribed distance from the MFE. The utilized DP recursions are similar to those for standard energy minimization as discussed above, but there is one important exception, namely that all secondary structures are considered once and only once by non-redundant recursions. However, since the number of suboptimal foldings grows exponentially with increasing length of considered energy interval (that is, with growing distance from the MFE), this algorithm can only be used to explore a small free energy interval close to the MFE.

3.2.3. Partition Function Approach

Notably, the suboptimal algorithms from [Zuk89b, WFHS99] have proven useful with respect to mitigating the uncertainties in the predictions. However, neither method is capable of analyzing the complete set of possible foldings in order to assess the relevance of a particular secondary structure. This discrepance led to a shift form the prevalent MFE paradigm towards the exploration of the complete *ensemble* of secondary structures, that is the set of all feasible foldings that can be folded for a given sequence.

3.2.3.1. Boltzmann-Weighted Ensemble

The essence of this alternative view of the RNA folding problem is that the free energy of possible foldings follows a so-called *Boltzmann distribution* (sometimes called *Boltzmann equilibrium probability distribution*), in which each secondary structure s compatible (according to the rules of base pairing) with a given RNA sequence r is associated with its *Boltzmann (equilibrium) probability* $\Pr(s \mid r)$ corresponding to its molecular free energy $\Delta G(s)$. Specifically,

$$\Pr(s \mid r) = \frac{\exp\left(-\frac{\Delta G(s)}{RT}\right)}{Z_r}, \tag{3.2}$$

where R is the universal gas constant, T is the temperature measured in Kelvin degrees and Z_r is the normalizing constant of the corresponding Boltzmann distribution, thus defined as follows:

$$Z_r := \sum_{s \text{ compatible with } r} \exp\left(-\frac{\Delta G(s)}{RT}\right).$$

Particularly, Z_r is the value of the *(equilibrium) partition function* for all feasible foldings of the RNA sequence r. In general, the partition function (PF) provides a measure of the total number of states (here secondary structures) weighted by their individual energy at a particular temperature. Hence, the Boltzmann distribution statistically characterizes the corresponding ensemble, the so-called *Boltzmann ensemble* or *Boltzmann-weighted ensemble*.

3.2.3.2. Partition Function Calculation

An elegant DP algorithm for calculating the complete secondary structure PF for a given input sequence was presented by McCaskill [McC90], where just like in case of DP methods for MFE computation, all possible structures are implicitly considered without explicitly generating them. In principle, DP recursions for computing PFs can be derived for any reasonable free energy model for RNA secondary structure, where the additivity of free energy readily implies a multiplicativity of contributions for distinguished structural elements to the PFs.

Independent on the utilized recursions, PFs are determined for all possible sequence fragments, starting with fragments of length zero and working outwards recursively on increasingly longer sequences. In contrast to the recursions used for MFE computations, however, which constantly take the minimum of a number of terms, for PF calculation the corresponding terms need to be summed, since the contributions to the PFs by mutually exclusive conformational cases are actually additive. One crucial prerequisite that must be taken care of is that the recursions that sum the PFs from shorter fragments need to be non-redundant, that is each configuration must be counted once and only once.

In principle, when considering a loop-dependent energy model (like Turner's), then in the derivation of the recursions for PF values $Z_{i,j}$, $1 \leqslant i, j \leqslant n$, for subsequences $r_i \dots r_j$ of a particular input sequence r of length n, we have to take into account the mutually exclusive and exhaustive cases illustrated in Figure 3.1. Accordingly, the corresponding recursion scheme for computing the equilibrium PF $Z_r = Z_{1,n}$ can be defined as follows [DL03]:

$$
\begin{aligned}
Z_{i,i-1} &= 1, \text{ for } 2 \leqslant i \leqslant n, \\
Z_{i,j} &= 1, \text{ for } 1 \leqslant i \leqslant j \leqslant i+3 \leqslant n, \text{ and otherwise,} \\
Z_{i,j} &= \exp\left(-\frac{e1_{i,j}}{RT}\right) + \\
&\quad ZP_{i,j} \cdot \exp\left(-\frac{e2_{i,j}}{RT}\right) + \\
&\quad \sum_{i<h<j} ZP_{h,j} \cdot \exp\left(-\frac{e3_{i,j,h}}{RT}\right) + \\
&\quad \sum_{i<l<j} ZP_{i,l} \cdot \exp\left(-\frac{e4_{i,j,l}}{RT}\right) \cdot Z_{l+2,j} + Z_{l+1,j} - Z_{l+2,j} + \\
&\quad \sum_{i<h<l<j} ZP_{h,l} \cdot \exp\left(-\frac{e5_{i,j,h,l}}{RT}\right) \cdot Z_{l+2,j} + Z_{l+1,j} - Z_{l+2,j},
\end{aligned}
$$

where $e1_{i,j}$, $e2_{i,j}$, $e3_{i,j,h}$, $e4_{i,j,l}$ and $e5_{i,j,h,l}$ are free energy contributions that result in the corresponding mutually exclusive case, respectively, according to the utilized thermodynamic parameters and energy rules.

Note that this DP scheme for calculating Z_r makes heavy use of PF values $ZP_{i,j}$, $1 \leqslant i, j \leqslant n$, for subsequences $r_i \dots r_j$ with the ends constrained to form a base pair. Those values thus have to be derived beforehand by a corresponding DP algorithm for computing the equilibrium PF

$ZP_r = ZP_{1,n}$, which exclusively considers paired structures without dangling ends, formally:

$$ZP_r := \sum_{s \text{ compatible with } r \text{ for which } 1.n \in s} exp\left(-\frac{\Delta G(s)}{RT}\right).$$

According to standard models of free energy, the corresponding recursion scheme is principally given as follows [DL03]:

$$ZP_{i,j} = 0, \text{ for } 1 \leqslant i \leqslant j \leqslant i+3 \leqslant n, \text{ and otherwise,}$$
$$ZP_{i,j} = exp\left(-\frac{eh_{i,j}}{RT}\right) + exp\left(-\frac{es_{i,j}}{RT}\right) \cdot ZP_{i+1,j-1} + \sum_{i<h<l<j} exp\left(-\frac{ebi_{i,j,h,l}}{RT}\right) \cdot ZP_{h,l} + ZM_{i,j},$$

where $eh_{i,j}$, $es_{i,j}$ and $ebi_{i,j,h,l}$ are free energies for a hairpin loop, stacked pair and bulge or interior loop closed by i.j, respectively, according to the underlying model. The contributions of the remaining mutually exclusive conformational cases to the PF $ZP_{i,j}$ are expressed in terms of another PF $ZM_{i,j}$ for sequence fragments $r_i \dots r_j$ of the overall sequence r, where the ends are constrained to form the foundation (closing base pair) of a multiple loop. Hence, the equilibrium PF $ZM_r = ZM_{1,n}$ of a given input sequence r of length n has to be computed before ZP_r such that finally Z_r can be derived. It remains to mention that the DP recursions for calculating the values of $ZM_{i,j}$, $1 \leqslant i,j \leqslant n$, are not quite as straightforward as the ones presented above if comprehensive Turner free energy parameters are used (see, for example [McC90, DL03]).

However, the complete DP algorithm for determining the normalizing constant Z_r for the Boltzmann equilibrium probability distribution of all secondary structures compatible with a given input sequence r of length n can still be implemented to scale $\mathcal{O}(n^3)$ in time and $\mathcal{O}(n^2)$ in storage. In fact, only one restriction has to be imposed on the class of feasible secondary structures in order to ensure cubic computation time, namely that single-stranded regions of bulges and interior loops may not contain more than a constant number max_{bulge} of unpaired nucleotides. This effectively means that the algorithm is cubic when long strands in bulges and interior loops (for instance of lengths $max_{bulge} > 30$) are disregarded, otherwise its running time would be $\mathcal{O}(n^4)$ for n the size of the input sequence.

3.2.3.3. Base Pairing Probabilities

Given the PF Z_r for a particular RNA sequence r, the Boltzmann probability of any possible secondary structure s of r can immediately be determined according to equation 3.2. However, due to the generally huge (specifically, exponential in the length n of r) size of the Boltzmann ensemble, the probability of a particular structure – even that of the MFE structure – is extremely small and hence not very meaningful.

Nevertheless, the number of possible base pairings for a given sequence is in most cases considerably smaller. To be precise, for any value of min_{HL} (which is implicitly considered to be equal to 3 in connection with energy models), there are at most $\lfloor \frac{n-min_{HL}}{2} \rfloor \in \mathcal{O}(n^2)$ possible base pairs i.j, $1 \leqslant i,j \leqslant n$, that can be formed on a particular sequence of length n. For this reason, the probabilities for two non-identical bases to be paired are typically greater and thus more enlightening in the context for RNA analysis than the overall structure probabilities. In fact, base pairing probabilities are of high significance, for the following reasons: First, they provide measures of confidence for MFE predictions, as many of the low free energy structures share the same base pairs [Mat04]. Second, it is known that base pairing probabilities are less affected by uncertainties in free energy parameters than is the MFE structure [LB05].

An elegant way for determining base pairing probabilities in accordance with a given input sequence is to use a corresponding DP traceback routine after PF calculations, as devised in [McC90].

3.2.3.4. Extensions

The paradigm-shifting DP algorithm from [McC90] for calculating the PF and base pairing probabilities has indeed laid the foundation for some relevant developments connected to the analysis of RNA structure. For instance, an efficient DP algorithm capable of calculating the mean and variance (and any moments in general) of the Boltzmann-weighted free energy distribution for a fixed RNA sequence has been presented and these ensemble characteristics have been reported to be useful for distinguishing biological sequences from random sequences [IMN05].

Although the algorithm in its original variant can not directly be applied for RNA folding, it has been incorporated in several state-of-the-art secondary structure prediction tools, including `RNAfold`, an integrated component of the famous `ViennaRNA` package [HFS+94], and `RNAstructure` [Mat04]. Notably, it therefore has been extended to include parameters for coaxial stacking [WT94, MDC+04, TM07], which is a favorable interaction of two helices stacked end to end (in multiple and exterior loops). More important, however, are the extended PF algorithms (such as [Mat04]), statistical sampling methods (especially [DL03]) and clustering techniques (for example [CLD05]) that have been invented over the last years. Of very special interest to us are the sampling extensions of the PF approach, which will therefore be discussed in detail in the following section.

3.2.4. Statistical Sampling

Note that all of the above described existing approaches have only partially addressed the ensemble. In fact, by employing the algorithm from [Zuk89b], the number of alternative suboptimal structures is inevitably limited. The complete suboptimal method from [WFHS99] does naturally stronger address the issue of ensemble, but the exhaustive suboptimal structure sets provided only a first picture of the structural landscape for energies close to the MFE structure[4]. However, neither suboptimal approach guarantees an unbiased representation of the entire Boltzmann-weighted ensemble. Furthermore, the PF variant of the complete suboptimal method, as proposed in [McC90], only provides statistics about the RNA secondary structures, but the algorithm itself does not solve the RNA folding problem, as no candidate foldings are generated.

The dilemma is that on the one hand, the presentation of suboptimal foldings through a designed set of moderate size is limited, but on the other hand, the complete enumeration and exploration of all suboptimal foldings is difficult and usually fundamentally inefficient. However, the generation and study of statistically representative sample sets of candidate foldings (secondary structures) may provide an efficient resolution to this dilemma, as suggested with prototype algorithms [DL99, Din02].

A corresponding secondary structure sampling algorithm that incorporates comprehensive structural features and the recent thermodynamic parameters has been devised a few years later by Ding and Lawrence [DL03]. This method essentially relies on PFs and can be used to generate a structurally diverse set of suboptimal foldings which – compared to the set of structurally quite similar suboptimal structures usually computed by MFE based DP algorithms – can be much closer to the structure determined by comparative analysis [Din06]. Hence, the algorithm described in [DL03] basically provides a sampling extension of the PF approach

[4]Precisely, the low-energy area of the unweighted energy landscape is addressed by this exact suboptimal method.

that effectively addresses the long-standing problem of a statistical representation of probable foldings.

Notably, the essence of the sampling algorithm is stochastic traceback. This means instead of choosing base pairs deterministically (as in all previously discussed algorithms), they are generated probabilistically in accordance with the PFs for all possible sequence fragments. Particularly, in contrast to its MFE counterpart, where base pairs are generated one at a time in the traceback step according to the energy minimization principle to form a particular suboptimal structure, the base pairs and unpaired base(s) that form a candidate structure are successively sampled in proportion to their Boltzmann weights, guaranteeing a statistical representation of the Boltzmann-weighted ensemble. In fact, the probability of sampling any particular secondary structure s for a given input sequence r is equal to its probability of occurring in the thermodynamic ensemble of all feasible structures for r and thus given by $\Pr(s \mid r)$, that is its Boltzmann probability.

3.2.4.1. Sampling Probabilities

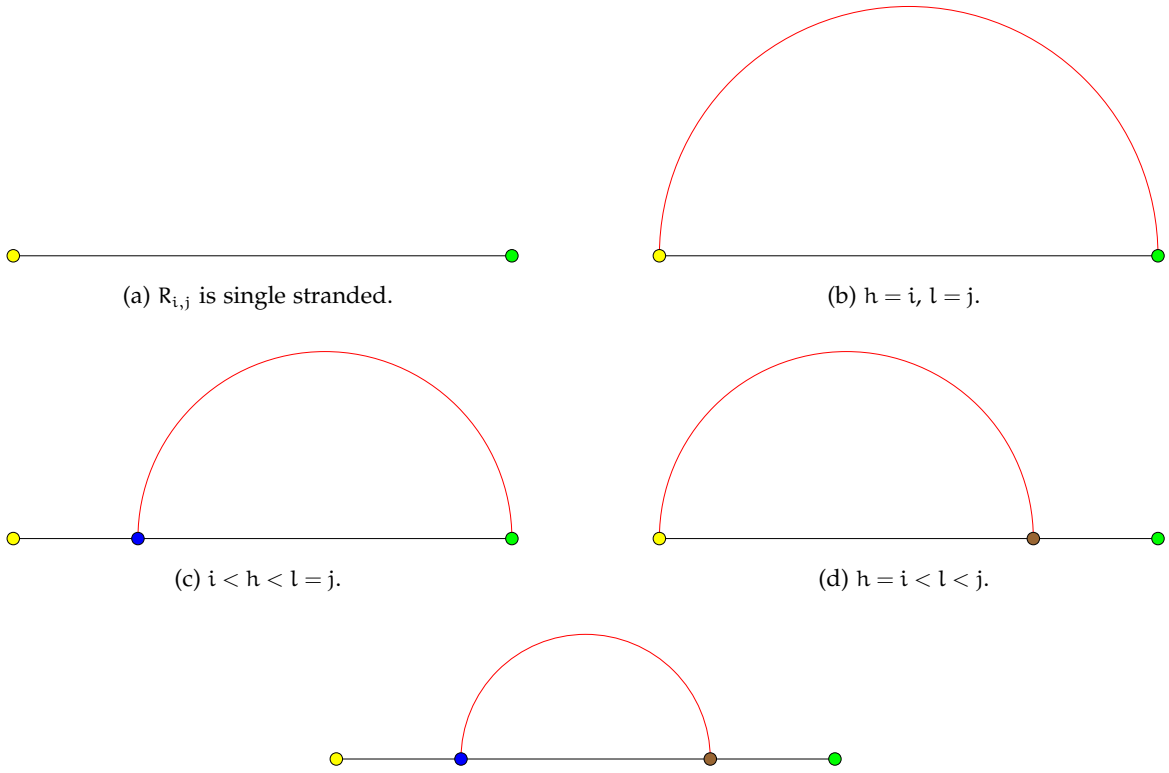
The task of structure sampling can easily be accomplished using a simple stacking model for free energies [DL99, Din02], but it is also possible to consider the comprehensive Turner parameters [DL03]. This is due to the fact that the recursions for restricted PFs (like ZP_r or ZM_r considered above) strongly correspond to *sampling probabilities* for mutually exclusive and exhaustive conformational cases according to the following relation:

$$\text{Sampling probability for case x} = \frac{\text{contribution of case x to PF Y}}{\text{PF Y}}.$$

Consequently, the base pairs and unpaired base(s) are in any case sampled according to *conditional* probability distributions for the considered fragment, given the partially formed structure. For both mathematical and algorithmic reasons, the base pairs are principally drawn in a two-stage process, that is by first sampling the $5'$ base and afterwards the corresponding $3'$ base, implying which bases are to be left single-stranded.

For instance, given a sequence fragment $R_{i,j} = r_i \dots r_j$, $1 \leqslant i, j \leqslant n$, of the input sequence $r = R_{1,n}$, which must be part of the exterior loop according to the partially formed structure (that is, it is not known whether the ends r_i and r_j form a pair), then we have to randomly sample one of the mutually exclusive and exhaustive cases illustrated in Figure 3.1. In order to guarantee that the random sampling of one of these conformational cases is in proportion to the underlying Boltzmann distribution for $R_{1,n}$, we draw one of these possible choices from the corresponding distribution conditioned on the considered sequence fragment $R_{i,j}$, which is characterized by the normalizing constant $Z_{i,j}$, the value of the PF for all feasible foldings of $R_{i,j}$. In accordance with the recursions for calculating $Z_{i,j}$, $1 \leqslant i, j \leqslant n$, we thus have to use the following sampling probabilities in order to draw one of the corresponding conformational cases [DL03]:

$$\begin{aligned}
P^E_0 &= \frac{1}{Z_{i,j}} \cdot exp\left(-\frac{e1_{i,j}}{RT}\right),\\
P^E_{ij} &= \frac{1}{Z_{i,j}} \cdot ZP_{i,j} \cdot exp\left(-\frac{e2_{i,j}}{RT}\right),\\
P^E_{hj} &= \frac{1}{Z_{i,j}} \cdot ZP_{h,j} \cdot exp\left(-\frac{e3_{i,j,h}}{RT}\right),\\
P^E_{il} &= \frac{1}{Z_{i,j}} \cdot \left(ZP_{i,l} \cdot exp\left(-\frac{e4_{i,j,l}}{RT}\right) \cdot Z_{l+2,j} + Z_{l+1,j} - Z_{l+2,j}\right),\\
P^E_{hl} &= \frac{1}{Z_{i,j}} \cdot s1_{h,j},
\end{aligned}$$



(a) $R_{i,j}$ is single stranded.

(b) $h = i$, $l = j$.

(c) $i < h < l = j$.

(d) $h = i < l < j$.

(e) $i < h < l < j$.

Figure 3.1.: **Conformational cases for an unrestricted sequence fragment.** Figures illustrate the different classes of mutually exclusive and exhaustive cases considered for a sequence fragment $R_{i,j}$ with the ends not being restricted to pair. Bases i and j are colored yellow and green, whereas h and l are colored blue and brown, respectively.

$$\widehat{P}^E_{hl} = \frac{1}{s1_{h,j}} \cdot \left(ZP_{h,l} \cdot \exp\left(-\frac{e5_{i,j,h,l}}{RT}\right) \cdot Z_{l+2,j} + Z_{l+1,j} - Z_{l+2,j} \right),$$

where

- P^E_0 is the sampling probability for case (a): $R_{i,j}$ is single stranded;
- P^E_{ij} is the sampling probability for case (b): $h = i$, $l = j$;
- $\{P^E_{hj}\}$ are the sampling probabilities for case (c): $i < h < l = j$;
- $\{P^E_{il}\}$ are the sampling probabilities for case (d): $h = i < l < j$;
- $\{P^E_{hl}\}$ are the probabilities for first sampling h for case (e): $i < h < l < j$; and
- $\{\widehat{P}^E_{hl}\}$ are the probabilities for sampling l after h is sampled.

Notably, $\{s1_{h,j}\}$ are auxiliary values needed to ensure that the computation of all these sampling probabilities for any fixed pair $i.j$ is linear (see [DL03] for details). Because the probabilities of all mutually exclusive and exhaustive cases sum up to 1, we have

$$P^E_0 + P^E_{ij} + \sum_{i<h<j} P^E_{hj} + \sum_{i<l<j} P^E_{il} + \sum_{i<h<j-1} P^E_{hl} = 1$$

and

$$\sum_{h<l<j} \widehat{P}^E_{hl} = 1.$$

If the ends r_i and r_j of a given sequence fragment $R_{i,j}$ are known to form a base pair (according to the partially folded structure), the type of loop closed by that pair $i.j$ can be sampled in proportion to the conditional sampling probabilities for the distinguished loop types. These probabilities are obviously based on the recursion for $ZP_{i,j}$ and are thus given by the following equations [DL03]:

$$
\begin{aligned}
Q_{ij}^{HL} &= \frac{1}{ZP_{i,j}} \cdot \exp\left(-\frac{eh_{i,j}}{RT}\right), \\
Q_{ij}^{SP} &= \frac{1}{ZP_{i,j}} \cdot \exp\left(-\frac{es_{i,j}}{RT}\right) \cdot ZP_{i+1,j-1}, \\
Q_{ij}^{BI} &= \frac{1}{ZP_{i,j}} \cdot \sum_{i<h<l<j} \exp\left(-\frac{ebi_{i,j,h,l}}{RT}\right) \cdot ZP_{h,l}, \\
Q_{ij}^{ML} &= \frac{1}{ZP_{i,j}} \cdot ZM_{i,j}
\end{aligned}
$$

where

- Q_{ij}^{HL} is the sampling probability for hairpin loop;
- Q_{ij}^{SP} is the sampling probability for base pair stack;
- Q_{ij}^{BI} is the sampling probability for a bulge or an interior loop; and
- Q_{ij}^{ML} is the sampling probability for a multi-branched loop.

Again, for mutually exclusive and exhaustive cases, we have

$$Q_{ij}^{HL} + Q_{ij}^{SP} + Q_{ij}^{BI} + Q_{ij}^{ML} = 1.$$

Conditional sampling probabilities for drawing base pairs within specific loop types can be derived accordingly, but the corresponding equations are mostly not quite as obvious as the presented ones (since the DP recursions for the respective PFs are in most cases more complicated, as mentioned above).

3.2.4.2. Overall Algorithm

Considering conditional sampling probabilities for mutually exclusive and exhaustive cases as described above, a secondary structure can be sampled in a straightforward recursive way. As proposed in [DL03], this can be done by starting with the entire RNA sequence and consecutively computing the adjacent substructures (single-stranded regions and paired substructures) of the exterior loop, where any paired substructure is completed by recursively sampling accessible substructures for each generated base pair. Notably, the sampling of substructures traditionally proceeds from left to right, in accordance with the definition of the corresponding sampling probabilities (or else, in compliance with the recursions for calculating the PFs). A schematic overview of the overall sampling process is given in Figure 3.2.

Example 3.2.1 *For an exemplary demonstration of the different steps of the sampling process, we decided to use dot-bracket representations of the partially formed structures, but with a slight modification. In particular, positions that have not yet been solved (that is, determined to be unpaired or paired) are represented by symbols* $*$*. Then, the recursive process of sampling the (correct) secondary structure of the* E.coli $tRNA^{Ala}$ *sequence (as given in Example 2.2.1) with the recursive strategy described by Figure 3.2 can be illustrated as follows:*

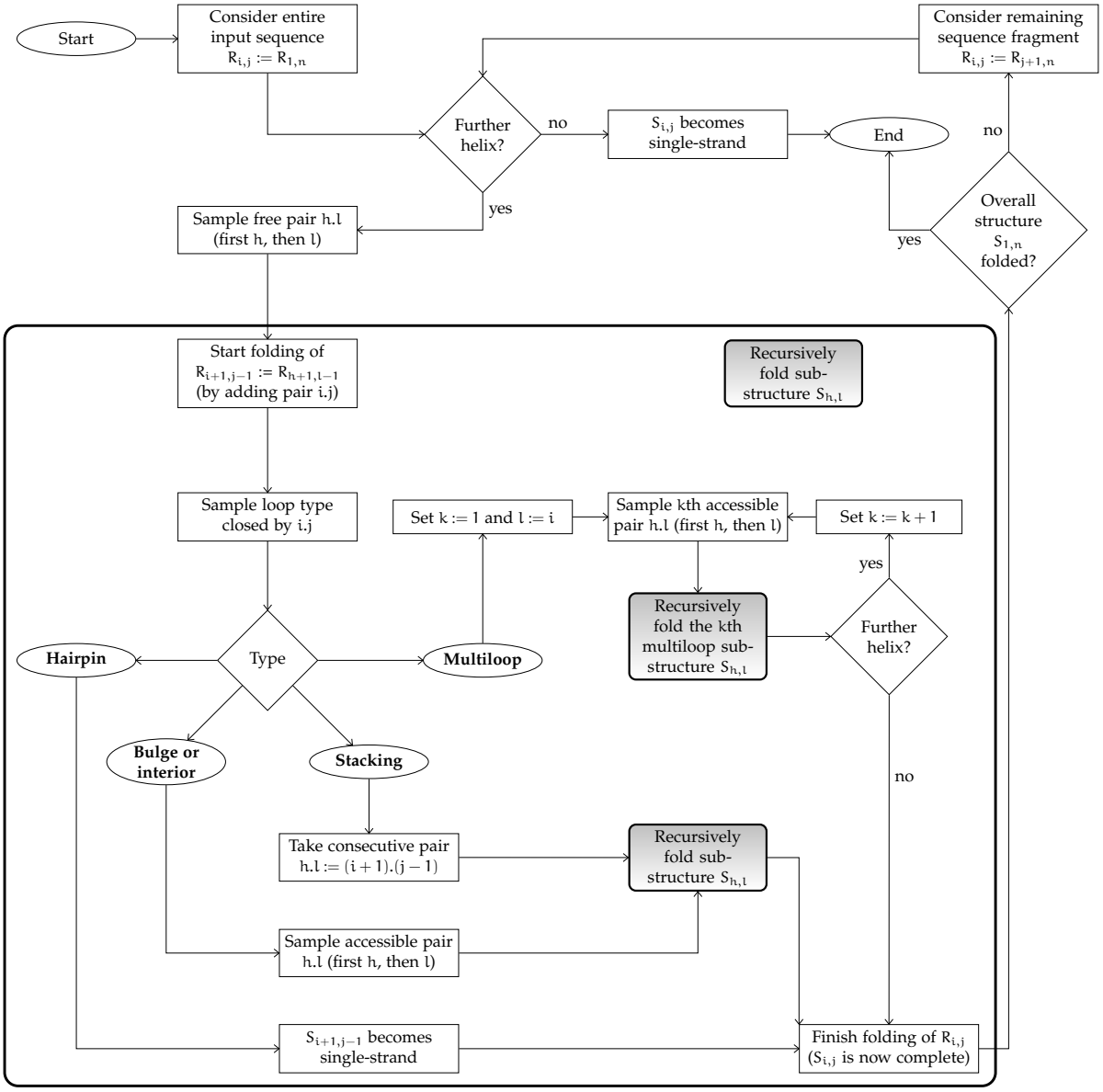


Figure 3.2.: **Flowchart for recursive sampling of an RNA secondary structure.** The flowchart describes the sampling of a complete secondary structure $s = S_{1,n}$ for a given input sequence $r = R_{1,n}$ of length n, in a quite similar fashion as proposed in [DL03].

- *Consider the overall sequence* $R_{1,n=76}$
  $\rightsquigarrow$ **********
- *Randomly draw new base pair on* $R_{1,76}$
  $\rightsquigarrow$ (**********)****
- *Extend helix and randomly choose* $S_{8,65}$ *to become a multiloop*
  $\rightsquigarrow$ (((((((**********)))))))****
- *Randomly draw external closing base pair of first multiloop helix*
  $\rightsquigarrow$ (((((((..(**********)**********)))))))****
- *Extend helix and randomly choose* $S_{14,21}$ *to become a hairpin loop*
  $\rightsquigarrow$ (((((((..((((........))))**********)))))))****
- *Randomly draw external closing base pair of second multiloop helix*
  $\rightsquigarrow$ (((((((..((((........))))).(**********)**********)))))))****
- *Extend helix and randomly choose* $S_{32,38}$ *to become a hairpin loop*
  $\rightsquigarrow$ (((((((..((((........)))).(((((.......)))))**********)))))))****
- *Randomly draw external closing base pair of third multiloop helix*
  $\rightsquigarrow$ (((((((..((((........)))).(((((.......))))).....(**********)))))))****
- *Extend helix and randomly choose* $S_{54,60}$ *to become a hairpin loop*
  $\rightsquigarrow$ (((((((..((((........)))).(((((.......))))).....(((((.......)))))))))))****
- *After finishing the construction of the multiloop, randomly choose* $R_{73,76}$ *to be unpaired*
  $\rightsquigarrow$ (((((((..((((........)))).(((((.......))))).....(((((.......)))))))))))....
- *Finish overall folding (no unsolved regions)!*

Note that when using this sampling variant, there is always only one unique way for sampling a particular structure, due to the fact that substructures are sampled from left to right.

Under the restriction that the length of single-stranded regions in bulges and interior loops is bounded by a constant value $\max_{bulge} \geqslant 1$, the complete statistical sampling method devised in [DL03] scales $\mathcal{O}(n^3)$ in time and requires $\mathcal{O}(n^2)$ storage, where the stochastic traceback scheme can actually be performed in $\mathcal{O}(n^2)$ time after the $\mathcal{O}(n^3)$ DP step for PF calculations.

It remains to mention that the statistical sampling algorithm from [DL03] has been implemented in the `Sfold` software package [DCL04], which besides others provides procedures for characterizing and visualizing the Boltzmann-weighted ensemble. Moreover, the current version of the `Sfold` web server provides a bunch of additional features, including clustering statistics, ensemble and cluster centroids, multi-dimensional scaling display and energy landscape representation of the Boltzmann-weighted ensemble [CLD05]. Note that `Sfold` actually predicts suboptimal foldings as *centroids* of *clusters* of candidate structures obtained from statistical sampling, rather than a number of suboptimal foldings derived from an MFE based DP traceback. However, if only the optimal (MFE) structure is needed, a strict DP variant should be preferred in terms of the running time.

3.2.4.3. Sample Size

As shown in [DL03], the sets of candidate structures generated by sampling rigorously from the Boltzmann distribution, the so-called *Boltzmann samples*, are indeed statistically representative even in case of rather moderate sample sizes (that is, numbers of sampled structures for a given input sequence). Moreover, characteristics of the Boltzmann ensemble revealed by sampling are statistically reproducible for independent samples of moderate size. However, it has not been described how to determine a generally valid, statistically sufficient sample size that can reliably be considered for any particular input sequence.

Typically, a sample size of 1000 is considered for applications (see, for instance [DL03, CLD05]), independent of the type, length or other characteristics of the input sequence that are crucial for the cardinality in the corresponding Boltzmann ensemble. As already pointed out in [Pon08],

this number of 1000 candidate structures is used in radically different contexts, without formulating an hypothesis on the distribution for their parameters[5].

However, due to the fact that the number of potential candidate structures grows exponentially with increasing sequence length, it would obviously be quite surprising if the same fixed moderate sample size that has been (experimentally) determined to be statistically sufficient for rather short RNA sequences would also be a reasonable choice for a longer input sequence, able to cover the larger set of alternative conformations for that sequence. For all practical purposes, it appears that for longer input sequences, larger sets of candidate structures might need to be generated by statistical sampling in order to avoid a decline in the accuracy of the corresponding method with respect to a particular application.

3.2.4.4. Impact

The impact of sampling approaches with respect to both RNA structure analysis and RNA folding is enormous. In fact, one of the main benefits of statistical sampling is that in contrast to the suboptimal structure sets computed by MFE approaches, the sets of generated candidate foldings are representative samples of the complete ensemble of feasible structures. This immediately allows for sampling estimates of the probabilities of any structural motifs, from the simplest elements of base pair and unpaired base, to loops of various types, to more complex structures consisting of stems and loops that may be of special interest in a given application. According to [Din06], a sample of secondary structures can also be used for computing other characteristics of the Boltzmann ensemble, for instance the mean and the variance of the free energy distribution can be estimated by a statistical sample rather than by using exact calculations as proposed in [IMN05] that require laborious algorithm development.

Anyway, as indicated earlier, statistical sampling can also be used in order to improve the accuracy of secondary structure predictions. Particularly, it has been shown that both the *ensemble centroid*[6] and the best *cluster centroid* are substantially more accurate than MFE structures [DCL05], a result that further validates ensemble based approaches. Note that the idea of producing a single structure which reflects the entire distribution of proposed foldings in a more proper way than the MFE structure (that is, the most probable structure according to the Boltzmann distribution, or in general according to any alternative probability distribution) has been further extended by studying a variety of *centroid estimators* in [DWB06, CL08, HKS+09].

Nevertheless, it should be mentioned that since the PF – and thus sampling based on it – is dependent on free energies, it is however also limited by the underlying thermodynamic model. In fact, as the most probable structures in the Boltzmann-weighted ensemble are equal to the MFE (or something close to it) structures, this approach inherits some of the problems associated with traditional MFE approaches.

3.2.5. Abstract Shapes Approach

Using a statistical sampling method or a corresponding suboptimal RNA folding algorithm for predicting the secondary structure of a single input sequence, the generated set of candidate or near-optimal foldings for that sequence usually contains many structures that are very similar and thus redundant, especially in the suboptimal case. However, for identifying the final

[5] Except for in a special case involving `RNAshapes` [VGR06], where the sample size 1000 is justified by assuming a Poisson distribution on the number of occurrences of each shape in a sample.

[6] According to [DCL05], the ensemble centroid is defined as the predicted secondary structure with the least total base pair distance to all the structures in the sample, where base pairing distance means the number of base pairs that differ between two structures. It hence describes the structure in the sample that is most representative of all the sampled foldings.

prediction, it may be sufficient to consider any subset of the proposed sample or suboptimal structure set that consists only of foldings with fundamental structural differences.

Abstract shape analysis is a fairly recent attempt in this direction, since any shape indeed describes a class of similar structures (see Section 2.2.3). Furthermore, within each abstract shape class, the secondary structure with lowest free energy is defined as the *shape representative structure (shrep)*, such that we can find the final structures among the top shape representatives. Notably, shape abstraction integrates well with DP algorithms, meaning it can readily be applied during the folding process rather than afterwards in order to avoid exponential explosion of proposed suboptimals.

Based on this idea, an integrated RNA analysis software package named `RNAshapes` has been developed [SVR+06b], which besides others implements the abstract shape variant of all physics-based approaches discussed above. This means it can be used to compute either a single shape and corresponding shrep or a small set of representative structures of different shapes based on the MFE paradigm (see [GVR04]), in analogy to the standard DP methods described in [ZS81, Zuk89b]. Furthermore, one can calculate shape probabilities based on the PF approach [McC90], where the probability of a shape is the sum of the probabilities of all structures that fall into this shape (see [VGR06]). Statistical sampling based on PFs is also implemented in `RNAshapes`, combining the sampling method from [DL03] with a posteriori shape abstraction.

3.2.6. Summary

For a very long time, the MFE paradigm has been the most common technique for predicting the secondary structure of a single RNA sequence. The respective methods are traditionally realized by DP algorithms that employ a particular thermodynamic model for the derivation of the corresponding recursions. They basically require $\mathcal{O}(n^3)$ time and $\mathcal{O}(n^2)$ storage for identifying a set of candidate structures for an input sequence of length n.

In fact, while early methods, like [NPGK78, NJ80, ZS81], computed only one structure (the MFE structure of the molecule), several more elaborate MFE based DP algorithms have been developed over the years for generating a set of suboptimal foldings (for instance [WFHS99, Zuk89b]). Some implementations are considered state-of-the-art tools for computational structure prediction from a single sequence, for example the `Mfold` software [Zuk89b, Zuk03] or the `ViennaRNA` package [HFS+94, Hof03].

However, one major drawback of these MFE approaches is that in the traceback steps of the corresponding DP algorithms, base pairs are successively generated according to the energy minimization principle, such that the predicted set of suboptimal foldings often contains many structures that are not significantly different (that have the same or very similar shapes and contain mostly the same actual base pairings).

To overcome these problems, several statistical sampling methods and clustering techniques have been invented over the last years that are based on the PF approach for computing base pair probabilities as devised by McCaskill [McC90]. Briefly, these methods produce a statistical sample of the thermodynamic ensemble of suboptimal foldings and rely on a statistical representation of the Boltzmann-weighted ensemble of structures for a given sequence [DL03]. They are implemented in the `Sfold` package [DCL04].

Anyway, common to all discussed methods is that their performance is strongly dependent on and thus limited by the applied thermodynamic model. In fact, they generally build on the standard energy model as proposed by Turner [XSB+98, MSZT99], which still contains many imprecisions and uses the same experimentally derived parameters for all RNA types, such that particular class specific properties can not be accounted for in corresponding prediction algorithms.

3.3. Methods Based on Specific Structural Information

Due to the problems even of comprehensive state-of-the-art thermodynamic models to capture some specific information buried in biological sequences, an attractive alternative is to use statistical approaches with parameters estimated from growing databases of structural RNAs. This actually motivated the development of a competing methodology towards computational RNA structure prediction analysis that builds on principles of *probabilistic modeling* of the class of possible foldings rather than on incomplete free energy models.

3.3.1. Probabilistic Models

Basically, a probabilistic model produces different outcomes with different probabilities and can thus simulate a whole class of objects, where each object is assigned an associated probability. In connection with RNA folding, the objects are traditionally RNA sequences with underlying secondary structures, and a corresponding model typically describes a family of related sequences and their respective feasible foldings.

Since the numbers of described sequences are infinite and there usually results a combinatorial explosion of possible structures for a single sequence, it is actually impossible the enumerate all these possibilities in order to obtain a corresponding probability distribution. In order to build a correct probabilistic model, we therefore need a formal system, usually a specific *stochastic context-free grammar (SCFG)* or more powerful variants, like for instance a corresponding *length-dependent stochastic context-free grammar (LSCFG)* or *conditional log-linear model (CLLM)*.

Briefly, SCFGs are an extension to the concept of traditional context-free grammars (CFGs) which does not only model the class of objects (language) to be generated, but also defines a (usually non-uniform) probability distribution on its elements. Just like in connection with considering the Boltzmann distribution of low-energy, corresponding prediction methods based on context-free modeling are traditionally realized by DP routines that run in $\mathcal{O}(n^3)$ time and require $\mathcal{O}(n^2)$ storage for an input sequence of length n. However, in contrast to physics-based methods which in general rely on a comprehensive thermodynamic model, the corresponding DP recursions are constructed in accordance with a specific (more or less complex) probabilistic model, making use of structural information obtained from reliable RNA databases.

Principally, the SCFG approach can be seen as a generalization of *hidden Markov models (HMMs)*, which are widely and successfully used in the large field of bioinformatics[7]. In fact, it is worth mentioning that by using SCFG theory, it becomes possible to tackle some questions of RNA secondary structure analysis that can actually not be handled with analogue HMMs or other primary sequence based approaches, including RNA secondary structure prediction, structure based alignment of RNAs, and structure based database search for homologous RNAs (see, for instance [DEKM98]).

3.3.2. Parameter Estimation

Essentially, any probabilistic model builds an a particular (more or less comprehensive) set of *probabilistic parameters* for distinguished features that induce the corresponding probability distribution on the considered objects. The parameters for a probabilistic model are typically estimated from large sets of trusted sample data, the so-called *training set*, where there exist several mathematically well-defined and widely accepted training approaches.

[7] A basic introduction to HMMs can be found, for instance in [RJ86, Kro98]. For more information on their use in bioinformatics, we refer to [DEKM98].

3.3.2.1. Maximum Likelihood Training

One popular method for estimating the parameters of a probabilistic model is to use the *maximum likelihood (ML)* method which was invented by R. A. Fisher between 1912 and 1922 [Ald97]. Intuitively, this technique works as follows: On a fixed sample from a larger population, the free parameters of the underlying probabilistic model are tuned in such a way that the sample has maximum *likelihood*, which is defined as the total probability of all the objects given the considered model. This means that other values for the free parameters make the observation of the sample less likely. Formally, given a model with parameters Φ and a set of data $\mathcal{D}$, the ML estimate for Φ is that value which maximizes the probability $\Pr(\mathcal{D} \mid \Phi)$. Details will follow in Section 3.3.6.3.

However, we want to make clear already at this point that in non-technical parlance, the terms likelihood and probability may be used synonymously, but in statistical usage, a clear technical distinction is made: Given specific values for Φ, then $\Pr(\mathcal{D} \mid \Phi)$ is the probability that we would observe the data represented by $\mathcal{D}$. However, we do not know Φ. We simply observe $\mathcal{D}$ and the goal is to arrive at an estimate for Φ that would be a plausible choice given the observed data $\mathcal{D}$. Thus, a natural estimation process is to choose that value of Φ that would maximize the probability that we would actually observe $\mathcal{D}$. In other words, we find the parameter values Φ that maximize the likelihood function $\ell(\Phi \mid \mathcal{D}) := \Pr(\mathcal{D} \mid \Phi)$, which by definition is conditioned on the observed data $\mathcal{D}$ and is actually a function of the unknown parameters Φ.

Anyway, note that the ML estimates of any considered feature generally equate to the corresponding relative frequencies observed in the same database of known objects. Counting the relative frequencies of distinguished features on the basis of trusted sample data is actually a widely-used approach. Under the condition that the data set is large enough and that the training data are not systematically biased towards a peculiar structural composition, these frequencies may be expected to represent reasonable estimates of the probabilities underlying the model. In fact, the relative frequencies of any distinguished feature as observed in a particular database then indeed maximize the likelihood.

Finally, it needs to be mentioned that when estimating parameters for a model from a limited amount of data, there is a danger of *overfitting*. In statistics, the term overfitting is used to express that a statistical model describes random error or noise instead of the underlying relationship, caused by the fact that there are too many parameters relative to the number of observations. Hence, in our context overfitting potentially occurs when the number of probabilistic parameters to be estimated is too large for the training data, that is if the considered training data are not rich enough to reliably estimate the distinct parameters.

3.3.2.2. Bayesian Parameter Estimation

Another popular approach for deriving the parameters of a probabilistic model is known as *Bayesian parameter estimation*, as it relies on *Bayes' theorem* which is fundamental to Bayesian statistics and is widely applied in numerous fields. Notably, Bayesian parameter estimation is often used in cases where there are not enough data to reliably estimate (at least some of) the model parameters, as it enables the convenient use of prior knowledge to constrain the parameter estimates in order to prevent overfitting.

Formally, let Φ denote a set of parameters to be estimated (that is, the unobserved parameters) and let $\mathcal{D}$ denote a particular training set (that is, the observed data). Then, the *prior probability* of Φ, $\Pr(\Phi)$, reveals what is known about the parameters without the knowledge of the data. Furthermore, as in classical statistics, let $\Pr(\mathcal{D} \mid \Phi)$ denote the *likelihood* based on the corresponding model. Thus, using Bayes' theorem, we can calculate the *posterior probability* of the set of parameters Φ given knowledge on the data $\mathcal{D}$ according to the following equation[8]:

[8]Note that the symbol $\propto$ is used here to mark the proportional relation.

$$\Pr(\Phi \mid \mathcal{D}) = \frac{\Pr(\Phi)\Pr(\mathcal{D} \mid \Phi)}{\Pr(\mathcal{D})} \propto \Pr(\Phi)\Pr(\mathcal{D} \mid \Phi), \tag{3.3}$$

where

$$\Pr(\mathcal{D}) = \int_{\Phi'} \Pr(\Phi')\Pr(\mathcal{D} \mid \Phi'),$$

that is we integrate over the parameter space (over all possible values for Φ), since the parameters are usually continuous rather than discrete quantities. Note that $\Pr(\mathcal{D})$ is obviously independent of the specific value of Φ. In this context, it should also be mentioned that the application of Bayes' theorem to update beliefs (or, in the context of parameter estimation, to re-estimate parameter values) is called *Bayesian inference*.

Furthermore, note that one typically computes the parameter values for Φ that maximize the posterior probability $\Pr(\Phi \mid \mathcal{D})$. This approach is actually known as *maximum a posteriori (MAP)* estimation. Due to the independence of $\Pr(\mathcal{D})$ on the specific value of Φ, MAP estimation corresponds to maximizing the likelihood $\Pr(\mathcal{D} \mid \Phi)$ times the prior $\Pr(\Phi)$, see equation (3.3). Note that if $\Pr(\Phi)$ is flat, then MAP estimation is the same as ML estimation.

It remains to mention that several related Bayesian statistics approaches also exist. For instance, rather than the maximum value, one might compute the mean of the posterior distribution as the estimate, which however requires that the posterior probability can either be calculated analytically or can be sampled. Nevertheless, due to the subjectiveness of issues like the choice of prior, Bayesian methods are not always the preferred framework for parameter estimation. In connection with training large parameter sets from small amounts of data, however, Bayesian methods provide a consistent formalism for bringing in additional information from previous experience with the same type of data (see [DEKM98]).

For more information on Bayesian approaches as part of a field of statistics, but also on Bayesian methods applied to a wide range of problems in bioinformatics and genomics, we refer to [BT92, GCSR97, DEKM98, BR04]. Other textbooks dealing with statistical inference approaches in general include [Jay03, Mac03].

3.3.3. Overview

The aim of this section is to introduce all the needed formalisms concerning SCFGs as mathematical objects for capturing biological sequence information, as well as to provide some additional information on existing SCFG based probabilistic approaches in computational structural biology. Note that in the sequel, we call an approach *probabilistic* if it makes no use of free energy based models. Hence, even if a PF based Boltzmann sample is a random event, we accordingly do not assume it to be probabilistic.

We start with a somewhat informal introduction to context-free grammars and languages as (non-stochastic) models of RNA structure, describing only the basic concepts and properties. For more formal and detailed information on CFGs, we refer to one of the many existing excellent textbooks on this topic, for instance [HU79, Har78, Sip96]. Then, we give a self-contained overview of probabilistic modeling of classes of combinatorial objects described by corresponding CFGs, with a special emphasis on RNA secondary structures. In fact, we first present the basic formal definitions and some aspects of SCFGs that are relevant in the context of this thesis, and specifically as they apply to the problem of RNA folding. Furthermore, we introduce and discuss some details on computational approaches towards structure prediction that rely on probabilistic RNA models, specifically on SCFGs. For a fundamental introduction on stochastic context-free languages in general, we refer to [HF71]. A review of the use of SCFGs for RNA folding can be found, for example in [DEKM98].

By the end of this section, we give a brief introduction to both previously mentioned (more powerful) variants of SCFGs, that is LSCFGs and CLLMs, where if needed we also describe how the corresponding computational methods connected to RNA structure prediction, especially the training approaches, differ from the traditional SCFG based algorithms. Finally, we briefly discuss some related statistical approaches for estimating thermodynamic parameters in order to improve the predictive accuracy of physics-based methods for RNA folding.

3.3.4. Context-Free Grammars and Languages

The formalism of CFGs was developed in the mid-1950s by Noam Chomsky [Cho56], who also described their classification as a special type of formal grammar (which he called *phrase-structure grammars*).

In principle, a CFG is a finite set of variables, called *nonterminal symbols*, *intermediate symbols* or simply *intermediates*. Each variable represents a class of strings, where any string is composed of a finite number of distinct primitive symbols, called *terminal symbols* or *terminals*. The variable representing the class with maximum cardinality among all other classes represented by variables of the grammar is called *start variable* or *axiom*. Moreover, string classes represented by the variables of a CFG are called *context-free languages (CFLs)*. They are described recursively in terms of each other, where rules relating the variables are called *production rules*, *productions* or simply *rules*.

3.3.4.1. Formal Definitions

In order to introduce the basic formal framework concerning CFGs, we may rely on any textbook providing a discussion of formal language theory. Here, we basically adhere to [HU79], but we decided to use different names for the specific components:

Definition 3.3.1 *A* context-free grammar (CFG) *is a* 4*-tuple* $\mathcal{G} = (I, T, R, S)$*, where*

- I *is a finite set of* intermediate *or* nonterminal *symbols,*
- T *is a finite set of* terminal *symbols called* alphabet*,*
- $R \subset I \times (I \cup T)^*$ *is a finite set of production rules, and*
- $S \in I$ *is a distinguished intermediate symbol called* axiom*.*

Additionally, I *and* T *are required to be disjoint. In the sequel, we will write* $f = A \to \alpha$ *instead of* $f = (A, \alpha) \in R$*.*

The *language generated* by a CFG $\mathcal{G}$ is defined as the class of strings represented by the axiom of $\mathcal{G}$, and is denoted by $\mathcal{L}(\mathcal{G})$; the empty string will be denoted by ϵ in the sequel. Note that in literature, one sometimes calls the left-hand side $A \in I$ of a production $A \to \alpha$ the *premise* of that rule, whereas the corresponding right-hand side $\alpha \in (I \cup T)^*$ is then called the *conclusion*. Traditionally, a word $\alpha \in (I \cup T)^*$ is called *sentential form*.

3.3.4.2. Construction of Grammars and Language Specification

A particular CFG can be constructed to describe a number of classical constraints (for instance, the presence of particular motifs in secondary structures). It can also express long-range interaction, that is hierarchically nested, pairwise correlations (for example, those induced by base pairings in secondary structures).

Example 3.3.1 *A fairly simple CFG, call it $\mathcal{G}_{e,1}$, with only one nonterminal symbol S representing the class of all dot-bracket strings related to secondary structures (under the assumptions of $\min_{HL} = 0$ and $\min_{hel} = 1$)[9] is described by the following production rules:*

$$S \to \circ S, \quad S \to S\circ, \quad S \to (S), \quad S \to SS, \quad S \to \epsilon.$$

Basically, the first two rules generate unpaired bases, the third produces base pairs, the fourth bifurcations and the last one is needed to guarantee that the string construction terminates.

Note that rather different sets of variables and production rules, that is grammar designs, can be used to model the same or quite similar classes of strings:

Example 3.3.2 *An alternative CFG, say $\mathcal{G}_{e,2}$, for modeling secondary structures in dot-bracket representation is defined by the following productions (see [KH99]):*

$$\begin{aligned} S &\to LS, \quad S \to L, \\ L &\to \circ, \quad L \to (F), \\ F &\to (F), \quad F \to LS. \end{aligned}$$

Briefly, the intermediate symbol S (axiom) initiates the generation of single-stranded bases and/or helices of the exterior loop. Symbol L actually decides which substructure (single free base or complete helix starting with a corresponding free base pair) is created. Finally, intermediate symbol F recursively generates substructures in between a given base pair, since in any case it either extends a particular ladder (by generating an additional base pair) or finishes its construction (by initializing a new loop). Note that grammar $\mathcal{G}_{e,2}$ implicitly assumes minimum allowed lengths of $\min_{HL} = 2$ and $\min_{hel} = 1$ for hairpin loops and helices, respectively.

In a quite similar fashion, recursive specifications of CFLs can be derived as detailed formal characterizations of classes of complex combinatorial objects. For example, the combinatorial class of all RNA secondary structures without pseudoknots typically considered by physics-based folding algorithms (precisely, that meet the stereochemical constraint of hairpin loops consisting of at least $\min_{HL} = 3$ unpaired nucleotides and that may not be completely unpaired) can be modeled by the following CFL:

Definition 3.3.2 ([NS11a]) *The language $\mathcal{L}$ containing exactly all RNA secondary structures is given by $\mathcal{L} := \mathcal{L}_u\mathcal{L}_{lu}^+$, where $\mathcal{L}_{lu} := (\mathcal{L}_l)\mathcal{L}_u$, $\mathcal{L}_u := \{\circ\}^*$ is the language of all dot-bracket representations of single-stranded regions and $\mathcal{L}_l$ is the language of all dot-bracket representations of other possible substructures, that is, is the smallest language satisfying the following conditions:*

1. *$\{\circ\}^+ \setminus \{\circ, \circ\circ\} \subset \mathcal{L}_l$ (dot-bracket representations of* hairpin loops*).*
2. *If $w \in \mathcal{L}_l$, then $(w) \in \mathcal{L}_l$ (dot-bracket representation of a* stacked pair*).*
3. *If $w \in \mathcal{L}_l$, then $\{\circ\}^+(w) \subset \mathcal{L}_l$ and $(w)\{\circ\}^+ \subset \mathcal{L}_l$ (dot-bracket representations of* bulge loops*).*
4. *If $w \in \mathcal{L}_l$, then $\{\circ\}^+(w)\{\circ\}^+ \subset \mathcal{L}_l$ (dot-bracket representations of* interior loops*).*
5. *If $w_1, \ldots, w_n \in \mathcal{L}_l$ and $n \geqslant 2$, then $\mathcal{L}_u(w_1)\mathcal{L}_u(w_2)\cdots\mathcal{L}_u(w_n)\mathcal{L}_u \subset \mathcal{L}_l$ (dot-bracket representations of* multi-branched loops*).*

Note that we will make use of this recursive formal language definition of the corresponding class of secondary structures in the sequel, particularly in Chapter 5.

[9] Note that by definition, secondary structures must admit a minimum hairpin loop size of $\min_{HL} \geqslant 1$, such that this CFG $\mathcal{G}_{e,1}$ does actually only model a secondary structure related class. However, it has anyhow been used in literature for describing RNA secondary structure, such as for example in [DE04], which is why we will consider it here, too.

3.3.4.3. Derivation and Ambiguity

By definition, all strings of terminal symbols that can be generated from the start symbol of a given grammar $\mathcal{G}$ constitute the language $\mathcal{L}(\mathcal{G})$ specified by that grammar. Accordingly, any grammar $\mathcal{G}$ can be used as a language specification mechanism by generating each string of the corresponding language $\mathcal{L}(\mathcal{G})$ in the following "top-down" manner:

1. Start with the first variable (axiom) representing the complete language.
2. Find a variable in current sentential form and a rule describing the relations of that variable and replace it with the relation.
3. Repeat step 2 until no variables remain in the generated string (until it is either equal to the empty string ϵ or consists exclusively of terminals).

The sequence of substitutions utilized to obtain a string $w \in \mathcal{L}(\mathcal{G})$ using the CFG $\mathcal{G}$ is called a *derivation* (or sometimes also *parse*). If in any step of the derivation process, the leftmost (rightmost) variable is substituted, then the corresponding derivation is called *leftmost derivation* (*rightmost derivation*). In this thesis, we will only consider leftmost derivations. Note that by convention, one uses the symbol $\Rightarrow$ for immediate derivation (or sometimes $\overset{rule}{\Rightarrow}$ to explicitly mark the production rule *rule* considered for the corresponding substitution), and $\Rightarrow^+$ for a series of at least one consecutive derivation steps. Accordingly, $\Rightarrow^*$ is used for an arbitrary number (zero or more) of consecutive derivation steps.

Anyway, derivations can also be represented as ordered trees, where each node represents an intermediate symbol, a terminal symbol or the empty string ϵ. More precisely, inner nodes always correspond to nonterminals and the labels of their children – in left-to-right order – have to form the right-hand side (conclusion) of a rule for the parent (the corresponding premise). The corresponding trees are conventionally named *derivation trees* or *parse trees*. There actually exists a bijection between leftmost derivations and derivation trees.

Example 3.3.3 *The derivation of the dot-bracket string* $((\circ\circ\circ))\circ$ *using grammar* $\mathcal{G}_{e,2}$ *as specified in Example 3.3.2 is given by*

$$\begin{aligned} S &\overset{S\to LS}{\Rightarrow} LS \overset{L\to(F)}{\Rightarrow} (F)S \overset{F\to(F)}{\Rightarrow} ((F))S \overset{F\to LS}{\Rightarrow} ((LS))S \overset{L\to\circ}{\Rightarrow} ((\circ S))S \overset{S\to LS}{\Rightarrow} ((\circ LS))S \\ &\overset{L\to\circ}{\Rightarrow} ((\circ\circ S))S \overset{S\to L}{\Rightarrow} ((\circ\circ L))S \overset{L\to\circ}{\Rightarrow} ((\circ\circ\circ))S \overset{S\to L}{\Rightarrow} ((\circ\circ\circ))L \overset{L\to\circ}{\Rightarrow} ((\circ\circ\circ))\circ. \end{aligned}$$

The corresponding parse tree is shown in Figure 3.3.

In many applications, for instance in connection with enumeration problems, it is crucial that each structure is generated only once by a corresponding CFG. This means that there must be only *one* unique parse tree for any string of the CFL, or equivalently only one leftmost derivation. A CFG $\mathcal{G}$ where some string in $\mathcal{L}(\mathcal{G})$ has more than one parse trees (or leftmost derivations) is said to *ambiguous*.

Example 3.3.4 *Using the CFG* $\mathcal{G}_{e,1}$ *introduced in Example 3.3.1, the string* $((\circ\circ\circ))\circ$ *has more than one leftmost derivations, for instance:*

$$\begin{aligned} d_1 = S &\overset{S\to SS}{\Rightarrow} SS \overset{S\to(S)}{\Rightarrow} (S)S \overset{S\to(S)}{\Rightarrow} ((S))S \overset{S\to\circ S}{\Rightarrow} ((\circ S))S \overset{S\to\circ S}{\Rightarrow} ((\circ\circ S))S \overset{S\to\circ S}{\Rightarrow} ((\circ\circ\circ S))S \\ &\overset{S\to\epsilon}{\Rightarrow} ((\circ\circ\circ))S \overset{S\to\circ S}{\Rightarrow} ((\circ\circ\circ))\circ S \overset{S\to\epsilon}{\Rightarrow} ((\circ\circ\circ))\circ, \end{aligned}$$

or

$$d_2 = S \overset{S\to S\circ}{\Rightarrow} S\circ \overset{S\to(S)}{\Rightarrow} (S)\circ \overset{S\to(S)}{\Rightarrow} ((S))\circ \overset{S\to S\circ}{\Rightarrow} ((S\circ))\circ \overset{S\to S\circ}{\Rightarrow} ((S\circ\circ))\circ \overset{S\to S\circ}{\Rightarrow} ((S\circ\circ\circ))\circ \overset{S\to\epsilon}{\Rightarrow} ((\circ\circ\circ))\circ,$$

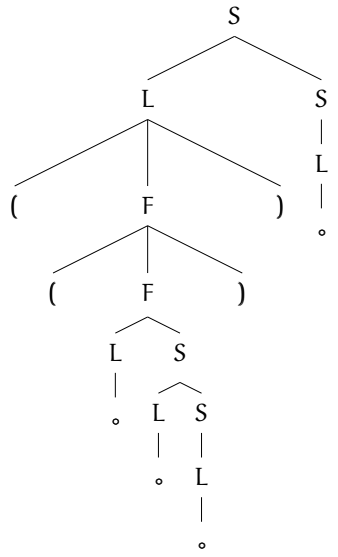


Figure 3.3.: **A simple derivation tree.** Figure shows the unique derivation tree of the dot-bracket word $((\circ\circ\circ))\circ$ using grammar $\mathcal{G}_{e,2}$.

or else

$$\begin{aligned} d_3 = S &\overset{S\to SS}{\Rightarrow} SS \overset{S\to(S)}{\Rightarrow} (S)S \overset{S\to(S)}{\Rightarrow} ((S))S \overset{S\to SS}{\Rightarrow} ((SS))S \overset{S\to SS}{\Rightarrow} ((SSS))S \overset{S\to\circ S}{\Rightarrow} ((\circ SSS))S \\ &\overset{S\to\epsilon}{\Rightarrow} ((\circ SS))S \overset{S\to\circ S}{\Rightarrow} ((\circ\circ SS))S \overset{S\to\epsilon}{\Rightarrow} ((\circ\circ S))S \overset{S\to SS}{\Rightarrow} ((\circ\circ SS))S \overset{S\to\circ S}{\Rightarrow} ((\circ\circ\circ SS))S \\ &\overset{S\to\epsilon}{\Rightarrow} ((\circ\circ\circ S))S \overset{S\to SS}{\Rightarrow} ((\circ\circ\circ SS))S \overset{S\to\epsilon}{\Rightarrow} ((\circ\circ\circ S))S \overset{S\to\epsilon}{\Rightarrow} ((\circ\circ\circ))S \overset{S\to SS}{\Rightarrow} ((\circ\circ\circ))SS \\ &\overset{S\to\circ S}{\Rightarrow} ((\circ\circ\circ))\circ SS \overset{S\to\epsilon}{\Rightarrow} ((\circ\circ\circ))\circ S \overset{S\to SS}{\Rightarrow} ((\circ\circ\circ))\circ SS \overset{S\to\epsilon}{\Rightarrow} ((\circ\circ\circ))\circ S \overset{S\to\epsilon}{\Rightarrow} ((\circ\circ\circ))\circ. \end{aligned}$$

The corresponding parse trees are presented in Figure 3.4. Another one is shown in Figure 3.5, illustrating the differences in the degree of redundancy of some derivations characterized by the number of useless applications of production rules. Redundancy is actually caused by sequences of substitutions made with $S \to SS$ in connection with $S \to \epsilon$ (see Figures 3.4c and 3.5). Moreover, considering the most reasonable (since comparably small) parse trees displayed in Figures 3.4a and 3.4b, it seems that either $S \to \circ S$ or $S \to S\circ$ represents a redundant production.

Accordingly, if for every word $w \in \mathcal{L}(\mathcal{G})$, exactly *one* leftmost derivation exists, then the CFG $\mathcal{G}$ is called *unambiguous*.

3.3.4.4. Other Aspects

Two CFGs $\mathcal{G}_1$ and $\mathcal{G}_2$ are called *(word-)equivalent* iff $\mathcal{L}(\mathcal{G}_1) = \mathcal{L}(\mathcal{G}_2)$. Furthermore, we call a CFG $\mathcal{G} = (I, T, R, S)$ *loop-free* iff there is no derivation of the form $A \Rightarrow^+ A$ for any $A \in I$. It is called ϵ*-free* iff there exists no $(A, \epsilon) \in R$ with $A = S$ and there exists no $(A, \alpha_1 S \alpha_2) \in R$. If there does not even exist an intermediate symbol $A \in I$ such that $A \Rightarrow A\alpha$, for $\alpha \in (I \cup T)^*$, we call $\mathcal{G}$ *free of left-recursion*.

Every CFG $\mathcal{G}$ that does not generate the empty string can be transformed into a word-equivalent CFG $\mathcal{G}'$ in which no rule has the empty string ϵ as conclusion. Furthermore, any CFG $\mathcal{G}$ without ϵ-rules has an equivalent grammar $\mathcal{G}'$ in *Chomsky normal form (CNF)* or *Greibach normal form (GNF)*. For details, we refer to [HU79].

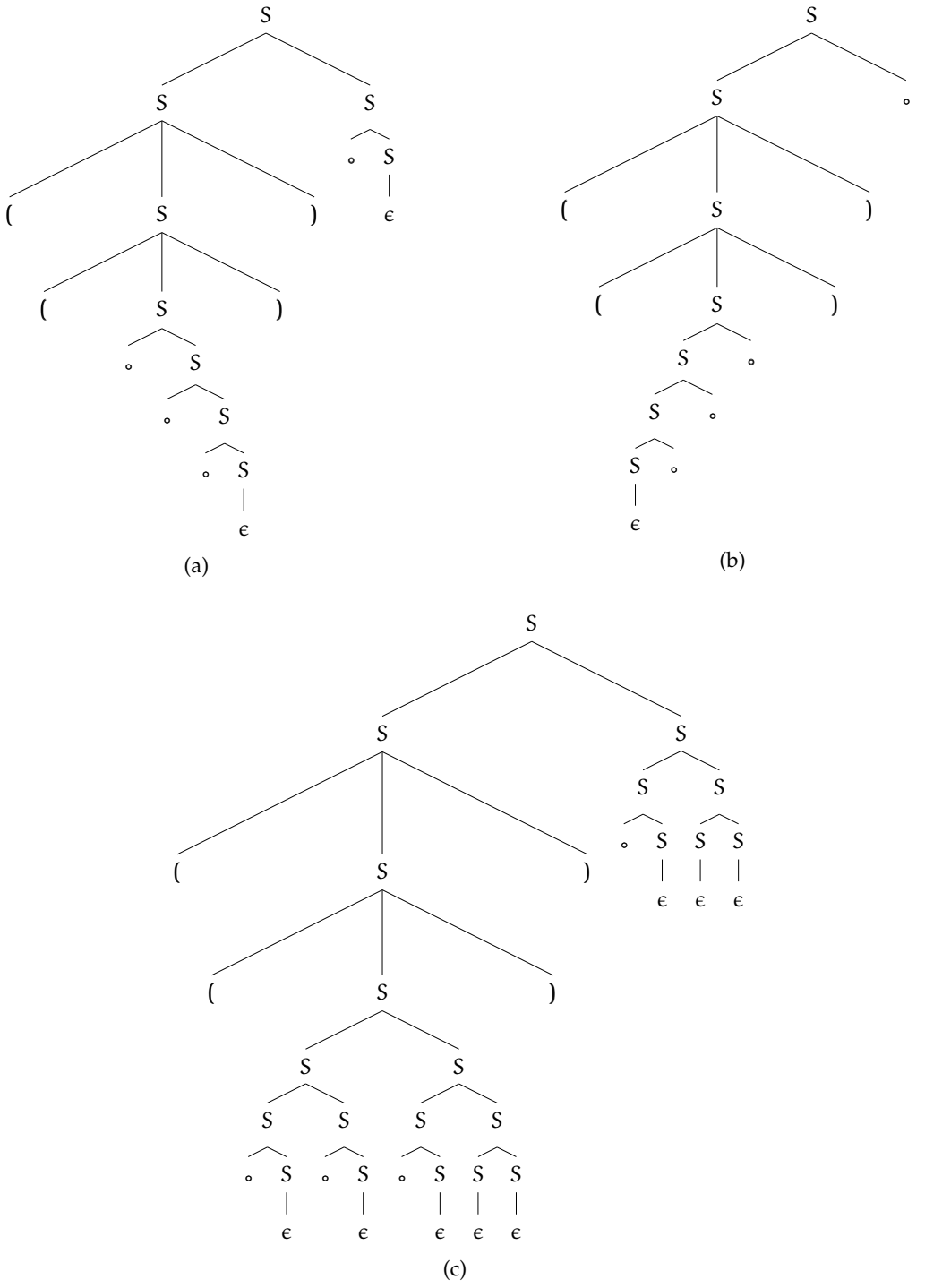


Figure 3.4.: **Derivation trees of different complexities.** Figures show three alternative derivation trees of the simple string $((\circ\circ\circ))\circ$ using grammar $\mathcal{G}_{e,1}$.

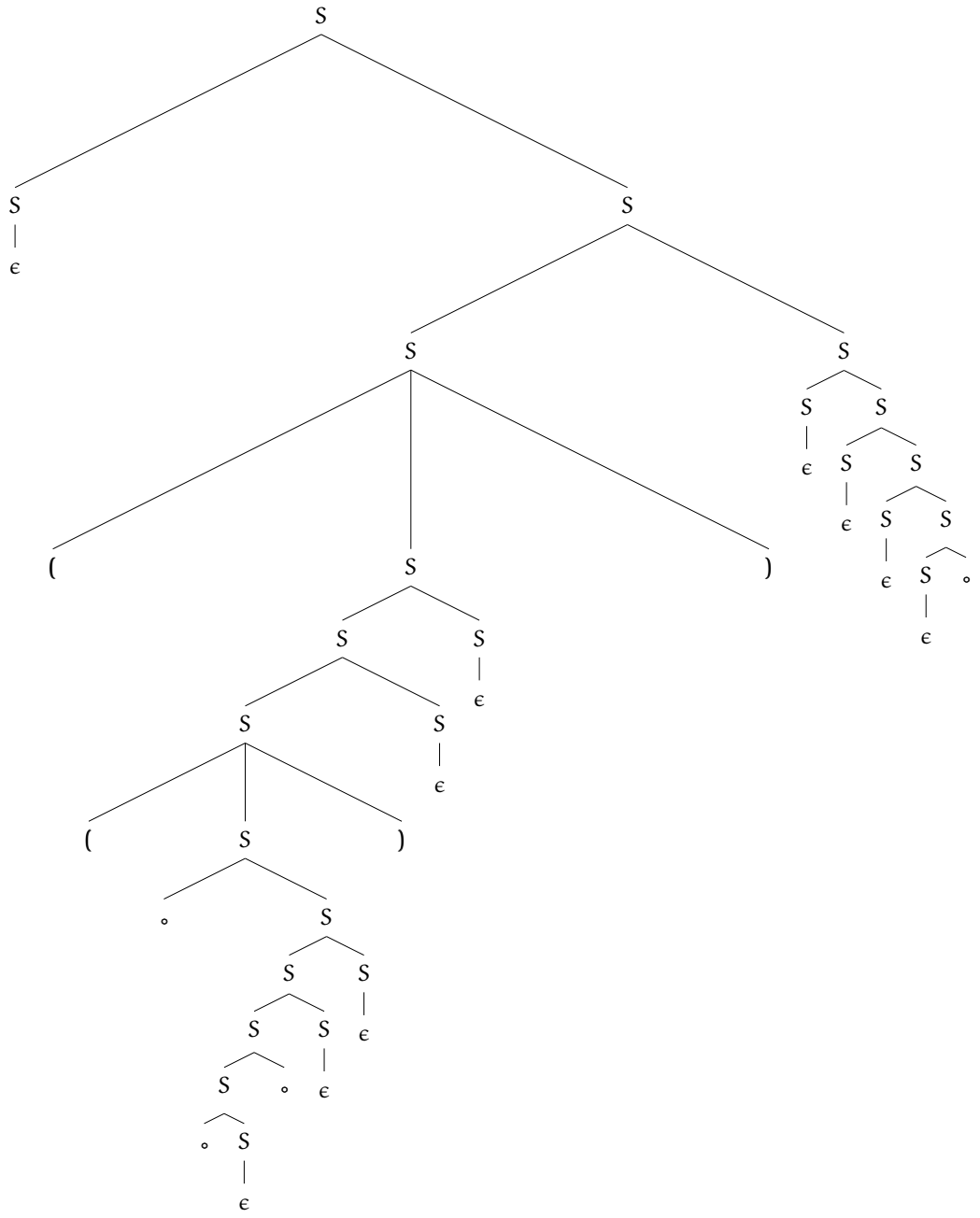


Figure 3.5.: **A more complex derivation tree.** Figure shows another derivation tree of the simple string $((\circ\circ\circ))\circ$ using grammar $\mathcal{G}_{e,1}$.

Notably, in CNF grammars $\mathcal{G} = (I, T, R, S)$, any production either is of the form $A \to BC$ or $A \to \alpha$, with A, B, $C \in I$ representing intermediate symbols and $\alpha \in T$ denoting a single terminal symbol. This essentially simple normal form has both theoretical and practical implications, as we will see in the sequel.

3.3.5. Stochastic Context-Free Grammars

As indicated above, SCFGs are a powerful tool for modeling combinatorial classes and the essence of probabilistic approaches towards RNA secondary structure prediction. We already know that secondary structures without pseudoknots can be encoded as words of a CFL and the class of all feasible structures can thus effectively be modeled via a corresponding CFG. However, the key idea is that extending this CFG to a corresponding SCFG, we can also model the fact that specific motifs of RNA secondary structures are more likely to be folded at certain stages than others (and not all possible motifs are equiprobable at any folding stage).

The proclaimed goal of the following section is to provide all needed formal definitions concerning SCFGs, and to discuss some basic concepts, methods and aspects that we will extensively rely on in the sequel.

3.3.5.1. Formal Definitions and Basic Concepts

Traditional CFGs are only capable of modeling the class of all generated string and thus inevitably induce a uniform distribution on the objects, while SCFGs additionally imply a (non-uniform) probability distribution on the considered class of objects. Basically, a particular SCFG is derived by equipping the productions of a corresponding CFG with probabilities such that the induced distribution on the generated language models as closely as possible the distribution of the sample data. The needed formalities are given as follows:

Definition 3.3.3 ([FH72]) *A* weighted context-free grammar (WCFG) *is a* 5*-tuple* $\mathcal{G} = (I, T, R, S, W)$*, where* (I, T, R, S) *is a CFG and* $W : R \to \mathbb{R}^+$ *is a mapping such that each rule* $f \in R$ *is equipped with a weight* $w_f := W(f)$*.*
If $\mathcal{G}$ *is a WCFG, then* $\mathcal{G}$ *is a* stochastic context-free grammar (SCFG) *iff the following additional restrictions hold:*

1. *For all* $f \in R$*, we have* $W(f) \in (0, 1]$*, which means the weights are probabilities.*
2. *The probabilities are chosen in such a way that for all* $A \in I$*, we have*

$$\sum_{f \in R, Q(f) = A} w_f = 1,$$

where $Q(f)$ *denotes the premise of the production* f*. In the sequel, we will write* $w_f : A \to \alpha$ *instead of* $f = (A, \alpha) \in R$*,* $w_f = W(f)$*.*

Consequently, for any SCFG $\mathcal{G} = (I, T, R, S, W)$, the mapping $W : R \to [0, 1]$ provides a probability distribution on the production rules. In many cases, the probability distribution on the production rules of a SCFG $\mathcal{G}$ also implies a probability distribution on the words of the language $\mathcal{L}(\mathcal{G})$, such that $\mathcal{G}$ might adequately be used as basis for probabilistic modeling of the generated class of objects; details will follow. Note that if $\mathcal{G}$ is a SCFG, we will mostly write $\mathcal{G} = (I, T, R, S, Pr)$ and hence consider the mapping $Pr : R \to [0, 1]$.

The concepts of derivation and ambiguity for SCFGs are the same as for usual CFGs. This means that each word $w \in \mathcal{L}(\mathcal{G})$ is generated in exactly the same way as for the corresponding CFG (I, T, R, S). However, considering $\mathcal{G} = (I, T, R, S, Pr)$, the mapping $Pr : R \to [0, 1]$ assigns a

probability $\Pr(d)$ to each derivation d of a word $w \in \mathcal{L}(\mathcal{G})$. The probability $\Pr(d)$ of a given derivation d is equal to the product of the probabilities of the production rules used in it, formally:

$$\Pr(d) = \prod_{\alpha \overset{f}{\Rightarrow} \beta \in d} \Pr(f).$$

Furthermore, we can use the mapping Pr to compute the probability $\Pr(w)$ for each word $w \in \mathcal{L}(\mathcal{G})$. To do this, we have to take into account that the SCFG $\mathcal{G}$ can be ambiguous, since then a word $w \in \mathcal{L}(\mathcal{G})$ may have more than one derivation. In fact, if a word $w \in \mathcal{L}(\mathcal{G})$ has k different leftmost derivations $d_1, \ldots, d_k$, then we have

$$\Pr(w) = \sum_{i=1}^{k} \Pr(d_i).$$

More generally, this means that we must sum up the probabilities of all possible leftmost derivations of a particular word $w \in \mathcal{L}(\mathcal{G})$, formally

$$\Pr(w) = \sum_{d = S \Rightarrow^* w} \Pr(d).$$

Thus, if the SCFG $\mathcal{G}$ is unambiguous, then the probability $\Pr(w)$ of a word $w \in \mathcal{L}(\mathcal{G})$ can be efficiently computed, as it is actually equal to the product of the probabilities $\Pr(f)$ of the production rules $f \in R$ that have to be used to generate this word, formally

$$\Pr(w) = \prod_{\alpha \overset{f}{\Rightarrow} \beta \in d = S \Rightarrow^* w} \Pr(f).$$

Example 3.3.5 *The corresponding SCFG of grammar $\mathcal{G}_{e,2}$ as introduced in Example 3.3.2 can be written as follows:*

$$\begin{aligned} &p_1 : S \to LS, \quad p_2 : S \to L, \\ &p_3 : L \to \circ, \quad p_4 : L \to (F), \\ &p_5 : F \to (F), \quad p_6 : F \to LS. \end{aligned}$$

Hence, considering the unique leftmost derivation d of the simple string $s = ((\circ\circ\circ))\circ$ as presented in Example 3.3.3, we find that

$$\begin{aligned} \Pr(s) &= \Pr(d) \\ &= \Pr(S \to LS) \cdot \Pr(L \to (F)) \cdot \Pr(F \to (F)) \cdot \Pr(F \to LS) \cdot \Pr(L \to \circ) \cdot \Pr(S \to LS) \cdot \Pr(L \to \circ) \cdot \\ &\quad \Pr(S \to L) \cdot \Pr(L \to \circ) \cdot \Pr(S \to L) \cdot \Pr(L \to \circ) \\ &= p_1 \cdot p_4 \cdot p_5 \cdot p_6 \cdot p_3 \cdot p_1 \cdot p_3 \cdot p_2 \cdot p_3 \cdot p_2 \cdot p_3. \end{aligned}$$

Example 3.3.6 *Consider the following SCFG based on grammar $\mathcal{G}_{e,1}$ of Example 3.3.1:*

$$1/15 : S \to \circ S, \quad 2/15 : S \to S\circ, \quad 3/15 : S \to (S), \quad 4/15 : S \to SS, \quad 5/15 : S \to \epsilon.$$

Then, the probabilities of the different leftmost derivations given in Example 3.3.4 are:

$$\begin{aligned} \Pr(d_1) &= (1/15)^4 \cdot (2/15)^0 \cdot (3/15)^2 \cdot (4/15)^1 \cdot (5/15)^2 \approx 2.34111 \times 10^{-8}, \\ \Pr(d_2) &= (1/15)^0 \cdot (2/15)^4 \cdot (3/15)^2 \cdot (4/15)^0 \cdot (5/15)^1 \approx 4.21399 \times 10^{-6}, \\ \Pr(d_3) &= (1/15)^4 \cdot (2/15)^0 \cdot (3/15)^2 \cdot (4/15)^7 \cdot (5/15)^8 \approx 1.15480 \times 10^{-14}. \end{aligned}$$

Note that the two more compact and thus more reasonable derivations d_1 and d_2 are significantly more probable than the comparably rather complex derivation d_3, since d_3 actually implies more rule

applications (and hence more probability terms) due to the high level of redundancy. Anyway, since there are many other possible leftmost derivations possible, the probability of the simple dot-bracket word $s = ((\circ\circ\circ))\circ$ *is given by*

$$\begin{aligned}\Pr(s) &= \Pr(d_1) + \Pr(d_2) + \Pr(d_3) + \sum_{d=S\Rightarrow^* s,\, d\notin\{d_1,d_2,d_3\}} \Pr(d) \\ &> \Pr(d_1) + \Pr(d_2) + \Pr(d_3) \approx 4.2374 \times 10^{-6}.\end{aligned}$$

Obviously, the computation of $\Pr(s)$ *is quite inefficient due to the ambiguity of the used grammar* $\mathcal{G}_{e,1}$.

Note that in derivations of words according to a WCFG $\mathcal{G} = (I, T, R, S, W)$, both $\overset{f}{\Rightarrow}$ and $\overset{W(f)}{\Rightarrow}$ may be used to explicitly mark the production $f \in R$ considered for a particular substitution.

Finally, it needs to be mentioned that if a SCFG $\mathcal{G} = (I, T, R, S, \Pr)$ indeed provides a probability distribution for the generated language $\mathcal{L}(\mathcal{G})$, that is if

$$\sum_{w\in\mathcal{L}(\mathcal{G})} \Pr(w) = 1$$

holds, then $\mathcal{G}$ is called *consistent*.

3.3.5.2. Parameter Estimation

In principle, SCFGs try to learn about the typical behavior of a particular class of objects from statistical grounds, by employing appropriate training procedures for estimating probabilities for the distinct production rules (that is, for calculating estimates for the respective grammar parameters). In fact, the probabilities of a SCFG $\mathcal{G}$ which generates the language $\mathcal{L}(\mathcal{G})$ can be trained from a database of words $w \in \mathcal{L}(\mathcal{G})$. As indicated in Section 3.3.2.1, SCFGs are trained according to the maximum likelihood principle on (hopefully) typical words of different sizes. Thereby, a SCFG $\mathcal{G}$ captures the probability distribution present in the sample set of words $w \in \mathcal{L}(\mathcal{G})$ provided for the training.

The conditions for consistency of such a trained grammar have been investigated by a number of scientists. As a result, several methods for the empirical estimation of SCFGs have been proposed in the literature which provide consistent SCFGs. For example, assigning relative frequencies found by counting the production rules used in the leftmost derivations of a finite sample of words $w \in \mathcal{L}(\mathcal{G})$ results in a consistent SCFG $\mathcal{G}$ [CG98]. In fact, it was shown that the maximum likelihood, the expectation maximization and a new cross-entropy minimization approach each provide a consistent SCFG, without restrictions on the grammar [NS03, NS06, CS06].

Note that training the probabilities of SCFG parameters by simply counting their relative frequencies actually yields a maximum likelihood estimate [CG98]. This is especially useful in connection with unambiguous SCFGs, since then the relative frequencies can be counted efficiently, as for every word, there is only one unique leftmost derivation to consider.

3.3.6. SCFGs for Structure Prediction

It is known for a long time that SCFGs can be used to model RNA secondary structure (see, for instance [SBH+94]). Moreover, SCFGs can be employed for deriving results on the expected structural behavior of RNAs, which might then be used for judging the quality of predictions made by any RNA folding algorithms [Neb02b, Neb04a]. Such results are quite realistic compared to other attempts to describe the structure of RNA quantitatively,

which for example assume an unrealistic combinatorial model [Wat78, Neb02a] or Bernoulli-model [HSS98, Neb04b] for RNA secondary structures. Furthermore, note that an SCFG mirror to the famous Turner energy model has been used in [NS11a] to perform the first analytical analysis of the free energy of RNA secondary structures.

However, SCFGs have also been directly and successfully used for the prediction of RNA secondary structure [KH99, KH03, DE04]. In this context, SCFGs traditionally model the combinatorial class (language) of all RNA sequences. Since the set of all possible base paired structures for a particular RNA sequence needs to be considered for calculating a corresponding prediction, any SCFG that will be useful in this context must be ambiguous in the general sense, that is there must be more than one possible derivation trees for some sequence, representing the feasible foldings.

Note that SCFGs are actually the standard way of implementing probabilistic (more specifically, *generative*) models for RNA. Basically, a generative method has a model parameterized with probability values and the probability distribution on the modeled RNA class is defined by the joint probabilities of sequences and their structures (details will follow).

Anyway, in order to identify the mathematically optimal (that is, highest probability) secondary structure for a given sequence, it is of high practical concern that each derivation tree for this sequence uniquely corresponds to one of its possible secondary structures. In fact, the popular CYK algorithm (details will follow in Section 3.3.7.1) can be used to find the optimal derivation tree, which is equal to the optimal folding if and only if there is a one-to-one correspondence between parse trees and secondary structures. If the same secondary structure is described by multiple valid derivation trees, the corresponding SCFG is called *structurally* ambiguous, which has been indicated to be of great disadvantage in connection with accurate DP methods (see, for instance [Gie00, DE04]).

3.3.6.1. Grammar Design

As already indicated in Section 3.3.4.2 (in connection with traditional CFGs), different SCFG designs can be used to model the same class of structures, where flexibility in model design comes from the fact that basically all distinct substructures can be distinguished. With increasing number of distinguished features, the resulting SCFG gains in both explicitness and complexity, which may result in a more realistic distribution on the modeled structure class. Principally, any grammar describing RNA secondary structures at least has to distinguish between paired and unpaired positions by using different productions to generate the corresponding symbols of the RNA sequence. For example, the SCFG $\mathcal{G}_{e,1}$ only captures the simplest folding features: unpaired bases, base pairings and bifurcations.

However, attempting to construct an elaborate SCFG that not only generates secondary structures but also models the distribution of the sample data as closely as possible, it is important to appropriately specify the set of production rules in order to guarantee that all substructures that have to be distinguished are derived from different rules. This is due to the fact that by using only one production rule f to generate different substructures (for instance, any unpaired nucleotides independent of the type of loop they belong to), there is only one weight (the probability $\Pr(f)$ of this production f) with which any of these substructures is generated, whereas the use of different rules $f_1, \ldots, f_k$ to distinguish between these substructures implies that they may be generated with different probabilities $\Pr(f_1), \ldots, \Pr(f_k)$, where $\Pr(f_1) + \ldots + \Pr(f_k) = \Pr(f)$. This way, we guarantee that more common substructures are generated with higher probabilities than less common ones.

Example 3.3.7 *A (rather simple) unambiguous SCFG $\mathcal{G}_{e,3}$ generating the language $\mathcal{L}$ characterized in Definition 3.3.2 is given by:*

$$\begin{aligned} &p_1 : S \to CA, \\ &p_2 : A \to (B)C, \quad p_3 : A \to (B)CA, \\ &p_4 : B \to \circ\circ\circ C, \quad p_5 : B \to CA, \\ &p_6 : C \to \epsilon, \qquad p_7 : C \to \circ C. \end{aligned}$$

This grammar unambiguously generates $\mathcal{L}$ for the following reasons:

- *Every sentential form $C(B)C(B)\cdots(B)C$ is obviously generated in a unique way; this resembles $\mathcal{L} := \mathcal{L}_u\mathcal{L}_{lu}^+$ and $\mathcal{L}_{lu} := (\mathcal{L}_l)\mathcal{L}_u$ of $\mathcal{L}$'s definition. The number of outermost pairs of brackets in the entire string uniquely determines the corresponding sentential form to be used.*
- *Now, B either generates a hairpin-loop from $\circ\circ\circ\mathcal{L}_u$, which is possible (in an unambiguous way) by rules $B \to \circ\circ\circ C$, $C \to \circ C$ and $C \to \epsilon$.*
- *Or else, B itself has to generate at least one additional pair of brackets. In this case, $B \to CA$ must be applied (only A can generate brackets) and then $A \to (B)C$ resp. $A \to (B)CA$ are used; the number of outermost brackets to be generated (from B under consideration) again uniquely determines that part of the derivation.*

Anyway, when changing the production $p_5 : B \to CA$ used to generate any possible k-loop for $k \geqslant 2$ (any loop that is not a hairpin loop) with probability p_5 into the two rules

$$p_{5.1} : B \to C(B)C, \quad p_{5.2} : B \to C(B)CA,$$

where $p_{5.1} + p_{5.2} = p_5$, it becomes possible to generate any possible 2-loop (that is, a stacked pair, a bulge (on the left or on the right), or an interior loop) and all kinds of multiloops (that is, any k-loop with $k \geqslant 3$) with different probabilities, which could increase the accuracy of the SCFG model. We denote the corresponding grammar by $\mathcal{G}'_{e,3}$.

By additionally replacing the first of these two new rules, $p_{5.1} : B \to C(B)C$, by the four productions

$$p_{5.1.1} : B \to (B), \quad p_{5.1.2} : B \to \circ C(B), \quad p_{5.1.3} : B \to (B)C\circ, \quad p_{5.1.4} : B \to \circ C(B)C\circ,$$

we obtain an even more specific grammar design, which we denote by $\mathcal{G}''_{e,3}$. Notably, we then have $(p_{5.1.1} + \ldots + p_{5.1.4}) + p_{5.2} = p_{5.1} + p_{5.2} = p_5$, meaning it becomes possible to distinguish between the different types of 2-loops more accurately, yielding a more realistic secondary structure model. In fact, in the case of significant differences of the new probabilities ($p_{5.1.1}, \ldots, p_{5.1.4}$ and $p_{5.2}$), we can expect a huge improvement in the model's accuracy.

Note that it is not hard to see that changes to a grammar like the ones just discussed do not change the language generated. However, this is not at all obvious with respect to ambiguity of the grammar. Hence, the corresponding changes need to be performed very carefully in order to ensure that the modified grammar remains unambiguous, which indeed has been done in the presented cases.

Notably, the (structural) unambiguity of rather complex SCFG designs can often readily be proven by describing the construction of their rule sets as done in Example 3.3.7. In brief, one starts with a rather simple and small (so-called *lightweight*) grammar that models only the basic structure motifs and specializes it (by replacing single productions that model one particular type of substructure by a bunch of corresponding new productions for generating the respective special types of substructures to be considered) until all substructures that need to be distinguished are represented by separate rules (and parameters). In order to avoid (structural) ambiguity, we only have to take care that at any point (where a more general old rule is replaced by a set of more specialized new ones), none of the considered alternative structure motifs can be constructed from more than one production. In this context, it is important to use different intermediate symbols for distinguished substructure types, thereby

ensuring that any intermediate symbol of the grammar uniquely corresponds to a particular class of substructures.

It is worth mentioning that different SCFG designs generally imply differences in the induced probability distributions, as illustrated by the following example.

Example 3.3.8 *The unique leftmost derivations of the secondary structure* $s = ((\circ\circ\circ))\circ$ *using the three grammars* $\mathcal{G}_{e,3}$, $\mathcal{G}'_{e,3}$ *and* $\mathcal{G}''_{e,3}$ *of Example 3.3.7, respectively, are given by*

$$\begin{aligned} d = S &\Rightarrow^* (B)C \overset{p_5}{\Rightarrow} (CA)C \overset{p_6}{\Rightarrow} (A)C \overset{p_2}{\Rightarrow} ((B)C)C \overset{p_4}{\Rightarrow} ((\circ\circ\circ C)C)C \overset{p_6}{\Rightarrow} ((\circ\circ\circ)C)C \overset{p_6}{\Rightarrow} ((\circ\circ\circ))C \\ &\Rightarrow^* ((\circ\circ\circ))\circ, \end{aligned}$$

$$d' = S \Rightarrow^* (B)C \overset{p_{5.1}}{\Rightarrow} (C(B)C)C \overset{p_6}{\Rightarrow} ((B)C)C \overset{p_4}{\Rightarrow} ((\circ\circ\circ C)C)C \overset{p_6}{\Rightarrow} ((\circ\circ\circ)C)C \overset{p_6}{\Rightarrow} ((\circ\circ\circ))C \Rightarrow^* ((\circ\circ\circ))\circ,$$

and

$$d'' = S \Rightarrow^* (B)C \overset{p_{5.1.1}}{\Rightarrow} ((B))C \overset{p_4}{\Rightarrow} ((\circ\circ\circ C))C \overset{p_6}{\Rightarrow} ((\circ\circ\circ))C \Rightarrow^* ((\circ\circ\circ))\circ$$

The corresponding parse trees are pictured in Figure 3.6. Hence, we have

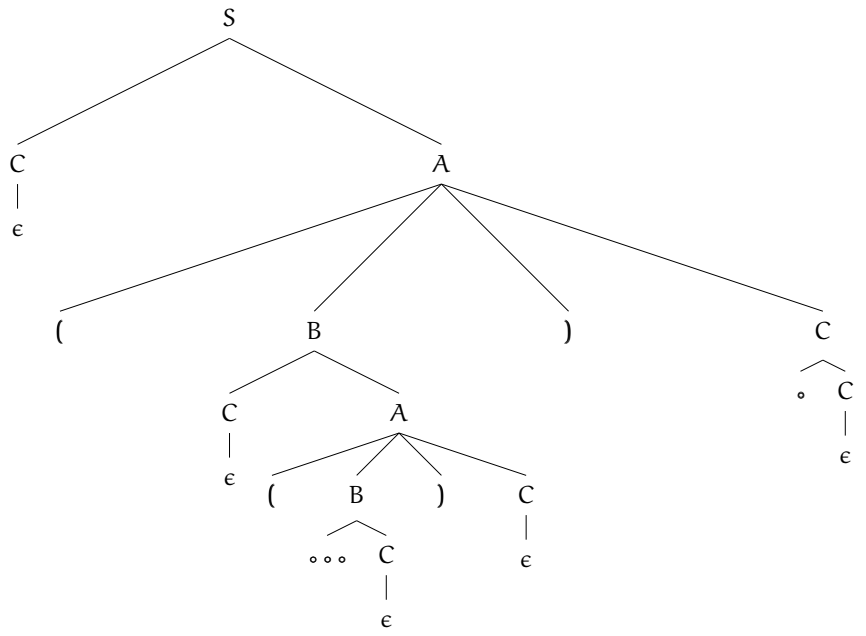
$$\begin{aligned} \Pr(d) &= \Pr(S \Rightarrow^\star (B)C) \cdot p_2^1 \cdot p_4^1 \cdot p_5^1 \cdot p_6^3 \cdot \Pr(((\circ\circ\circ))C \Rightarrow^\star s), \\ \Pr(d') &= \Pr(S \Rightarrow^\star (B)C) \cdot p_4^1 \cdot p_{5.1}^1 \cdot p_6^3 \cdot \Pr(((\circ\circ\circ))C \Rightarrow^\star s), \\ \Pr(d'') &= \Pr(S \Rightarrow^\star (B)C) \cdot p_4^1 \cdot p_{5.1.1}^1 \cdot p_6^1 \cdot \Pr(((\circ\circ\circ))C \Rightarrow^\star s). \end{aligned}$$

Nevertheless, standard loop-dependent thermodynamic models factor secondary structures in a more complex way, namely into terms for base pair stacking interactions (as opposed to individual base pairs) and diverse terms for different kinds of loops, which in many cases strongly depend on the lengths of the respective loops. Therefore, in order to closely mirror a particular state-of-the-art energy model, base pair stacking and explicit loop lengths might be considered the most important two features that should and actually can be captured by a corresponding SCFG.

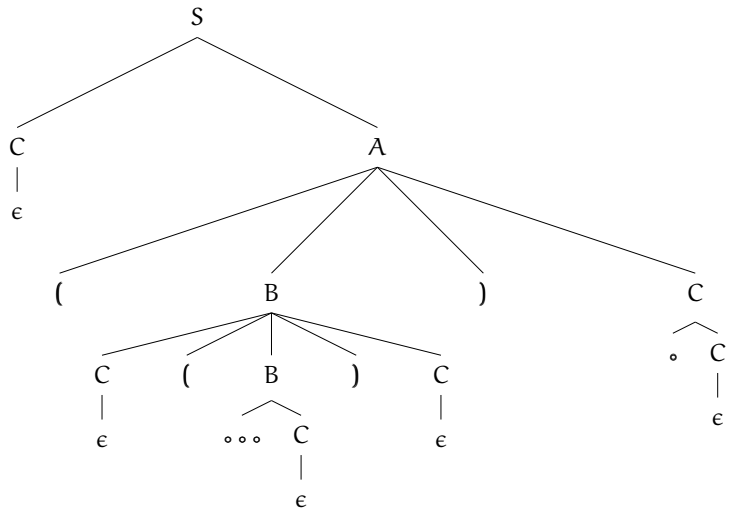
More information on how to deal with those and similar features can be found, for instance in [DE04]. In this thesis, we will only consider SCFGs that do not or only partially model base stacking, and we will also rarely use explicit loop lengths. This might actually be sufficient for obtaining a reliable probabilistic model, as indicated by previous works (see, for example [NS11a]).

Anyway, a straightforward approach for deriving a suitable grammar $\mathcal{G}_r$ for generating RNA sequences is given as follows: Initially, we construct a corresponding *unambiguous* grammar $\mathcal{G}_s = (I, T, R, S)$ that models the language of all possible secondary structures. Afterwards, we replace any production rule of the form $X \to \alpha(^z\beta)^z\gamma$ or $X \to \alpha\circ^z\beta$ (with α, β, $\gamma \in (I \cup T)^*$, $X \in I$, and $(^z\,)^z$ or $\circ^z$ representing $z \geqslant 1$ consecutive base pairs or unpaired bases) by corresponding new productions generating all considered individual base pairs and unpaired bases, respectively.

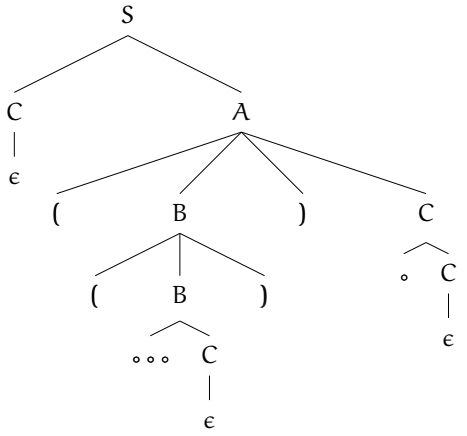
Example 3.3.9 *For applications to structure prediction, one could for example use the following ambiguous yet still structurally unambiguous SCFG for RNA sequences which has been constructed in the described way on the basis of the corresponding unambiguous SCFG* $\mathcal{G}_{e,2}$ *for RNA secondary structures from Example 3.3.5:*



(a) Parse tree according to $\mathcal{G}_{e,3}$.



(b) Parse tree according to $\mathcal{G}'_{e,3}$.



(c) Parse tree according to $\mathcal{G}''_{e,3}$.

Figure 3.6.: **Parse trees according to different grammar designs.** Figures display the unique parse trees for the dot-bracket word $((\circ\circ\circ))\circ$ using the three different (more and more specialized) variants of a same lightweight grammar as discussed in Example 3.3.7.

$$
\begin{array}{llll}
p_1 : S \to LS, & p_2 : S \to L, & & \\
p_{3.1} : L \to a, & p_{3.2} : L \to c, & p_{3.3} : L \to g, & p_{3.4} : L \to u, \\
p_{4.1} : L \to aFu, & p_{4.2} : L \to cFg, & p_{4.3} : L \to gFc, & p_{4.4} : L \to uFa, \\
p_{5.1} : F \to aFu, & p_{5.2} : F \to cFg, & p_{5.3} : F \to gFc, & p_{5.4} : F \to uFa, \\
p_6 : F \to LS. & & &
\end{array}
$$

Note that according to this grammar, only Watson-Crick pairs are allowed in all possible base paired secondary structures for a particular RNA sequence; other parings are prohibited since there are no corresponding production rules for generating them[10]. Obviously, the transformation of the secondary structure grammar $\mathcal{G}_{e,2}$ into the presented SCFG for RNA sequences implies a higher complexity by means of cardinality of the underlying rule set and hence results in a larger number probabilistic parameters that need to be estimated by corresponding training procedures.

3.3.6.2. Conditional Structure Probabilities

The essence of SCFG based approaches towards structure prediction is that the parameters of the underlying grammar actually provide a compact representation of a *joint* probability distribution over RNA sequences and their secondary structures, which is induced by the *joint* probabilities $\Pr(r, d)$ of generating a particular leftmost derivation d and some RNA sequence r using the considered SCFG.

Example 3.3.10 *Using the structurally unambiguous RNA grammar presented in Example 3.3.9, the joint probability of the secondary structure* $s = ((\circ\circ\circ))\circ$ *to be generated along with the sequence* $r = aucgaaug$ *is given by:*

$$
\begin{aligned}
\Pr(r, s) = \Pr(r, d = S &\overset{S\to LS}{\Rightarrow} LS \overset{L\to aFu}{\Rightarrow} aFuS \overset{F\to uFa}{\Rightarrow} auFauS \overset{F\to LS}{\Rightarrow} auLSauS \overset{L\to c}{\Rightarrow} aucSauS \\
&\overset{S\to LS}{\Rightarrow} aucLSauS \overset{L\to g}{\Rightarrow} aucgSauS \overset{S\to L}{\Rightarrow} aucgLauS \overset{L\to a}{\Rightarrow} aucgaauS \\
&\overset{S\to L}{\Rightarrow} aucgaauL \overset{L\to g}{\Rightarrow} aucgaaug) \\
= \Pr(S \to LS) &\cdot \Pr(L \to aFu) \cdot \Pr(F \to uFa) \cdot \Pr(F \to LS) \cdot \Pr(L \to c) \cdot \Pr(S \to LS) \cdot \\
&\Pr(L \to g) \cdot \Pr(S \to L) \cdot \Pr(L \to a) \cdot \Pr(S \to L) \cdot \Pr(L \to g).
\end{aligned}
$$

However, the goal of single sequence RNA secondary structure prediction is to find the best folding s for a given input sequence r. In connection with probabilistic parsing techniques, this requires a way to calculate the *conditional* probability $\Pr(s \mid r)$ of the secondary structure s given the RNA sequence r.

If generative probabilistic models (like SCFGs or HMMs) are used in this context, these conditional probabilities can readily be derived from the corresponding joint probabilities. Formally, for $\mathcal{S}$ denoting a set of valid derivation trees sharing the same secondary structure, it follows that

$$
\Pr(\mathcal{S} \mid r) = \sum_{d \in \mathcal{S}} \Pr(d \mid r) = \frac{\sum_{d \in \mathcal{S}} \Pr(r, d)}{\sum_{d' \in \mathcal{F}(r)} \Pr(r, d')}, \tag{3.4}
$$

where $\mathcal{F}(r) = \{d' = S \Rightarrow^\star r\}$ is the set of all possible derivation trees for sequence r (here S denotes the axiom of the used SCFG). Consequently, for structurally unambiguous SCFGs, the probability for generating a particular secondary structure s (with corresponding unique derivation tree d) given some RNA sequence r is equal to

$$
\Pr(s \mid r) = \Pr(d \mid r) = \frac{\Pr(r, d)}{\sum_{d' \in \mathcal{F}(r)} \Pr(r, d')},
$$

[10] This obviously corresponds to the case that such rules actually exist but are assigned weights or probabilities 0.

where $\mathcal{F}(r)$ then obviously defines the set of all feasible secondary structures for r (due to structural unambiguity), that is the so-called *folding space* for the considered sequence.

3.3.6.3. Parameter Estimation

When using SCFGs as a language for describing secondary structures on RNA sequences, the parameters have to be estimated from a given sample set of real-life sequences with annotated trusted secondary structures in order to derive corresponding sequence-dependent structural information. Like in the sequence-independent case (where only secondary structures without annotated sequences are considered), the training task involves finding exactly those probabilities for each of the production rules of particular SCFG $\mathcal{G}$ (that is, the set of parameters $\Phi = \{p_1, \ldots, p_n\}$ if $\mathcal{G}$ defines n rules) that maximize some specified objective function. However, the popular maximum likelihood technique as one of the most well-understood algorithms for parameter estimation can still be applied for this purpose.

Formally, let $\mathcal{D} = \{(x^{(1)}, y^{(1)}), \ldots, (x^{(m)}, y^{(m)})\}$ denote the considered set of training data, composed of m pairs of RNA sequences $x^{(i)}$ with trusted (that is, in general experimentally validated or sometimes alternatively computationally derived) secondary structures $y^{(i)}$, $1 \leqslant i \leqslant m$. Then, Φ is chosen to maximize the *joint likelihood* of the training sequences and their structures. Under the constraints traditionally imposed on the parameters of generative probabilistic models (namely that all parameters must be non-negative and certain groups of parameters must sum up to one), this likelihood[11] is given as follows:

$$\ell_{\mathrm{ML}}(\Phi \mid \mathcal{D}) = \prod_{i=1}^{m} \Pr(x^{(i)}, y^{(i)}; \Phi).$$

Note that the solution Φ_{ML} to this constrained optimization problem actually exists in closed form for structurally unambiguous SCFGs, being the reason why this technique is most commonly used in practice for estimating a particular set of grammar parameters. For more details, see for instance [DEKM98].

Anyway, if structural unambiguity is assured, then generative (SCFG based) training can easily and indeed efficiently be realized by counting the observed frequencies of applications of the distinct production rules needed for generating the structures in the considered sample set, as this not only yields a consistent SCFG, but also a maximum likelihood estimator of the SCFG parameters (see Section 3.3.5.2). More specifically, for training a structurally unambiguous RNA grammar, we make use of the fact that for any $i \in \{1, \ldots, m\}$, we know the trusted secondary structure $y^{(i)}$ for each sequence $x^{(i)}$ of the training set $\mathcal{D}$, such that the unique derivation tree that corresponds to $y^{(i)}$ can be used along with $x^{(i)}$, for $1 \leqslant i \leqslant m$, in order to determine the relative frequency of each production among all productions with the same premise. The relative frequencies that are obtained in this manner are indeed a maximum joint likelihood estimator for the probabilities that lead to the generation of the training data (see [Pre03] for details).

3.3.6.4. Choice of Training Data

If parameters for RNA secondary structure models are to be estimated, we basically have two different choices: First, we may consider a training set where only structures of a single biological class (for example tRNA) are contained. Then, we may expect that all structural properties (including aspects which are caused by interaction with proteins or by other *non-energetic* details of RNA folding) that are typical to this class are trained into the respective parameter values. For a general model of RNA folding, this has to be assumed some sort of

[11] Not probability, see Section 3.3.2.1.

"over-specialization", since we cannot expect the model to generalize well to new data from a different class. Second, we may use a rich training set of mixed biological classes. In that case, the danger of a potential lack of generalization is much smaller, but we lose the chance to capture some class-specific properties of the structures within our model.

In both cases, the main problem that comes inherently with the SCFG approach for modeling RNA structures and limits the performance of the corresponding computational prediction methods is that it is obviously highly dependent on the availability of a rich, reliable training set in order to minimize the danger of overfitting. Intuitively, this might especially be the case when using an excessively complex SCFG design that distinguishes between all different features in RNA structure aiming at a highly realistic model (for which a large number of parameters needs to be determined). The reason lies in the fact that attempting to obtain reliable estimation results for SCFGs with large numbers of parameters, we need comprehensive training sets for ensuring that enough observations are made for any structural motif modeled by one of the production rules, thereby avoiding that a particular structure (or shape) is trained into the model.

In this context, it should be mentioned that just recently, it has been shown that for complex models (SCFGs or otherwise), performance is highly sensitive to the structural diversity present in the training sample, not just to the total size and sequence diversity of the sample [RLE12]. This eventually means that for constructing robustly trained models, a larger number of structurally diverse RNA families must be considered, each containing a large number of diverse single sequences with well annotated structures. However, despite the zoo of publicly available RNA databases resulting from the fact that the number of solved secondary structures has dramatically increased over the past years, such an ideal set of structural RNA data for training statistical models could still not be designed. In fact, even the latest currently existing datasets like RNA STRAND [ABHC08] do not support the satisfactory training of complex probabilistic models.

3.3.6.5. Separation of Parameters

In algorithms and applications based on SCFGs, the grammar parameters are often split into a set of *transition probabilities* and corresponding sets of *emission probabilities*. This separation into transition and emission probabilities actually corresponds to the standard treatment of (generative) model parameters as applied for example in the case of HMMs. For a particular RNA grammar $\mathcal{G}_r$ with underlying SCFG $\mathcal{G}_s = (I, \Sigma_r, R, S, Pr)$ for modeling secondary structures, the probabilities of the rules of $\mathcal{G}_r$ are thus split into transition probabilities $Pr_{tr}(\textit{rule})$ for $\textit{rule} \in R$ and corresponding emission probabilities $Pr_{em}(r_x)$ for $r_x \in \Sigma_r$ and $Pr_{em}(r_{x_1} r_{x_2})$ for $r_{x_1} r_{x_2} \in \Sigma_r^2$, that is for the individual unpaired bases and possible base pairings, respectively.

Note that in cases of SCFGs for modeling RNA secondary structure, it has become custom that all emission probabilities (for the 4 individual unpaired bases and the resulting 16 distinct possible base pairings, respectively) come from the same distribution. That is, for any considered loop type, one uses the same emission probabilities for unpaired bases located within and base pairs closing a corresponding loop. Hence, this separation into rule and emission probabilities allows us to only consider the productions of the underlying unambiguous SCFG modeling the class of all feasible secondary structures, although we actually had to deal with the larger set of productions of the corresponding ambiguous SCFG generating any possible RNA sequence (where the derivation trees uniquely correspond to the different secondary structures for that sequence).

Example 3.3.11 *By linking together the emissions of base pairs generated with different rules, the joint probability of generating the sequence* $r = \mathtt{aucgaaug}$ *and the secondary structure* $s = ((\circ\circ\circ))\circ$ *(corresponding to the unique leftmost derivation* d *given in Example 3.3.10) is computed as follows:*

$$\begin{aligned}\Pr(r,s) = {} & \Pr_{tr}(S \to LS) \cdot \Pr_{tr}(L \to (F))\Pr_{em}(au) \cdot \Pr_{tr}(F \to (F))\Pr_{em}(ua) \cdot \Pr_{tr}(F \to LS) \cdot \\ & \Pr_{tr}(L \to \circ)\Pr(c) \cdot \Pr_{tr}(S \to LS) \cdot \Pr_{tr}(L \to \circ)\Pr_{em}(g) \cdot \Pr_{tr}(S \to L) \cdot \\ & \Pr_{tr}(L \to \circ)\Pr_{em}(a) \cdot \Pr_{tr}(S \to L) \cdot \Pr_{tr}(L \to \circ)\Pr_{em}(g).\end{aligned}$$

It is easy to see that for a particular SCFG, this separation might actually reduce the number of free parameters that need to be estimated by the employed training procedures in a very significant way, such that the ever-present danger of overfitting becomes less threatening.

With respect to training, separating RNA grammar parameters into transition and emission terms is quite unproblematic in practice: Let $\mathcal{D} = \{(x^{(1)}, y^{(1)}), \ldots, (x^{(m)}, y^{(m)})\}$ be the considered set of RNA data. Then, instead of using the derivation tree that corresponds to the correct secondary structure $y^{(i)}$ for sequence $x^{(i)}$, $1 \leqslant i \leqslant m$, to determine the relative frequency of each production among all productions with the same premise, we simply have to count the relative frequencies of applications of the production rules of the underlying secondary structure grammar, along with the corresponding relative frequencies of emissions of unpaired bases and base pairs that are observed in the training set. Using a particular parsing technique (like for instance *probabilistic Earley parsing*, see Section 3.3.7.1) in order to count these frequencies, this might often even be more efficient in practice, due to the smaller number of production rules of the underlying secondary structure grammar. However, the relative frequencies that are obtained in this manner are still a maximum likelihood estimator for the probabilities of the more complex (by means of number of production rules) RNA grammar.

3.3.7. SCFG Based Algorithms

When considering structure prediction as a mathematically well-defined optimization problem (as described in Section 3.1), then given a stochastic RNA grammar $\mathcal{G}$, one traditionally looks for a valid derivation tree with maximal probability among all possible derivation trees for the given input sequence[12]. This derivation tree corresponds to the most likely secondary structure using the SCFG $\mathcal{G}$ and is usually called *Viterbi parse*. To compute Viterbi parses, one can utilize adapted versions of well-established parsing algorithms, as we will see in Section 3.3.7.1. A related alternative parsing variant (following a different optimization goal) that has also been successfully applied to the RNA folding problem will be discussed in Section 3.3.7.3. This method actually calculates a so-called *MEA parse* rather than a corresponding Viterbi parse.

3.3.7.1. Adapted Parsing Techniques

CFGs are simple enough to allow the construction of efficient (polynomial-time) *recognition* and *parsing* algorithms which, for a given input string, determine whether and how (in terms of corresponding parse tree) it can be generated from the considered CFG. These algorithms are usually described by recursive DP routines. Two popular examples which we sometimes directly build on in the sequel will be briefly discussed in this section. For a more detailed introduction to parsing techniques for CFGs and a quite exhaustive collection of existing parsing strategies in general, we refer to [GJ08].

[12]Note that when using a non-stochastic RNA grammar, all derivation trees are actually equiprobable, due to the implicit assumption of a uniform distribution.

Cocke-Younger-Kasami Algorithm The most well-known and fairly simple variant due to Cocke [CS70], Younger [You67] and Kasami [Kas65], usually called the *Cocke-Younger-Kasami (CYK)* algorithm, can only handle some restrictive subsets of CFGs. In fact, the CYK algorithm in its original form is only described for non-stochastic CFGs in CNF. The CYK parsing algorithm can readily be modified to compute the most probable parse tree of a given input sequence according to the probability distribution on all possible parse trees for that sequence, as induced by a given SCFG. This is typically realized by incorporating log probabilities[13] of the production rules of any considered SCFG into the DP recursions.

Since the transformation of an arbitrary CFG $\mathcal{G}$ into CNF may lead to an undesirable bloat in the number of productions, it has proven convenient to avoid this conversion in applications and express the CYK recursions directly in terms of the production rules of $\mathcal{G}$. For instance, when considering the CFG $\mathcal{G}_{e,1}$ of Example 3.3.1, the probabilistic variant of the CYK algorithm relies on the following recursions for a given input sequence r:

$$M_{i,i-1} = \log(\Pr(S \to \epsilon)), \text{ for } 1 \leqslant i \leqslant n, \text{ and}$$

$$M_{i,j} = \max \begin{cases} \log(\Pr(S \to r_i S)) + M_{i+1,j}, \\ M_{i,j-1} + \log(\Pr(S \to S r_j)), \\ \log(\Pr(S \to r_i S r_j)) + M_{i+1,j-1}, \\ \max\limits_{i<k<j-1} M_{i,k} + M_{k+1,j} + \log(\Pr(S \to SS)), \end{cases} \quad \text{for } 1 \leqslant i \leqslant j-1 \text{ and } 1 \leqslant j \leqslant n.$$

Obviously, there is a near-exact correspondence between probabilistic CYK parsing and standard DP algorithms for MFE calculations. In fact, SCFG based DP methods for computing the optimal (maximum probability) parse tree essentially use the same recursion scheme as physics-based DP algorithms for deriving the optimal (MFE) conformation (see Section 3.2.2.1). However, while thermodynamic methods are based on factoring the structure into a sum of energy terms or arbitrary base pair scores (according to a particular energy model), the scoring system of the CYK variant is in fact probabilistic, based on factoring the score for a structure down into a sum of log probability terms (according to the underlying SCFG model). Hence, the main difference is that in case of a sophisticated CYK recursion scheme which distinguishes the same canonical substructures and structural motifs as a particular MFE algorithm, the corresponding scores are all derived from observations in known RNA structures rather than from experimental studies and extrapolations.

Furthermore, there is a notable difference between thermodynamic and probabilistic approaches to optimal structure prediction with respect to structural ambiguity:

- Physics-based approaches traditionally use free energy values as scores, such that the thermodynamic scoring scheme is not normalized. Hence, even if the free energy of a particular structure is scored more than once, the scores (free energies) of all considered structures effectively remain the same. This actually means that regardless of how many different ways there are of scoring the energy of a structure, the MFE structures still win. Consequently, structural ambiguity is not an issue in connection with methods for computing optimal (by means of MFE) foldings.
- Probabilistic approaches, however, use normalized probabilities for distinguished structural motifs as scores, resulting in a normalized scoring scheme. Thus, if a structure is considered more than once due to structural ambiguity of the underlying grammar, then the scores (structure probabilities) are inherently biased, since the probability of

[13] Using the logarithm of all probabilities has two main advantages: speed and accuracy. In fact, since the log of a product is equal to the sum of the logs of all factors, all products are turned into sums and addition is less expensive than multiplication. Furthermore, the use of log probabilities improves numerical stability, as the underflow problem can be essentially solved. Note that the base of the logarithm is not important as long as it is larger than 1 (for instance 2, Euler's number e, or 10).

any secondary structure is then generally dependent on (somehow "weighted" by) the number of its corresponding valid derivations. As a consequence, if the considered SCFG is structurally ambiguous, then the most probable parse tree represents only its own probability, but not the probabilities of the other alternatives. Hence, in connection with methods for computing the optimal (most probable) folding for a given sequence, structural ambiguity is an important practical concern (as shown for instance in [DE04]).

Earley Parsing Another widely known parsing algorithm that can actually handle *arbitrary* CFGs is known as *Earley's algorithm*, or *Earley parser* [Ear70]. Besides the obvious benefit that no normal form is required for applying this algorithm, it is in fact one of the most efficient known parsing techniques. Furthermore, it efficiently handles *left-recursion* [GJ08]. Moreover, *probabilistic* Earley parsing can actually be used for determining the most probable derivation of a given input string with respect to a considered SCFG, hence providing an attractive alternative to the CYK algorithm with respect to computing Viterbi parses.

Principally, a (non-probabilistic) Earley parser operates on lists of *items*, which define position indices along with so-called *dotted rules* and are used to represent partial derivations. Specifically, a dotted rule is a production of the grammar with a dot symbol $\bullet$ inserted somewhere in the right-hand side of this production, where dots at the very left and right positions are explicitly allowed. For example, the production $A \to \alpha\beta$ of a grammar $\mathcal{G} = (I, T, R, S)$, with $A \in I$ and $\alpha, \beta \in I \cup T$, implies the three dotted rules $A \to \bullet\alpha\beta$, $A \to \alpha \bullet \beta$ and $A \to \alpha\beta\bullet$ to be considered by Earley's algorithm.

For any dotted rule $\mathit{rule} = A \to \alpha \bullet \beta$, $A \in I$ and $\alpha, \beta \in (I \cup T)^*$, we can write items in the form $[i, \mathit{rule}, j]$, for $1 \leqslant i, j \leqslant n+1$, where n is the length of the input string w to be parsed. An item of this form is derived iff there exists a sentential form $\gamma \in (I \cup T)^*$ such that

$$S' \Rightarrow^\star w_{1,i-1} A \gamma \overset{A \to \alpha\beta}{\Rightarrow} w_{1,i-1} \alpha\beta\gamma \Rightarrow^\star w_{1,i-1} w_{i,j-1} \beta\gamma.$$

Briefly, the Earley parser is initialized with the start item $[1, S' \to \bullet S, 1]$, for $S' \notin I \cup T$ and S is the axiom of the used CFG $\mathcal{G}$[14]. Then, the transitive closure with respect to the following operations is computed:

- Predictor: If $\exists\, ([i, C \to \gamma \bullet A\delta, j]$ and $A \to \alpha \in R)$, then add $[j, A \to \bullet\alpha, j]$,
- Scanner: If $\exists\, [i, A \to \alpha \bullet w_{j-1}\beta, j-1]$, then add $[i, A \to \alpha w_{j-1} \bullet \beta, j]$,
- Completer: If $\exists\, ([i, A \to \alpha \bullet B\beta, k]$ and $[k, B \to \gamma\bullet, j])$, then add $[i, A \to \alpha B \bullet \beta, j]$.

Intuitively, the predictor adds all the productions that might yield a valid extension of the (nonterminal) symbol following the dot, whereas the scanner advances the dot past terminal symbols if they match the corresponding symbol of the parsed string and the completer advances the dot symbol past (nonterminal) symbols if they could actually be validly extended. The goal item $[1, S' \to S\bullet, n+1]$ is thus derived if and only if the considered grammar $\mathcal{G}$ generates the input string w, this is if and only if $w \in \mathcal{L}(\mathcal{G})$.

Note that a corresponding probabilistic variant of Earley's algorithm simply keeps track of the probabilities of partial derivations by adding them to the corresponding items as an additional parameter. For further information on probabilistic Earley parsing, we refer to [Sto95].

[14] Thus, S' is a new nonterminal symbol and $S' \to S$ a new production.

3.3.7.2. Inside-Outside Algorithm

An additional popular and highly relevant algorithm for SCFGs in CNF which besides others has been successfully applied to the RNA folding problem is the so-called *inside-outside algorithm* [LY90, LY91]. This algorithm actually represents the natural counterpart of the *forward-backward algorithm* for HMMs (for details, we refer to [Bis07, DEKM98]), where its computational complexity is substantially greater.

Briefly, the inside-outside algorithm has two basic steps, each realized by a corresponding DP method. In particular, for X an intermediate symbol of the considered grammar $\mathcal{G}$, the first step called *inside algorithm* (which corresponds to the forward step in case of HMMs) computes all *inside probabilities* $\alpha_X(i,j)$, $1 \leqslant i,j \leqslant n$, for a given word $w \in \mathcal{L}(\mathcal{G})$ of size n. Afterwards, the *outside algorithm* (corresponding to the backward step for HMMs) calculates all *outside probabilities* $\beta_X(i,j)$, $1 \leqslant i,j \leqslant n$, for the same word w, making use of the inside values derived in step one.

Definition of Inside and Outside Values Specifically, for an underlying CFG $\mathcal{G} = (I, T, R, S)$ and an input $w \in \mathcal{L}(\mathcal{G})^n$, the inside and outside values for any two indices $i,j \in \{1,\ldots,n\}$ are defined as follows:

- $\alpha_X(i,j)$ is the probability of a leftmost derivation that generates the subword $w_i \ldots w_j$ from the intermediate symbol $X \in I$, formally

$$\alpha_X(i,j) := \Pr(X \Rightarrow^* w_i \ldots w_j). \tag{3.5}$$

  Hence, this value is the probability of covering a range of the input w starting from a specified nonterminal symbol X. It might thus be computed simply as the sum of the probabilities of all derivations of the subword $w_i \ldots w_j$ starting with X.

- $\beta_X(i,j)$ is the probability of a derivation which, starting with the intermediate symbol S (the axiom of grammar $\mathcal{G}$), generates the sentential form $w_1 \ldots w_{i-1}\, X\, w_{j+1} \ldots w_n$, formally

$$\beta_X(i,j) := \Pr(S \Rightarrow^* w_1 \ldots w_{i-1}\, X\, w_{j+1} \ldots w_n). \tag{3.6}$$

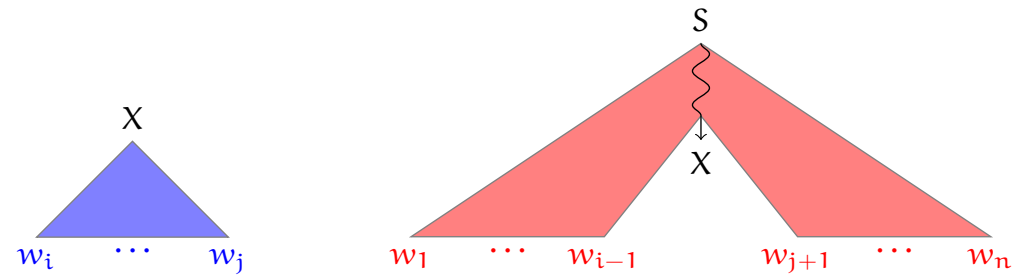
  Intuitively, this value is the probability of everything surrounding a certain nonterminal symbol, and thus in some way dual to the corresponding inside probability. Note that for the computation of this outside probability, one always summarizes over all corresponding derivation trees.

The product of a particular inside probability and its corresponding outside probability actually covers the whole input by using a specified nonterminal symbol for a certain range, since

$$\begin{aligned}\alpha_X(i,j) \cdot \beta_X(i,j) &= \Pr(X \Rightarrow^* w_i \ldots w_j) \cdot \Pr(S \Rightarrow^* w_1 \ldots w_{i-1}\, X\, w_{j+1} \ldots w_n) \\ &= \Pr(S \Rightarrow^* w_1 \ldots w_{i-1}\, X\, w_{j+1} \ldots w_n \Rightarrow^* w_1 \ldots w_n).\end{aligned}$$

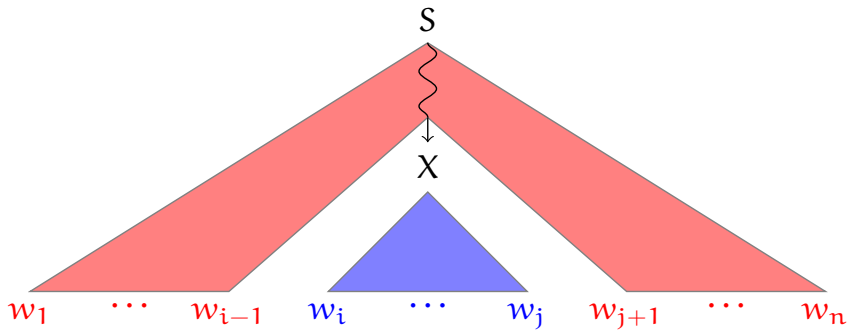
Hence, the outside values describe a convenient relationship to the inside values. In fact, if $X \in I$ describes a particular feature of the class of strings modeled by the considered grammar $\mathcal{G}$, then any product $\alpha_X(i,j) \cdot \beta_X(i,j)$, $1 \leqslant i,j \leqslant n$, equals the sum of the probabilities of all leftmost derivations using the given intermediate symbol X at a particular point. An illustration of individual inside and outside probabilities and their relationship is presented in Figure 3.7.

Computation of Inside Values Conceptually, the inside algorithm is strongly related to the CYK parsing algorithm. In fact, for any input sequence r of length n, the goal value of the inside algorithm is given by $\alpha_S(1,n)$ for S the axiom of the used SCFG. This value is equivalent to the total probability of the input sequence given the underlying stochastic model and might



(a) Inside value $\alpha_X(i,j)$.

(b) Outside value $\beta_X(i,j)$.



(c) Product $\alpha_X(i,j) \cdot \beta_X(i,j)$.

Figure 3.7.: **Inside and outside values.** Figures illustrate the inside and outside probabilities $\alpha_X(i,j)$ and $\beta_X(i,j)$, $1 \leqslant i,j \leqslant n$, for a particular intermediate symbol X of a SCFG $\mathcal{G}$ with start symbol S, as parts of a corresponding parse tree for input word $w = w_1 \ldots w_n$ using $\mathcal{G}$.

thus be obtained by summing over all parse trees for the input sequence. To do this, the inside algorithm replaces the max operations in the corresponding CYK recursions with sums, and additions of terms with multiplications. It does not consider log probabilities.

Just like for the CYK algorithm itself, the corresponding recursions for its inside variant may conventionally be expressed directly in terms of the productions of a particular SCFG $\mathcal{G}$, albeit in its original form [LY90] it can only handle grammars in CNF. For instance, given an arbitrary input sequence r of length n, then the CYK inside algorithm for the CFG $\mathcal{G}_{e,1}$ of Example 3.3.1 can be described by the following recursion scheme:

$$\alpha_S(i,i-1) = \Pr(S \to \epsilon), \text{ for } 1 \leqslant i \leqslant n, \text{ and}$$

$$\alpha_S(i,j) = \sum \begin{cases} \Pr(S \to r_i S) \cdot \alpha_S(i+1,j), \\ \alpha_S(i,j-1) \cdot \Pr(S \to S r_j), \\ \Pr(S \to r_i S r_j) \cdot \alpha_S(i+1,j-1), \\ \sum\limits_{i<k<j-1} \alpha_S(i,k) \cdot \alpha_S(k+1,j) \cdot \Pr(S \to SS), \end{cases} \quad \text{for } 1 \leqslant i \leqslant j-1 \text{ and } 1 \leqslant j \leqslant n.$$

Note that the inside algorithm is somehow analogous to the McCaskill algorithm [McC90] for calculating PFs on the basis of (experimentally obtained) thermodynamic parameters for RNA secondary structure. Anyway, there is also a noticeable difference between both algorithms: When calculating the equilibrium PF for a given sequence, one must be careful not to count any structure more than once in order to obtain an unbiased Boltzmann distribution of possible structures. In contrast, structural ambiguity is not an issue in connection with summed inside calculation (the analogue of the summed PF computation), since it actually sums over all possible parse trees rather than over all possible secondary structures, such that the inside algorithm gives the correct result even for ambiguous SCFGs.

Computation of Outside Values While inside probabilities of components of a given input string and for a particular SCFG can easily be computed with extended variants of CYK or Earley parsers, the construction of corresponding parsing algorithms for calculating the outside probabilities is in general significantly more complex. Actually, a generally valid method (of similar simplicity) corresponding to [LY90] that can be used in connection with arbitrary grammar specifications has not yet been devised and is thus an unsolved task.

However, an attractive alternative is to use the concept of *semiring parsing* [Goo98, Goo99] in order to derive a grammar specific parser that computes the outside values of all string fragments using an arbitrary SCFG. Briefly, semiring parsing represents a formalism for describing parsers such that a single simple description can be used to generate parsers that compute all the interesting quantities for a given grammar and a corresponding string, including recognition, derivation forests, and in case of SCFGs among others also Viterbi paths or inside probabilities. In fact, the idea of semiring parsing provides a beautifully unified view on the problem of generating parsing algorithms for calculating any of these *forward values* and also the corresponding *reverse values* which are typically much more difficult to compute[15]. As pointed out in [Goo98], this makes it especially useful for finding parsers for outside probabilities, and for parsers that can handle arbitrary grammar designs, like Earley-style parsers.

3.3.7.3. MEA Parsing

In connection with RNA structure analysis, both the inside values and the outside values are of great relevance for several practical applications, including parameter re-estimation by expectation maximization[16] and a special variant of SCFG based structure prediction, called *maximum expected accuracy (MEA)* parsing, as opposed to Viterbi parsing.

Briefly, just like Viterbi parsing, MEA parsing is also an optimization problem, but a different objective function is considered. In fact, the optimal structure is defined as the one with the highest expected number of correctly predicted positions. Thus, an *MEA parse* for a given sequence is actually a valid derivation tree which maximizes the number of correctly unpaired and paired positions with respect to the true folding of that sequence.

Among all possible derivation trees for the given input sequence, this parse can be identified by a corresponding DP method similar to the CYK algorithm. Basically, the sole difference is that the recursion scheme makes use of sequence-specific probabilities for unpaired bases and base pairs, which are traditionally calculated from the corresponding inside and outside values of components of the input sequence. Originally (see [KH03], supplemental material), for an input sequence of length n, the following recurrence system is used for calculating the maximum expected accuracy $M_{1,n}$:

$$M_{i,i} = q_i, \text{ for } 1 \leqslant i \leqslant n, \text{ and}$$

$$M_{i,j} = \max \begin{cases} q_i + M_{i+1,j}, \\ M_{i,j-1} + q_j, \\ p_{i,j} + M_{i+1,j-1}, \\ \max\limits_{i<k<j-1} M_{i,k} + M_{k+1,j}, \end{cases} \qquad \text{for } 1 \leqslant i \leqslant j-1 \text{ and } 1 \leqslant j \leqslant n,$$

[15] Goodman [Goo98] introduces forward value and reverse value as generalizations of inside value and outside value, respectively, for arbitrary semirings.

[16] The inside and outside values can be used to re-estimate the probabilities (parameters) of a given SCFG by expectation maximization in a similar way as the forward and backward variables can be used for HMM training by expectation maximization, see for instance [DEKM98].

where q_i denotes the probability that i remains unpaired and $p_{i,j}$ denotes the probability that i pairs with j. Under the assumption that the underlying CFG $\mathcal{G} = (I, T, R, S)$ for modeling the considered class of all feasible secondary structures exclusively uses productions of the form $A \to (B)$ for generating base pairs, where $A, B \in I$, the probability (under the model) that positions i and j, $1 \leqslant i, j \leqslant n$, form a base pair is effectively computed by summing over all rules forming pairs:

$$\begin{aligned} p_{i,j} &= \sum_{A,B} \beta_A(i,j) \cdot \sum_{xy} \Pr(A \to xBy) \cdot \alpha_B(i+1, j-1) \\ &= \sum_{A,B} \beta_A(i,j) \cdot \Pr(A \to (B)) \cdot \alpha_B(i+1, j-1). \end{aligned} \tag{3.7}$$

It immediately follows that the probability for position i, $1 \leqslant i \leqslant n$, remaining unpaired is given by:

$$q_i = 1 - \sum_{j \neq i} p_{i,j}.$$

It should be mentioned that an efficient implementation of this MEA parsing algorithm is given by the popular `Pfold` tool [KH99, KH03]. Notably, `Pfold` makes use of inside-outside probabilities derived on the basis of a particular stochastic variant of the (lightweight) CFG $\mathcal{G}_{e,2}$ of Example 3.3.2. However, `Pfold` has actually not been created for single sequence structure prediction in the first place. In fact, the underlying algorithm has originally been designed to take an alignment of related RNA sequences as input and then predict a common (consensus) structure for all sequences. Recently, a new parallelized version of the algorithm has been created, named `PPfold` [SKV+11], which manages to solve the structure of much longer alignments than `Pfold` without breaking down.

Anyway, note the MEA calculations principally only require the posterior probabilities of any two positions being base paired, which can obviously be derived by employing the inside-outside algorithm [LY90, LY91] on the basis of a particular probabilistic model. However, these pairing probabilities might equivalently be calculated with the McCaskill algorithm [McC90] when considering a thermodynamic model. Furthermore, different MEA methods have been implemented for RNA folding. Originally, the posterior probabilities of base pairs were maximized [KH99, DWB06], as described above, but it is also possible to calculate centroid estimators [CLD05, HKS+09]. Since any of these methods for finding the MEA structure have consistently improved performance over corresponding traditional algorithms that compute the most probable structure (both for thermodynamic and probabilistic models), most RNA folding packages currently prefer a MEA estimator in order to compute their predictions [LGM09].

3.3.8. Length-Dependent Stochastic Context-Free Grammars

An essential basic fact in connection with SCFG approaches is that at any point, the probability for generating a particular structure motif (as modeled by the grammar) is given by the corresponding estimated parameter value (of the corresponding production rule), which does actually not depend on the *length* of the generated substructure, although in reality it often does. For example, using a SCFG the probability for leaving a particular fragment unpaired instead of folding at least one additional base pair on it (resulting for instance in a simple hairpin loop instead of a more stable paired substructure) is identical for any fragment length, but in nature short fragments are much more likely to be left unpaired than longer ones (as it is usually energetically more favorable to fold additional base pairs if possible).

In order to model this native behavior of RNA molecules, it seems reasonable to additionally include *length-dependencies* into traditional SCFG models. To the best of our knowledge, this idea has first been applied in [NE07] in connection with database similarity searching based

on *covariance models (CMs)*. Briefly, CMs are profile SCFGs (a particular SCFG architecture) for cleanly describing both the secondary structure and the primary sequence consensus of an RNA (see, for example [ED94]). They are widely used in general approaches to several RNA analysis problems, such as consensus structure prediction, multiple sequence alignment and, in fact, database similarity searching. Principally, in [NE07], it is described how to accelerate CM searches by using a *banded* DP strategy (a standard approach in many areas of sequence analysis), which actually for each node calculates the probability of generating a subsequence of a particular length.

Nevertheless, with respect to traditional probabilistic RNA secondary structure prediction methods, one might easily consider an appropriate *length-dependent* SCFG (LSCFG), as formally introduced in [WN11]. Basically, LSCFGs exactly address the problem sketched above, that is they are defined as an extension to the concept of conventional SCFGs such that the probabilities of the productions depend on the length of the generated subword. One important benefit of this new formalism is that in general, LSCFG based algorithms can be implemented to have the same worst-case time and space requirements as their length-independent counterparts. However, due to the larger number of grammar parameters implied, algorithms implementing LSCFG models are obviously not only more explicit (due to the higher level of specialization), but unfortunately also more prone to overfitting than the corresponding traditional SCFG variants.

3.3.8.1. Formal Definitions

When attempting to improve the ability of a particular stochastic model to capture typical features of a particular language within its parameters, length-dependencies can be incorporated into traditional SCFGs according to the following definition:

Definition 3.3.4 ([WN11]) *A* length-dependent stochastic context-free grammar (LSCFG) *is defined as a SCFG* $\mathcal{G} = (I, T, R, S, \Pr)$ *with the following exceptions:*

- $\Pr : R \times \mathbb{N} \to [0, 1]$ *now takes a second argument (length of subword generated).*
- *The constraint on the probabilities changes to:*

$$\forall A \in I \; \forall n \in \mathbb{N} : \sum_{A \to \alpha \in R} \Pr(A \to \alpha, n) \in \{0, 1\}.$$

- *Additionally, we introduce a probability distribution* $\Pr(n)$ *on the lengths of the words in* $\mathcal{L}(\mathcal{G})$*, that is*

$$\sum_{n \in \mathbb{N} \,:\, T^n \cap \mathcal{L}(\mathcal{G}) \neq \emptyset} \Pr(n) = 1.$$

- *Let* $\mathrm{len}(A \to \alpha)$ *denote the length of a specific rule application* $A \to \alpha$ *in a parse tree, which is defined as the length of the (terminal) subword finally generated from* $A \to \alpha$*. Furthermore, for* $\alpha \in (I \cup T)^*$ *and* $n \in \mathbb{N}$*, we denote by* $c_{\alpha,n}$ *the number of different assignments of lengths to the symbols of* α *that satisfy:*
  - *Terminal symbols are always assigned a length of 1.*
  - *A nonterminal symbol* B *can be assigned any length* l *for which there is* $w \in T^l$ *such that* $\Pr(B \Rightarrow^* w) > 0$*.*
  - *The assigned lengths add up to* n*.*

  The probability of a parse tree for a word of length n *is then* $\Pr(n)$ *times the product of the probabilities of all rule applications* $A \to \alpha$ *in the tree multiplied by* $\left(c_{\alpha,\mathrm{len}(A \to \alpha)}\right)^{-1}$*.*

The factors $\left(c_{\alpha,\mathrm{len}(A\to\alpha)}\right)^{-1}$ and $\Pr(n)$ are necessary to ensure a probability distribution on the language that is generated by the LSCFG (see [WN11] for details). In fact, considering a conventional SCFG, the probability of a parse tree δ is given by

$$\prod_{A\to\alpha \text{ applied in } \delta} \Pr(A \to \alpha),$$

whereas for the corresponding LSCFG, the probability of a parse tree δ for a terminal word $w \in T^n$ is defined by

$$\Pr(n) \cdot \prod_{A\to\alpha \text{ applied in } \delta} \Pr(A \to \alpha, \mathrm{len}(A \to \alpha)) \cdot \left(c_{\alpha,\mathrm{len}(A\to\alpha)}\right)^{-1}.$$

Thus, when using LSCFG based models, we need to add the length (of rule applications) as an additional parameter for probability lookup in order to obtain $\Pr(A \to \alpha, \mathrm{len}(A \to \alpha))$, and multiply each resulting probability by a corresponding factor (the reciprocal of $c_{\alpha,\mathrm{len}(A\to\alpha)}$).

Note that in connection with RNA structure analysis, LSCFGs are interesting since they indeed make it possible to assign probabilities for a specific loop structure that depends on the size of this loop. Conventional SCFGs can model such dependencies only to a certain extent. For instance, let X denote a nonterminal symbol of some SCFG $\mathcal{G}$ that generates single-stranded regions. Hence, $\mathcal{G}$ typically defines exactly two rules with premise X, for instance $p : X \to \circ X$ and $(1-p) : X \to \circ$. Hence, any sequence of unpaired bases is generated from X according to a geometric distribution, since $\Pr(\circ^n) = p^{n-1} \cdot (1-p)$ for $n \geqslant 2$. In order to induce a different distribution (characterized by a corresponding function in n), a clever modification of the underlying grammar $\mathcal{G}$ is needed, where most distributions will not be compactly representable. Hence, plain (length-independent) SCFGs are somewhat limited in the kind of functions in the length they realize, whereas the concept of LSCFGs embraces many more distributions.

3.3.8.2. Parameter Representation

Due to the previously mentioned facts, we can easily use the following transition probabilities for the production rules of the secondary structure grammar $\mathcal{G}_s$ underlying a particular RNA grammar $\mathcal{G}_r$:

$$\Pr_{tr}(X \to \gamma, \mathrm{len} = \mathrm{len}(X \to \gamma)) := \begin{cases} \Pr_{tr}(X \to \gamma, \mathrm{len}) \cdot \frac{1}{c_{\gamma,\mathrm{len}}} & \text{if } \mathcal{G}_r \text{ length-dependent,} \\ \Pr_{tr}(X \to \gamma) & \text{else.} \end{cases}$$

Accordingly, for any two positions x_1, x_2 of a given sequence r, with $1 \leqslant x_1 < x_2 \leqslant |r|$, representing a base pair $r_{x_1}r_{x_2} \in \Sigma_r^2$, we can use the corresponding emission probability

$$\Pr_{em}(r_{x_1}r_{x_2}, \mathrm{len} = x_2 - x_1 + 1) := \begin{cases} \Pr_{em}(r_{x_1}r_{x_2}, \mathrm{len}) & \text{if } \mathcal{G}_r \text{ length-dependent,} \\ \Pr_{em}(r_{x_1}r_{x_2}) & \text{else.} \end{cases}$$

Note that any subword $r_x \in \Sigma_r$ representing a single unpaired base, $1 \leqslant x \leqslant |r|$, has length $\mathrm{len} = x - x + 1 = 1$, such that the corresponding emission probabilities will always be the same for both the length-dependent and the traditional case. Thus, we use

$$\Pr_{em}(r_x, 1) := \Pr_{em}(r_x).$$

However, for the sake of simplicity, we will mostly omit the length (second parameter), no matter if the underlying SCFG $\mathcal{G}_r$ has been trained length-dependently or in the traditional length-independent way. Only in cases where the length might be crucial for the understanding

of an actual situation or method, we will explicitly make use of that second parameter. Furthermore, to keep it even more simple, we will in many cases interchangeably utilize Pr for denoting rule probabilities for $\mathcal{G}_r$, as well as the corresponding transition and emission probabilities, rather than consistently distinguishing between Pr, $\Pr_{tr}$ and $\Pr_{em}$.

3.3.8.3. Grouping of Lengths

As proposed in [WN11], we can confine ourselves with grouping the lengths together in several intervals which allows us to store the needed transition probabilities as a vector and thus makes it possible to retrieve them in algorithms and applications without further computational efforts. Obviously, the needed emission probabilities for base pairs (and for unpaired bases) can be stored and retrieved in the same way. As indicated in [WN11], when choosing appropriate intervals, this restriction is – under certain circumstances – still powerful enough to yield a significant improvement over traditional SCFGs with respect to the prediction accuracy[17].

However, if lengths are to be grouped into intervals, we have to deal with the fact that not all such groupings yield a consistent grammar. Nevertheless, the following definition of consistency offers a sufficient condition that they do (see [WN11] for details):

Definition 3.3.5 ([WN11]) *Let* $\mathcal{G} = (I, T, R, S)$ *a CFG and* Q *a partitioning of* $\mathbb{N}$*. We call* Q consistent with $\mathcal{G}$ *if it satisfies the following condition:*

$$\forall q \in Q, i, j \in q : \exists A \to \alpha \in R, w_i \in T^i : \alpha \Rightarrow^* w_i \textit{ implies } \exists w_j \in T^j : \alpha \Rightarrow^* w_j.$$

Intuitively, to satisfy this condition, we may not group lengths i and j together if there exists a production rule that can lead to a word of length i but not to one of length j (or vice versa). The partitioning into sets of one element each (which corresponds to not grouping lengths into intervals at all) is trivially always consistent. Moreover, the interval lengths should principally grow with increasing subword length, since with increasing subword length, any training set will typically contain fewer data points per length. Hence, if the considered databases are rich enough to obtain a sufficiently large number of points per interval, we can hope for accurate estimated probabilities. Additionally, the lengths implied by the last interval should be chosen in accordance with the longest words in the training set for ensuring that the estimates for that interval do not influence the induced probability distribution.

Finally, note that as already mentioned, the danger of overfitting of the induced model is indeed much more present if length-dependent probabilities rather than their traditional length-independent counterparts are considered, since then the observations made for a particular structural motif have to be split into distinct subsets of observations according to the corresponding lengths (or length intervals). In fact, LSCFGs typically imply significantly greater numbers of free parameters than the corresponding conventional SCFGs, where the actual numbers indeed increase with growing complexity (by means of number of distinguished length intervals) of the considered partitioning of $\mathbb{N}$. This is bad, since increasing the number of parameters to be estimated requires more training data, and there is already a lack of data in connection with training (heavyweight) plain SCFGs, see Section 3.3.6.4.

[17] Note that length-dependencies can also be modeled with plain grammars, no matter if the lengths are grouped into (a finite number of) intervals or not. In fact, as stated in [WN11], any LSCFG is equivalent to a corresponding *indexed grammar* [Aho68, Gaz88] with significantly increased numbers of intermediate symbols and production rules. Hence, the concept of LSCFGs in principle only allows for a more compact representation.

3.3.9. Conditional Log Linear Models

The aim of this section is to provide some information on another class of probabilistic models, called *conditional log-linear models (CLLMs)*, which are more flexible than vanilla SCFGs, as they generalize upon the representation of conditional probabilities. In principle, these mathematical objects constitute a specialized class of structured classification models and a generalization of *conditional random fields (CRFs)*. Note that CRFs are in fact a specialized class of CLLMs whose probability distributions are defined in terms of graphical models. For details, we refer to [LMP01, SM07].

In the context of RNA folding, CLLMs describe *discriminative* models, as opposed to generative models (like SCFGs and HMMs). Briefly, a corresponding discriminative method has a model parameterized with arbitrary values and the conditional probabilities of structures given sequences are actually obtained by normalizing the Boltzmann factor over all possible structures for the sequence. As we will see, due to mathematical reasons, techniques for parameter estimation must optimize the sum of conditional probabilities of structures given sequences in the training set, rather than the sum of the corresponding joint probabilities.

3.3.9.1. Basic Concepts

Basically, CLLMs are a probabilistic framework for sequence labeling or parsing problems, which typically deal with an exponentially large space of possible input sequences, $\mathcal{X}$, and an also exponentially large space of candidate label sequences or parse trees, $\mathcal{Y}$. Formally, let $F : \mathcal{X} \times \mathcal{Y} \to \mathbb{R}^n$ be a fixed vector-valued mapping from input-output pairs to an n-dimensional feature space. Then, given a particular set of parameters $w_1, \ldots, w_n$ (as an n-dimensional vector w^T), the corresponding CLLM models the *conditional* probability of candidate $y \in \mathcal{Y}$ given input $x \in \mathcal{X}$, defined as follows[18]:

$$\Pr(y \mid x; w) = \frac{\exp\left(w^T \cdot F(x, y)\right)}{Z(x)},$$

where

$$Z(x) = \sum_{y' \in \mathcal{Y}} \exp\left(w^T \cdot F(x, y')\right)$$

is the normalizing constant for input sequence x.

Thus, like traditional SCFGs, CLLMs are probabilistic models for defining *conditional* probability distributions on possible RNA secondary structures s given a particular sequence r. To illustrate the differences of both formalisms in connection with probabilistic modeling of RNA secondary structure, let us first consider a feature-based representation of SCFGs in order to highlight several important assumptions made when modeling with SCFGs (as done in [DWB06]). Therefore, let d be a particular derivation tree of a given RNA sequence r according to some SCFG $\mathcal{G}$. If the number of production rules of $\mathcal{G}$ is equal to n, then $F(r, d) \in \mathbb{R}^n$ denotes an n-dimensional *feature vector* whose i^{th} dimension, given by $F_i(r, d)$, indicates the number of times the i^{th} rule is used in leftmost derivations corresponding to d. If we denote the probability of this rule p_i and the corresponding log probability[19] by $w_i = \ln(p_i)$, the joint likelihood of

[18] Note that in linear algebra, a *row vector* or $1 \times n$ matrix v^T is the transpose of the *column vector* or $n \times 1$ matrix v such that $v^T_{1,i} = v_{i,1}$ for $1 \leqslant i \leqslant n$. The *dot product* $a \cdot b$ of two vectors a and b is equivalent to multiplying the row vector representation of a by the column vector representation of b.

[19] Note that the base of the logarithm must be Euler's number e here. In fact, we need to use the natural logarithm function $\ln = \log_e$, which represents the inverse function of the exponential function exp.

the sequence r and derivation tree d can be rewritten in *log-linear* form as follows [DWB06]:

$$\Pr(r, d) = \prod_{i=1}^{n} p_i^{F_i(r,d)} = \exp\left(\ln\left(\prod_{i=1}^{n} p_i^{F_i(r,d)}\right)\right) = \exp\left(\sum_{i=1}^{n} F_i(r, d) \cdot \ln(p_i)\right) = \exp\left(w^T \cdot F(r, d)\right),$$

such that according to equation (3.4), we get

$$\Pr(\mathcal{S} \mid r) = \frac{\sum_{d \in \mathcal{S}} \exp\left(w^T \cdot F(r, d)\right)}{\sum_{d' \in \mathcal{F}(r)} \exp\left(w^T \cdot F(r, d')\right)}. \tag{3.8}$$

As stated in [DWB06], this alternative form perfectly demonstrates that SCFGs are actually log-linear models with the restrictions that

- the parameters $w_1, \ldots, w_n$ correspond to log probabilities and hence obey a number of constraints (like for instance, all parameters must be negative), and
- the features $F_1(r, d), \ldots, F_n(r, d)$ derive directly from the grammar, such that the types of features are restricted by the complexity of the grammar.

Note that both restrictions can be removed if one simply wishes to ensure that the conditional probability according to equation (3.8) is well-defined, which immediately lies the foundation for the CLLM framework. In fact, the major advantage of CLLMs over vanilla SCFGs is due to the fact that

- the parameters $w_1, \ldots, w_n$ may take on any real values, and
- the features $F_1(r, d), \ldots, F_n(r, d)$ are similarly unrestricted.

Hence, unlike conventional SCFGs, CLLMs have the generality to represent complex scoring schemes, such as those used in modern physics-based secondary structure prediction tools.

3.3.9.2. Parameter Estimation

According to equation (3.8), the conditional probability $\Pr(\mathcal{S} \mid r)$ is defined as a log-linear function of the feature vectors $F(r, d)$ of a particular CLLM. However, CLLMs provide no manner for calculating the joint probability $\Pr(r, \mathcal{S})$. This is due to the fact that by definition, CLLMs do not model the joint distribution of r and $\mathcal{S}$ – they only model the distribution of $\mathcal{S}$ *given* r. For this reason, straight maximum likelihood techniques – as used for optimizing such joint probabilities on the basis of traditional SCFGs – can obviously not be applied to CLLMs.

For training CLLMs, one therefore has to rely on the *conditional maximum likelihood (CML)* principle. That is, one needs to calculate the parameter vector $w_{\mathrm{CML}} \in \mathbb{R}^n$ that maximizes the *conditional likelihood* of the candidate structures given the input sequences, which is formally defined as follows [DWB06]:

$$\ell_{\mathrm{CML}}(w \mid \mathcal{D}) = \prod_{i=1}^{m} \Pr(y^{(i)} \mid x^{(i)}; w),$$

where $\mathcal{D} = \{(x^{(1)}, y^{(1)}), \ldots, (x^{(m)}, y^{(m)})\}$ is the considered training set. For details, we refer to [JR91, Kro94].

Note that conditional likelihood training is mostly referred to as *discriminative* training in literature, since the objects to be trained are classified as discriminative models, as opposed to generative models. However, the mechanics of performing the probabilistic inference tasks required in the optimization of the conditional likelihood function ℓ_{CML} follow closely the traditional inside and outside algorithms for SCFGs [DEKM98].

Thus, just like SCFGs, CLLMs enjoy the ease of computationally-driven parameter learning. In fact, many of the benefits of the discriminative learning approach can be obtained for SCFGs by converting an SCFG to a corresponding CLLM (by removing restrictions on the parameter vector w) and training via conditional likelihood. It should be clear, however, that such conversions do not take full advantage of the expressivity of CLLMs. In particular, the ability of CLLMs to use generic feature representations means that in some cases, CLLMs can conveniently represent models which do not have compact parameterizations as SCFGs [DWB06].

3.3.9.3. CLLMs for Structure Prediction

As suggested in [DWB06], for prediction problems, conditional maximum likelihood (or discriminative) training is arguably more natural than maximum joint likelihood (or generative) training, as it focuses on finding parameters that give good predictive performance without attempting to model the distribution over input sequences. Actually, the complex scoring terms of state-of-the-art thermodynamic models can be transferred to CLLMs without any difficulties.

For instance, a modern secondary structure prediction tool based on CLLMs, named `CONTRAfold` [DWB06], uses a simplified `Mfold`-like scoring scheme in order to incorporate features chosen to closely match the energetic terms found in standard physics-based models of RNA secondary structure. In fact, the `CONTRAfold` secondary structure model extends and simplifies traditional energy-based scoring schemes while retaining the parameter learning ease of common probabilistic methods.

Basically, `CONTRAfold` uses estimated RNA scores (rather than the corresponding free energy parameters) which are obtained by maximizing the conditional likelihood of a set of known structures. However, the training database for the `CONTRAfold` v1.0 software package was fairly small (only 151 structures, each from one of the different families tagged as "published" by the Rfam database [GJMM+05], the so-called "S-151Rfam" set[20]. The latest `CONTRAfold` v2.02 package is much more efficient, implementing a gradient-based algorithm to learn multiple regularization parameters [DFN07]. Moreover, it was actually trained on a much larger set of known structures than the original version (the "S-Processed-TRA" set from [ACH+07] which contains a mixture of 3,439 sequences of various RNA types).

Evaluations show that `CONTRAfold` (even in its original version) provides a rather high single sequence prediction accuracy. Actually, `CONTRAfold` in several cases manages to provide the highest single sequence prediction accuracy to date, closing the performance gap between the best physics-based and the best probabilistic RNA structure prediction methods [DWB06]. However, there are some benchmarks that show better performance by other methods, suggesting in the least that the performance of structure prediction can vary considerably depending on RNA family [HKS+09, LGM09, ACH+10]. Notably, following `CONTRAfold`, several other statistical methods have been subsequently developed, such as for instance *constraint generation* [ACH+07] or `ContextFold` [ZGEZU11]. These are all classified as discriminative statistical methods which implement different variants of standard thermodynamic models[21].

In fact, to date, the best RNA folding methods are discriminative. This might be due to the facts that on the one hand, strict thermodynamic approaches are inherently limited by the experimentally determined energy parameters. On the other hand, SCFGs for RNA folding generally use rather simple designs that do not implement many of the complex features of RNA secondary structure as considered by standard thermodynamic models (see

[20] This structural data set is currently available under `http://www.rnasoft.ca/CG/data/S-151Rfam.txt` and will, among others, be used in the sequel for training and evaluation of our probabilistic methods.

[21] Basically, these methods condition on a set of RNA sequences being given (in order to obtain estimates for the free energy parameters, see Section 3.3.10), whereas a generative SCFG approach models the probabilities of the input RNA sequences (in order to induce corresponding ensemble distributions).

for instance [KH99, DE04]), albeit the parameterization of an SCFG depends only on the existence of enough trusted structures.

3.3.10. Statistical Estimation of Thermodynamic Parameters

As discussed throughout Section 3.2, the accuracy of predictions made by physics-based approaches is inherently dependent on the quality of the thermodynamic parameters used in the underlying (incomplete) free energy model. Notably, it is known that structure predictions might be improved through kinetic modeling of RNA folding [DCCG04], and through improvements in the accuracy of the set of experimentally derived free energy parameters. However, for several reasons described in Section 3.2.1.3, current thermodynamic models still lag behind the implementation of these demands, such that the quality of any physics-based method is not only strongly dependent on but also evidentially limited by the energy model used.

In fact, only if the free energy functions were complete, and if all thermodynamic parameters were accurate, the biological structure could often be expected to have the lowest energy among all possible foldings for a given sequence, which would inevitably yield an accuracy gain for any MFE based algorithm. However, even if the set of experimental thermodynamic parameters is expected to continue to improve, the vision of such an ideal model of biophysics is utopian.

Therefore, an increasingly accepted and highly appreciated alternative is to statistically estimate the energy parameters by taking advantage of known structural information from reliable RNA structure databases. This problem can be formally described by writing the free energy of a given pair of RNA sequence r and secondary structure s in the following way:

$$\Delta G(r, s, \Phi) = g(r, s)^T \Phi = \sum_{i=1}^{n} g_i(r, s)\Phi_i,$$

where n is the number of features (for instance distinguished energetic terms for the various structural motifs), $g_i(r, s)$ is the number of times feature i occurs in secondary structure s of sequence r, and Φ_i is a parameter modeling the energy contribution of each occurrence of feature i. One typically uses the features proposed in [XSB+98, MSZT99], as those are widely accepted as biologically realistic and therefore considered in most of the modern state-of-the-art software packages for RNA folding.

However, for any considered set of features, we are faced with the problem of estimating the model parameters Φ such that they match as closely as possible the data given in a particular sample set $\mathcal{D}$. In our context, the data set must consist of pairs (r, s), where s is the true (MFE or something close to it) structure of sequence r (as determined using trusted and highly accurate methods). Different approaches are known for estimating the model parameters Φ from such a set of pairwise RNA data. Some of them will be discussed in brief in the following subsections. For a recent more comprehensive summary on computational methods that have been used in the literature to infer RNA energy parameters, we refer to [And08].

3.3.10.1. Conditional Maximum Likelihood

One of the most prominent and straightforward techniques would be to estimate the parameter vector Φ_{CML} that maximizes the conditional likelihood of $\mathcal{D}$ (as described in Section 3.3.9.2). However, this approach is rather slow, such that is can hardly be applied to large training sets. This is quite problematic in practice, since the most widely used Turner energy model has hundreds of parameters, such that a robust parameter estimation scheme should be able to efficiently handle large data sets with thousands of structures. Moreover, it should be capable of using available experimental free energy data in addition to structural data for training attempting for more reliable results.

3.3.10.2. Constraint Generation

Due to these reasons, an alternative computational approach to RNA free energy parameter estimation, named *constraint generation (CG)*, has been devised [ACH+07], which is substantially faster than a conditional maximum likelihood method on relatively small training sets, and hence can be practically used on large training sets with thousands of trusted structures. In fact, the constraint-based parameter estimation algorithm was the first to efficiently combine structural and thermodynamic RNA secondary structure data. Note that there are several techniques similar to this CG approach, for instance the *large margin approach* [Tas05, TCKG05] or *nonlinear inverse optimization* [LHP05].

It should also be mentioned that in [ACH+07], it was shown that using the CG method on biologically sound data, revised parameters for the standard Turner energy model [MSZT99] can be obtained which actually yield significant improvements over current state-of-the-art methods in terms of accuracy of predicted foldings.

3.3.10.3. Bayesian Statistical Inference

Bayesian inference has also become highly appreciated in connection with RNA folding problems. In fact, the Bayesian statistical inference approach appears to be well suited to the problem of estimating thermodynamic parameters from RNA structure databases, since it has been shown that the primary sequence data, the secondary structure and the free energy parameters can be described together through Bayesian statistical modeling, where inference on any variable can be made from the knowledge of the other two variables (see [DL99]).

Consequently, inferences on any of the diverse parameters used in standard thermodynamic models can be made on the basis of structural information, that is trusted RNA secondary structures with annotated sequences (typically derived from comparative analysis). Conceptually, inferences for thermodynamic parameters can be performed for any secondary structure in a database or for a set of structures, and the experimentally derived Turner parameter values can be used for prior specification. It is, however, advantageous to pool information from diverse structures, for improved accuracy in the estimates [Din06]. Nevertheless, for the large number of parameters used in state-of-the-art free energy models, the high dimensionality of the estimation problem will pose computational challenges.

3.3.10.4. Consequences

It should be clear that the application of training methods like the Bayesian inference approach finally makes it possible to derive energy estimates that are suited for structure prediction. If applied to a reliable set of RNA data from a single biological class, it may also manage to indirectly incorporate non-energetic effects (like, for instance, modified nucleotides that present problems for tRNA structure predictions [RCM99]) into the model, since those are actually observed in the trusted data set and thus may alter the energy parameters derived.

Furthermore, note that due to the existence and increasingly accepted use of learning techniques for the thermodynamic models, it generally no longer holds that energy approaches are based on a standard physical model, whereas SCFG based methods are based on data learning. In fact, with the advance of machine learning approaches for the free energy models, the distinction between SCFG based approaches and thermodynamic methods can arguably no longer be maintained. More or less, statistical inference approaches to the energy based models just use the original parameters to reduce the search space, and to prevent overfitting (which can be avoided partially by using penalized approaches). Otherwise, the parametrization is rich enough that one can consider this fully as a data-driven approach.

An extreme example is given by [ZGEZU11], where a move to much richer parameterizations is proposed. Briefly, the applied scoring models constitute a refinement of previous models, examining more types of structural features and especially a larger sequential context for each feature.

Anyway, it should be clear that a potential disadvantage of statistical parameter estimation over the use of the standard thermodynamic parameters is that in any case, the accuracy of the estimated parameters strongly depends on the quality of the employed data. For instance, thermodynamic parameters suited for a particular class of RNAs may not reliably be estimated if there exists hardly knowledge on these RNAs in the form of trusted databases. Actually, with respect to the training data, we have to face basically the same problems as in connection with SCFG based approaches (if a similar complex grammar design is considered).

In fact, the approach of [ZGEZU11] is made possible in the first place due to the availability of large training sets (such as the RNA STRAND database [ABHC08]), combined with advances in machine-learning [Col02, CDK+06]. Notably, it is supported in practice by recent accelerations to RNA folding algorithms which will be discussed in Section 3.4.4.

3.3.11. Summary

Due to the limitations of standard thermodynamic models, including the inability to capture certain relevant physiological and chemical aspects in their parameters, probabilistic models derived on the basis of structural information obtained from reliable RNA databases have emerged as an alternative methodology towards single sequence RNA secondary structure prediction. The corresponding DP algorithms basically compute either the Viterbi parse (corresponding to the most likely structure) or the MEA parse (corresponding to the structure with highest expected number of correctly predicted positions). Just like their physics-based counterparts, they inherently require $\mathcal{O}(n^3)$ time and $\mathcal{O}(n^2)$ storage for an input sequence of length n. Popular software packages implementing such probabilistic approaches are for instance `Pfold` [KH99, KH03] or `CONTRAfold` [DWB06].

Originally, the class of all feasible secondary structures (that obey to certain structural constraints like for example the non-existence of isolated base pairs) was described by a corresponding SCFG, which induces a (non-uniform) probability distribution on the considered class. However, over the years several extensions and generalizations of SCFGs have also proven useful in this context, like for instance LSCFGs which naturally imply a more specific probability distribution on the modeled structure, or CLLMs which are a generalization of traditional SCFGs according to the following facts: While SCFGs (like HMMs) are generative probabilistic models, which are intuitive and allow convenient (generative) parameter training via maximum *joint* likelihood techniques, CLLMs are discriminative probabilistic models, where the parameters are learned by (discriminative) training methods maximizing the *conditional* likelihood.

As a consequence, CLLMs have the power to represent more complex scoring schemes than the corresponding SCFG can represent, for instance a `Mfold`-like energy based one as considered in [DWB06]. However, generative probabilistic models are generally easier to train and to use than discriminative variants: Principally, generative (SCFG based) training can easily be realized by counting the observed frequencies of applications of the distinct production rules of an unambiguous SCFG (yielding a maximum likelihood estimator), by expectation maximization or similar methods from machine-learning. That way, the resulting estimates of the grammar parameters are adapted to a particularly considered data set.

In any case, there are two important issues that inherently arise when estimating the corresponding set of probabilistic parameters: the resulting model may suffer from overfitting or lack of generalization. In fact, the features of CLLMs used for structure prediction usually

model the (energy contributions of) various structural motifs that occur in RNA secondary structure, such that a rich and reliable training set providing enough structural information is required in order to avoid overfitting, just like in case of SCFGs of comparable complexity.

In many supervised learning methods, overfitting is actually controlled through the use of regularization penalties for limiting model complexity. For instance, methods for statistical estimation of thermodynamic parameters, for example Bayesian inference approaches (see Section 3.3.2.2), use the experimentally obtained Turner values [MSZT99] as an a priori specification. However, in connection with common probabilistic approaches that completely abstract from free energies, there is no *lab-based* prior to the grammar parameters like the standard Turner model for physics-based approaches. Therefore, the corresponding distribution (on the modeled structure class) has to be derived exclusively from a collection of sample data, more specifically real-life RNA data (given by pairs of RNA sequences with annotated secondary structures).

Nevertheless, it should be clear by now that the dependence of the resulting accuracy on the used model parameters is not only a major problem in connection with probabilistic approaches that rely exclusively on structural information from reliable databases, but it indeed also limits the performance of physics-based methods. In fact, the (usually many hundreds of) thermodynamic parameters used in standard state-of-the-art energy models for RNA folding are mostly estimated from experimental results (on the basis of diverse structural RNAs) and are far from being perfect, since folding processes of RNA molecules are usually to a large part controlled by a number of additional non-energetic effects. Those might only be accounted for by estimating the thermodynamic parameters from reliable sets of real-life data, but then the same problems concerning parameter estimation arise as in case of probabilistic prediction methods based on SCFGs or similar models.

3.4. Concluding Remarks

To complete this chapter, we want to provide some additional information on existing related methods for computational RNA structure prediction that might be nice to know with respect to the global context of this thesis.

3.4.1. Pseudoknot Prediction

It should be noted that all the above mentioned DP algorithms only work for secondary structures without pseudoknots, as they consider only the mutual planar arrangements of base paired helices and are restricted to generating only non-crossing base pairs (crossing pairs are prohibited). A short review on how RNA folding algorithms work and why they can't deal with pseudoknots can be found in [Edd04].

In fact, the problem of RNA secondary structure prediction including arbitrary pseudoknots is $\mathcal{NP}$-complete, even for rather simple models of free energy underlying the corresponding algorithms (see [Aku00, LP00]), and no known search procedure can solve it in less than exponential time in the worst-case. However, much effort has been put into the development of polynomial-time algorithms for predicting RNA secondary structures that contain certain restricted classes of pseudoknots, but due to their quite high time and space complexities, these algorithms are principally not applicable for long sequences.

As a result, a number of DP algorithms for predicting different classes of RNA pseudoknotted structures characterized in terms of recursion equations and/or stochastic grammars were devised. These include MFE approaches (such as [RE99, Aku00, RG04]), PF methods (for example [DP03, DP04]), grammar based variants (for instance [KSK06, MSS05]), and topological methods (such as [RHA+11]). The inter-relationships of some of the considered RNA structures

classes have been clarified partially in [CDR+04]. A unified algebraic characterization of many classes has been provided recently in [NW].

Notably, a rather general MFE based DP algorithm for predicting pseudoknotted RNA structures, which is capable of predicting nearly all known classes of pseudoknots, was presented by Rivas and Eddy [RE99]. However, this algorithm has a theoretical worst-case complexity of $\mathcal{O}(n^6)$ in time and $\mathcal{O}(n^4)$ in storage for n the length of the RNA sequence to be folded and is thus only practical for very short RNA sequences.

Furthermore, various heuristic prediction methods dealing with pseudoknotted RNA structure have been proposed in the literature in order to speed up computations. For example, the algorithm devised by Metzler and Nebel [MN08] relies on a *Markov-Chain Monte-Carlo (MCMC)* sampling approach rather than on exact MFE calculations. This method is not only more general than the one presented in [RE99], but it also has a better theoretical worst-case complexity.

3.4.2. Comparative Methods

Obviously, the algorithms and methods discussed within this chapter so far are all connected to the classical RNA folding problem, that is predicting the secondary structure of a single RNA strand which is given as an input. Nevertheless, if a set of homologous RNA sequences is available, then comparative structure analysis can be applied for accurate identification of functional RNAs and determination of RNA secondary structures [PTW99]. In principle, there are three different approaches to automated comparative RNA sequence analysis.

Particularly, many of the comparative methods require a multiple sequence alignment as input, which is then used to infer a consensus secondary structure (see, for instance [KH99, HFS02, WH04, PBS+06]). However, the multiple alignment step requires data sets with high sequence homology. Furthermore, functional RNAs are not necessarily conserved on their primary sequence level and hence, an alignment based on RNA strands alone is generally not sufficient for the detection of conserved secondary structure [RE00].

This has motivated the use of Sankoff's algorithm [San85] for simultaneous alignment and structure prediction of homologous structural RNA sequences. Note that the problem of simultaneously folding and aligning homologous sequences is sometimes referred to as *co-folding* problem (see, for instance [ZUGVWS08]). In principle, the Sankoff co-folding algorithm combines recursions for sequence alignment (such as [WS86]) with those of classical RNA folding algorithms (particularly [NJ80]) and takes into account both sequence and structure homology. However, its practical applicability is highly limited due to its prohibitive computational cost of $\mathcal{O}(n^{3\cdot m})$ in time and $\mathcal{O}(n^{2\cdot m})$ in space for m sequences of length n.

In cases where no helpful level of sequence conservation is observed, the sequence alignment step is usually excluded and instead, a single secondary structure is predicted separately for each sequence (or subgroup of sequences). A set of known homologous RNA secondary structures can then be used to directly derive a multiple structure alignment. A comprehensive comparison of comparative RNA structure prediction approaches, including an overview of available algorithms for supporting comparative studies, can be found in [GG04].

Anyway, it should be noticed that with respect to accurate RNA comparative structure analysis, the Sankoff co-folding algorithm is of great importance. Therefore, many articles have suggested practical heuristic methods for speeding up Sankoff's alignment process, where some current implementations make restrictive assumptions in that they impose pragmatic limits on the size or shape of the RNA substructures [GHS97, MT02a, HBS04, Hol05, DE06, THG07]. In addition to these heuristic approaches, a non-heuristic speedup could recently be reached [ZUGVWS08]. In principle, that algorithm retains the Sankoff style recurrence. The speed up is based on some simple pruning of the branching points as suggested in [WZZU07], yielding a time reduction by a linear factor on average without compromising optimality. Notably, this concept was

earlier applied in connection with standard RNA folding algorithms; details will follow in Section 3.4.4.

3.4.3. Local Prediction

The classical algorithms for RNA secondary structure prediction as discussed throughout this chapter are named *global* approaches. This means that structures are considered for the entire RNA molecule and there is no restriction on the span of base pairs. Basically, global RNA foldings algorithms have two major limitations: First, the algorithms are computationally demanding when being applied to very long sequences, limiting their practical use for genome-wide applications. Second, their prediction accuracy decreases with sequence length [DCCG04, DE04]. In fact, a major challenge in connection with global folding is the correct prediction of long-ranging base pairs [DCCG04].

When dealing with long RNA sequences, a sensible alternative is thus to use *local* prediction methods, see [LMM+12]. Actually, some local folding approaches have been proposed over the past years to account for the challenges implied by global methods. Briefly, one of these approaches keeps structures local by restricting the maximum distance allowed between the two nucleotides that form a base pair (see, for instance [HPS04, KKA08]). Another local prediction approach is essentially window-based in order to further accommodate the uncertainty of global structure by multiple stabilizing and destabilizing factors (see, for instance [BHS06]). Notably, all local folding algorithms can be implemented to run in linear time with respect to sequence length and they are easily applicable on a genome-wide scale [LMM+12].

3.4.4. Accelerated Methods

As we have seen in this chapter, the basic RNA folding problem of predicting a secondary structures without pseudoknots for a single input RNA sequence of length n can easily be solved in $\mathcal{O}(n^3)$ time and with $\mathcal{O}(n^2)$ storage requirements by applying a particular DP algorithm. However, improving the computational costs of accurate RNA secondary structure prediction is a challenging task and nowadays has become an area of active research. In fact, the problem of developing accelerated methods for solving the RNA folding problem is motivated from both the theoretical and the practical point of view, since it is often solved multiple times in the inner loop of more complex algorithms, as well as for long RNA molecules in the study of RNA genomes (see Section 3.4.3).

Over the past years, several algorithmic approaches were applied to improve the complexity of standard RNA folding methods. Notably, in most cases *sparsification* techniques tailored for reducing the time and space complexity of MFE based DP algorithms were applied [EGGI92a, EGGI92b, WZZU07, BTZZU11, DB12]. In brief, sparsification allows to discard large portions of DP matrices without loosing optimality. The basic idea is to prune certain computation paths encountered in the DP recursions. Sparsification improves the expected running time and space usage of RNA related DP algorithms without introducing additional restrictions on the structure class handled or compromising the optimality of solutions.

For instance, the approach of [WZZU07] yields an expected time complexity of $\mathcal{O}(n^2 \cdot \psi(n))$, where $\psi(n)$ is shown to be constant *on average* under standard polymer folding models. Essentially, sparsification was applied on the time-consuming multiloop computations, by exploiting the observed *triangular inequality* property of DP matrices commonly used in RNA secondary structure prediction. In particular, the multiloop computations are executed conditionally and the optimal closed substructures are kept in a so-called *candidate list* for later use. The crucial point is that there are only relatively few candidates, which in turn implies a significant

reduction in time and space complexity. Note that in [WZZU07], a *polymer-zeta*[22] behavior of RNA folding with respect to increased sequence length is assumed, yielding the conclusion that sparsified MFE folding can achieve a (roughly) linear reduction in the time complexity on average – without sacrificing the optimality of the results (no heuristics are used). Although experimental work has shown that the distribution of RNA foldings obeys the polymer-zeta property [KS05, KMP00], the results of a recent study show that the polymer-zeta property does not apply for RNA structures [HR]. In accordance with these facts, the complexity of those algorithms is typically expressed in terms of a sparsity parameter Z that satisfies $n \leqslant Z < n^2$ for n the length of the input sequence, implying worst-case bounds for time and space of $\mathcal{O}(n \cdot Z)$ and $\mathcal{O}(n^2)$, respectively [BTZZU11]. Notably, the results from [WZZU07] were extended in [ZUGVWS10] to the RNA co-folding problem (see Section 3.4.2).

Moreover, using the observations of [WZZU07], it was shown in [BTZZU11] how the space complexity for the RNA folding problem could also be improved using sparsification, resulting in $\mathcal{O}(n^2 + P \cdot Z)$ time and $\mathcal{O}(Z)$ space requirements for $P < n \leqslant Z \leqslant n \cdot (P + 1)$, where P is an additional sparsity parameter. In principle, the space reduction is achieved by combining two independent sparsification criteria that effectively restrict the number of expressions to be considered in bottleneck computations. This actually builds on the observation that some solutions for substructures are not examined after a certain stage of the algorithm and may hence be discarded from memory. Notably, this sparsification technique has also been successfully applied to the RNA co-folding problem [BTZZU11]. Since no complex data structures or subroutines are used (which might imply long computation times in practice), the algorithms presented in [BTZZU11] provide practical speedups.

It is worth mentioning that sparsification has also been applied in the context of RNA pseudoknot structures [MSW+10] as well as for RNA-RNA interactions [SMW+10]. In contrast to prediction algorithms for unknotted RNA secondary structures, however, not every decomposition rule in the DP recurrences for RNA pseudoknot structures is amendable to sparsification [HR]. Note that by construction, acceleration techniques based on sparsification are applicable only to optimization problems, such as MFE structure prediction. It will not work for non-optimization problems, like RNA partition function related problems or the inside-outside algorithm for SCFGs – in which the goals are to compute the sum of scores of all solutions of the input, instead of computing the score of an optimal solution. This is due to the fact that the corresponding DP methods need to take into account all substructures (see Section 3.2.3).

However, to improve the complexity of algorithms for the non-optimization problem variants, such as partition function calculations or inside-outside computations, Valiant's approach [Val75] might be applied. In fact, this is the only technique currently known to reduce the running times of such algorithms. Although Valiant's classical algorithm only solves the problem of CFG recognition in subcubic time [Val75], he suggested that it could be extended to additional related problems. A known extension of the technique is that to SCFG parsing and RNA folding [Aku99, BS07].

It should be mentioned that with the (MFE and SCFG based) algorithms suggested in the theoretical paper [Aku99], a slight worst-case speedup of $\mathcal{O}(n^3 \cdot \log(\log(n))^{1/2} / \log(n)^{1/2})$ time can indeed be reached, but its practicality is unlikely and unestablished. Nevertheless, Valiant's approach has recently been used for deriving generic algorithms that can be applied to reduce the theoretical asymptotic worst-case time bounds for a large family of important problems within the world of RNA secondary structures and CFGs [ZTZU11].

Anyway, another general approach that leads both to a theoretical (by means of worst-case) and to a practical speedup of many DP algorithms is known as *Four-Russians* technique, which goes back to an article [ADKF70] concerning boolean matrix multiplication. The main idea

[22] The polymer-zeta property basically states that the probability of two bases to be paired with each other decreases with their distance in the RNA chain, that is there are only few long-range base pairs.

is to partition the matrix into small square blocks, and to use a lookup table to perform the algorithm quickly within each block. In [FG10a], that technique, has been applied to the RNA folding problem in oder to reach an improved slightly subcubic worst-case time complexity of $\mathcal{O}(n^3/\log(n))$ for sequence length n. Particularly, a simple, complete and practical Four-Russians algorithm for the simplest case of the RNA folding problem, that is finding a maximum cardinality, non-crossing, matching of complimentary nucleotides in an RNA sequence, is presented. Notably, it is shown that the improved time bound can also be obtained for richer nucleotide matching scoring schemes, and that the method actually achieves significant speedups in practice.

In this context, it is worth mentioning that the algorithm devised in [FG10a] relies on some technical insights from [GWR80] which gives a Four-Russians solution to the problem of context-free language recognition. Nevertheless, it is stressed that the achieved speedup could not be reached by simply reducing RNA folding to context-free parsing and then applying the Four-Russians technique devised in [GWR80], since only a reduction of RNA folding to stochastic context-free parsing is known [DEKM98]. It remains to note that the Four-Russians idea has also been successfully applied in order to achieve a practical and worst-case speedup for the RNA co-folding problem [FG10b].

In summary, several different approaches have been employed for developing accelerated methods for single sequence RNA secondary structure prediction, but they generally retain the $\mathcal{O}(n^3)$ worst-case time bound for n the length of the input sequence. To date, no computational method (either based on a general thermodynamic model or on a specific probabilistic model derived from structural data) has been devised that actually manages to yield a significant (at least linear) reduction of the time complexity in the *worst-case*, while still ensuring an acceptable predictive accuracy.

Part II

Results

Chapter 4

Research Plan

This chapter provides an outline of the research plan that will be pursued in Chapters 5 to 9.

The main goal of the present thesis is the development and design of efficient algorithms for predicting the secondary structure of an RNA molecule from a single input sequence, where pseudoknots are deliberately not allowed. To achieve this, we want to apply two widely used techniques known from computer science, namely random sampling and approximation, in connection with an appropriate stochastic context-free modeling of RNA secondary structure. Accordingly, the following objectives will be pursued in detail throughout the following chapters:

1. Implementation of an efficient sampling method that relies on (weighted) unranking for generating random RNA secondary structures of a given fixed size (according to a native distribution):
   - Sequence-independent approach (abstraction from primary structure of a molecule),
   - quality of generated structures provides feedback on the suitability of elaborate SCFG models (like the one considered) to be used as basis for the subsequent design steps.
2. Design and evaluation of a probabilistic statistical sampling algorithm for RNA secondary structure prediction:
   - Sequence-dependent approach (primary structure of the molecule is considered),
   - generation of candidate foldings by consecutively sampling base pairs according to conditional distributions implied by the inside and outside probabilities for the input sequence (derived using a sophisticated (L)SCFG similar to that of step 1),
   - different ways are presented for deriving predictions from generated sets of candidate foldings;
   - main focus of attention is on quality of predicted structures compared to other RNA folding approaches.
3. Empirical studies on the effect of disturbed distributions on the quality of samples and predictions derived according to step 2:
   - Indicates how precise the probabilities need to be approximated (see design step 4).
4. Reduction of the worst-case time complexity for single sequence structure prediction based on statistical sampling:
   - Design of efficient methods for computing approximations of needed sampling probabilities (using the observations of step 3),
   - empirical evaluations based on comparisons with prediction results derived by exact variants (algorithms designed in step 2) and by other leading tools.

The realization of this research plan is divided into Chapters 5 to 9 in a straightforward way, which we now want to describe in detail.

Chapter 5:
Random Generation of RNA Secondary Structures According to Native Distributions

The main goal of this chapter is the derivation of a new and efficient algorithm for the random generation of RNA secondary structures of a given preliminary fixed size n according to a native distribution on all feasible structures having this size. For this purpose, we use and generalize the weighted unranking approach from [WN10] in connection with an elaborate and rather complex (by means of grammar rules) SCFG model that distinguishes between the same structural features as standard thermodynamic models.

Note that for generating RNA secondary structures of size n, we take on a sequence-independent point of view. This means we abstract from the actual sequence of nucleotides in the molecule and consider only its length. This corresponds to a reduction of the problem of predicting a secondary structure of an RNA molecule of length n to the simplest case where all base pairs are allowed and actually equiprobable.

The contribution of this chapter with respect to the main goal of this thesis, the design of efficient probabilistic RNA foldings methods based on random sampling, is that the resulting quality of randomly generated structures (for different sizes of n) provides feedback on the power of complex SCFG models to capture the expected shapes of RNA molecules and thus on their applicability in the context of statistical sampling from the ensemble of feasible structures for a given input sequence.

Chapter 6:
Evaluation of a Sophisticated SCFG Design for RNA Secondary Structure Prediction

In this chapter, we design and evaluate a sophisticated SCFG (similar to that used in Chapter 5) that mirrors current thermodynamic models applied in modern physics-based RNA secondary structure prediction methods. Particularly, this rather complex SCFG represents an exact probabilistic counterpart to the energy model employed for calculating the needed PFs for the sampling strategy implemented in the `Sfold` tool.

We effectively use that elaborate SCFG design as foundation of a corresponding sampling method that samples possible foldings of a given RNA molecule rigorously from the induced probability distribution. In principle, this SCFG based algorithm produces a statistically representative sample of secondary structures for a given input sequence in proportion to the distribution on the entire ensemble of feasible foldings, which is implied by the learned grammar parameters. It actually heavily relies on the corresponding inside and outside values for the sequence. Thus, this sampling method represents a probabilistic counterpart to the energy-based PF variant of `Sfold`, where structures are sampled in proportion to their Boltzmann weights, guaranteeing a statistical representation of the Boltzmann-weighted ensemble.

Nevertheless, for adequate evaluations of SCFG based statistical sampling by means of predictive accuracy compared to other approaches, several methods for constructing or extracting predictions from arbitrary sample sets need to be designed that are strongly related to known techniques.

Chapter 7:
Statistical Sampling Based on a Length-Dependent SCFG Model

In this chapter, we describe how to extend the SCFG based statistical sampling method studied in Chapter 6 to additionally incorporate length-dependencies, yielding a corresponding LSCFG variant that samples possible foldings of a given RNA molecule recursively from the induced probability distribution. Accordingly, this algorithm is capable of producing a statistically representative sample of secondary structures for a given RNA sequence in proportion to the distribution on the entire ensemble of feasible foldings, where the corresponding distribution is immediately implied by the learned length-dependent grammar parameters. It thus makes heavy use of the corresponding length-dependently derived inside-outside values.

Just like the conventional variant originated from Chapter 6, this LSCFG method represents a probabilistic counterpart to the energy-based PF variant of `Sfold`. Therefore, large parts of this chapter are dedicated to fundamental studies on the differences in resulting sampling quality obtained with all three approaches.

Chapter 8:
Evaluating the Effect of Disturbed Ensemble Distributions on Statistical Sampling

This chapter marks the point in the present thesis where the major focus of attention turns from the mere accuracy of predicted foldings and generated samples to the reduction of the running time in connection with predicting secondary structures of competitive (or at least acceptable) quality. In fact, the aim of this chapter is to prove or disprove the hypothesis that the concept of approximation known from theoretical computer science could be applied for improving the worst-case complexity of statistical RNA secondary structure sampling.

Therefore, we perform a comprehensive experimental analysis on the effect of disturbances in the ensemble distribution for a given sequence to the quality of corresponding sets of candidate structures generated with the (L)SCFG based statistical sampling method studied in Chapters 6 and 7. Basically, (variations of) two different levels of errors are considered for randomly creating disturbances on the needed sampling probabilities for a given input sequence according to the underlying grammar model: relative and absolute ones. This way, we hope to get an indicator on how precise the probabilities need to be approximated in order to still be able to obtain satisfactory prediction results.

Chapter 9:
Heuristic Statistical Sampling Methods for Efficient Secondary Structure Prediction

In this chapter, we finally develop innovative new variants of the (L)SCFG based statistical sampling approach for predicting the secondary structure of RNA molecules and evaluate their applicability from different perspectives. Essentially, the idea is to propose elegant ways for approximating the inside and outside probabilities for a given input sequence using the underlying grammar – in oder to significantly reduce the worst-case time complexity of the preprocessing step and thus the overall time complexity for statistical sampling.

Notably, in connection with approximated probabilities, a novel sampling strategy might be more favorable than the well-established strategy considered in the preceding chapters (and originated in [DL03]). Depending on the implementation of the concept of approximation in order to achieve a significant time reduction, the design of such an alternative strategy is an additional challenge to be pursued in this chapter, along with a detailed description of the differences implied by the strategies as regards the corresponding preprocessing steps and overall prediction algorithms.

Obviously, this chapter must include comprehensive empirical evaluations based on comparisons with prediction results derived by the exact sampling algorithms studied in Chapters 6 and 7 in order to quantify the decline in predictive quality that results from approximation. Furthermore, similar comparisons to the accuracies obtained with several leading RNA folding tools are required for validating the applicability of the developed methods.

Chapter 5

Random Generation of RNA Secondary Structures According to Native Distributions

Random biological sequences are a topic of great interest in genome analysis since, according to a powerful paradigm, they represent the *background noise* from which the actual biological information must differentiate. Accordingly, the generation of random sequences has been investigated for a long time. Similarly, random objects of a more complicated structure like RNA molecules or proteins are of interest.

In this chapter, we present a new general framework for deriving algorithms for the non-uniform random generation of combinatorial objects according to the encoding and probability distribution implied by a SCFG. Briefly, the framework extends on the well-known recursive method for (uniform) random generation and uses the popular framework of admissible specifications of combinatorial classes, introducing weighted combinatorial classes to allow for the non-uniform generation by means of unranking. This framework is used to derive an algorithm for the generation of RNA secondary structures of a given fixed size. We address the random generation of these structures according to a realistic distribution obtained from real-life data by using a very detailed CFG (that models the class of RNA secondary structures by distinguishing between all known motifs in RNA structure).

Note that well-known sampling approaches used in several structure prediction tools (such as `Sfold`) that randomly generate secondary structures conform with a given input sequence could easily be adapted to (sequence-independently) sample random secondary structures of a given fixed size. Compared to these methods, however, ours has two major advantages: Firstly, after a preprocessing step in time $\mathcal{O}(n^2)$ for the computation of all weighted class sizes needed, a set of m random secondary structures of a given structure size n can be computed in worst-case time complexity $\mathcal{O}(m \cdot n \cdot \log(n))$ with our approach, while other algorithms typically have a runtime in $\mathcal{O}(m \cdot n^2)$. Secondly, our approach works with integer arithmetic only which is faster and saves us from all the discomforting details of using floating point arithmetic with logarithmized probabilities.

A number of experimental results shows that our random generation method produces realistic output, at least with respect to the appearance of the different structural motifs. The algorithm is available as a web service at `http://wwwagak.cs.uni-kl.de/NonUniRandGen` and can be used for generating random secondary structures of any specified RNA type. Note that a link to download an implementation of our method (in Wolfram Mathematica) can be found there, too.

5.1. Background and Motivation

The topic of random generation algorithms (also called *samplers*) has been widely studied by computer scientists. As stated in [FFP07], it has been examined under different perspectives, including combinatorics, algorithmics (design and/or engineering), as well as probability theory, where two of the main motivations for random sampling are the testing of combinatorial properties of structures (for example, conjectured structural properties, quantitative aspects), as well as the testing of properties of the corresponding algorithms (with respect to correctness and/or efficiency).

As considers software engineering, the so-called *random testing* approach is commonly used to test implementations of particular algorithms, as it is usually not feasible to consider all possible inputs and unknown which of these inputs are among the most interesting ones. In fact, this approach requires for the generation of random instances of program inputs that obey various sorts of syntactic and semantic constraints (where the random instances usually ought to be of a preliminarily fixed input size in order to be comparable to each other).

In the bioinformatics area, algorithms for generating random biological sequences have been investigated for a long time (see, for example [Fit83, AE85]). As stated in [DPT03], random sequences are a topic of great interest in genome analysis, since according to a powerful paradigm, they represent the *background noise* from which the actual biological information must differentiate. Thus, random generation of combinatorial objects can be used in this context for simulation studies in order to isolate signal (unexpected events) from noise (statistically unavoidable regularities). In fact, according to [DPT03], random biological sequences are for instance widely used for the detection of over-represented and under-represented motifs, as well as for determining whether scores of pairwise alignments are relevant or not: although there exist analytic approaches for these kinds of problems, for the most complex cases, it is often still necessary to be able to alternatively use a corresponding experimental approach (based on randomly generated sequences obtained from a computer program). For this purpose, random sequences must obviously obey to a certain model that takes into account some relevant properties of actual real-life sequences, where such models are usually based on statistical parameters only. However, it is known that these classical models can be enriched by adding structural parameters (see [DPT03]).

Over the past years, several methods have been proposed for the random generation of more complex structures, where special attention has been paid to RNA secondary structures. Most of the existing random generation algorithms for RNA secondary structures are used for predicting the structure of a given RNA sequence (see, for example [DL03, Pon08]), while others can be employed for instance for evaluating structure comparison softwares [AdCC+08]. Furthermore, as discussed in Section 3.3.7, several authors made use of SCFGs and employed machine-learning techniques to train parameter values from a set of known secondary structures. Such grammars have widely been used in a predictive mode (see, for example [KH99]) but there are also successful examples of applications where the random sampling of derivation trees has been the core of the method (see, for example [PMF+04, PFMH04, WM10] but also [GvH06]).

In this chapter, we follow that line of ideas and rely on the SCFG based approach of the technical report [WN10] to develop a new algorithm for the (non-uniform) random generation of RNA secondary structures (without pseudoknots) according to a distribution induced by a set of sample RNA data. Note that the algorithm actually generates secondary structures for a preliminary fixed size, *not* for a given RNA sequence of this size, which means we take the combinatorial point of view and completely abstract from sequence. According to this fact, it may not directly be used for the prediction of RNA secondary structures from a given input sequence. Nevertheless, positive evaluation results might validate the consideration of the underlying stochastic RNA secondary structure model as basis for a corresponding statistical

sampling algorithm that randomly generates possible foldings for a given RNA sequence in proportion to the induced native distribution derived from the utilized set of RNA data, which could then immediately be applied for single sequence RNA structure prediction.

5.2. Overview

According to [WN10], our sampling method involves a weighted *unranking* algorithm for obtaining the final structures. Briefly, considering an arbitrary structure class of size (cardinality) C, a corresponding unranking method uses a well-defined ordering of all class elements (according to a particular numbering scheme, the so-called *ranking* method) and for a given input number $r \in \{1, \ldots, C\}$ outputs the structure with rank r in the considered ordering. That way, the random sampling based on a stochastic grammar – building heavily on the use of small floating point numbers – is translated into an unranking algorithm using integer values only. Notably, a complete structure of size n is generated by recursively unranking the distinct structural components from the corresponding subclasses (of substructures with sizes less than n). In our case, the weighted unranking algorithm requires a precomputation step with worst-case time $\mathcal{O}(n^2)$ for computing all weighted class sizes up to input size n. The worst-case complexity for generating a secondary structure of size n at random is then given by $\mathcal{O}(n \cdot \log(n))$ since we are ranking structures according to the *boustrophedon order* (see, for example [Pon08]).

In the end of this chapter, we analyze the quality of randomly generated structures by considering some experimental results. First, we will consider statistical indicators of many important parameters related to particular structural motifs and compare the ones observed in the used sample set of real world RNA data to those observed in a corresponding set of random structures. Their comparison measures indicate that our method actually generates realistic RNA structures. Obviously, an algorithm which, for a given structure size n, produces random RNA secondary structures that are – related to expected shapes of such structures – in most cases realistic is a major improvement over existing approaches which, for example, are only capable of generating secondary structures uniformly for size n. Furthermore, we will consider the two different free energy models defined in [NS11a] for RNA secondary structures (with unknown RNA sequence) to get further evidence of the good quality of our random generation method (with respect to free energies and thus rather likely also with respect to appearance of the different structural motifs of RNA).

5.3. Prior Results and Basic Definitions

Before we start developing the algorithm, we first want to present some prior results and introduce the formal framework concerning (non-uniform) random generation, where we want to start with the general (uniform) case.

5.3.1. Uniform Random Generation

In the past, the problem of *uniform* random generation of combinatorial structures, that is the problem of randomly generating objects (of a preliminary fixed input size) of a specified class that have the same or similar properties, has been extensively studied. Special attention has been paid on the wide class of *decomposable structures* which are basically defined as combinatorial structures that can be constructed recursively in an unambiguous way.

In principle, two general (systematic) approaches have been developed for the uniform generation of these structures: First, the *recursive method* originated in [NW78] (to generate

various data structures) and later systematized and extended in [FZC94] (to decomposable data structures), where general combinatorial decompositions are used to generate objects at random based on counting possibilities. Second and more recently, the so-called *Boltzmann method* [DFLS04, FFP07], where random objects (under the corresponding Boltzmann model) have a fluctuating size, but objects with the same size invariably occur with the same probability.

Note that according to [DFLS04], Boltzmann samplers may be employed for approximate-size (objects with a randomly varying size are drawn) as well as fixed-size (objects of a strictly fixed size are drawn) random generation and are an alternative to standard combinatorial generators based on the recursive method. However, fixed-size generation is considered the standard paradigm for the random generation of combinatorial structures.

5.3.2. (Admissible) Constructions and Specifications

According to [FZC94], a decomposable structure is a structure that admits an equivalent *combinatorial specification*:

Definition 5.3.1 ([FZC94]) *Let $\mathcal{A} = (\mathcal{A}_1, \ldots \mathcal{A}_r)$ be an r-tuple of classes of combinatorial structures. A* specification *for $\mathcal{A}$ is a collection or r equations with the i^{th} equation being of the form $\mathcal{A}_i = \phi_i(\mathcal{A}_1, \ldots, \mathcal{A}_r)$, where ϕ_i denotes a term built of the $\mathcal{A}_j$ using the* constructions *of disjoint union, cartesian product, sequence, set and cycle, as well as the initial (neutral and atomic) classes.*

The needed formalities that will also be used in the sequel are given as follows:

Definition 5.3.2 ([FS09]) *If $\mathcal{A}$ is a combinatorial class, then $\mathcal{A}^n$ denotes the class of objects in $\mathcal{A}$ that have size (defined as number of atoms) n. Furthermore:*

- *Objects of size 0 are called* neutral objects *or* tags *and a class consisting of a single neutral object ϵ is called a* neutral class, *which will be denoted by $\mathcal{E}$ ($\mathcal{E}_1, \mathcal{E}_2, \ldots$ to distinguish multiple neutral classes containing the objects $\epsilon_1, \epsilon_2, \ldots$, respectively).*
- *Objects of size 1 are called* atomic objects *or* atoms *and a class consisting of a single atomic object is called an* atomic class, *which will be denoted by $\mathcal{Z}$ ($\mathcal{Z}_a, \mathcal{Z}_b, \ldots$ to distinguish the classes containing the atoms $a, b, \ldots$, respectively).*
- *If $\mathcal{A}_1, \ldots, \mathcal{A}_k$ are combinatorial classes and $\epsilon_1, \ldots, \epsilon_k$ are neutral objects, the* combinatorial sum *or* disjoint union *is defined as*

$$\mathcal{A}_1 + \ldots + \mathcal{A}_k := (\mathcal{E}_1 \times \mathcal{A}_1) \cup \ldots \cup (\mathcal{E}_k \times \mathcal{A}_k),$$

  where $\cup$ denotes set theoretic union.

- *If $\mathcal{A}$ and $\mathcal{B}$ are combinatorial classes, the* cartesian product *is defined as*

$$\mathcal{A} \times \mathcal{B} := \{(\alpha, \beta) \mid \alpha \in \mathcal{A} \text{ and } \beta \in \mathcal{B}\},$$

  where $\text{size}(\alpha, \beta) = \text{size}(\alpha) + \text{size}(\beta)$.

Note that the constructions of disjoint union, cartesian product, sequence, set and cycle are all admissible:

Definition 5.3.3 ([FS09]) *Let ϕ be an m-ary construction that associates to a any collection of classes $\mathcal{B}_1, \ldots, \mathcal{B}_m$ a new class $\mathcal{A} := \phi[\mathcal{B}_1, \ldots, \mathcal{B}_m]$. The construction ϕ is* admissible *iff the counting sequence (a_n) of $\mathcal{A}$ only depends on the counting sequences $(b_{1,n}), \ldots, (b_{m,n})$ of $\mathcal{B}_1, \ldots, \mathcal{B}_m$, where the counting sequence of a combinatorial class $\mathcal{A}$ is the sequence of integers $(a_n)_{n \geqslant 0}$ for $a_n = \texttt{card}(\mathcal{A}^n)$.*

The framework of (admissible) specifications obviously resembles that of CFGs known from formal language theory (see Section 3.3.4). In order to translate a CFG into the framework of admissible constructions, it is sufficient to make each terminal symbol an atom and to assume each nonterminal A to represent a class $\mathcal{A}$ (the set of all words which can be derived from nonterminal A). However, for representing CFGs, only the admissible constructions disjoint union, cartesian product and sequence are needed: Words are constructed as cartesian products of atoms, sentential forms as cartesian products of atoms and the classes assigned to the corresponding nonterminal symbols. For instance, a production rule $A \to aB$ translates into the symbolic equation $\mathcal{A} = \mathfrak{a} \times \mathcal{B}$. Different production rules with the same left-hand side give rise to the union of the corresponding cartesian products; alternatives like $A \Rightarrow \epsilon$ and $A \Rightarrow AB$ implicate sequence generation.

Nevertheless, it should be noted that [FZC94] also shows how to reduce specifications to *standard form*, where the corresponding standard specifications constitute the basis of the recursive method for uniform random generation and extends the usual CNF for CFGs. Briefly, in standard specifications, all sums and products are binary and the constructions of sequences, sets and cycles are actually replaced with other constructions (for details, see [FZC94]).

The prime advantage of standard specifications is that they translate directly into procedures for computing the sizes of all combinatorial subclasses of the considered class $\mathcal{C}$ of combinatorial objects. This means they can be used to count the number of structures of a given size that are generated from a given nonterminal symbol. Moreover, standard specifications immediately translate into procedures for generating one such structure uniformly at random. The corresponding procedures (for class size calculations and structure generations) are actually required for (uniform) random generation of words of a given CFG by means of unranking.

Note that in this context of unranking particular elements from a considered structure class, the corresponding algorithms make heavy use of their decomposability, as the distinct structural components are unranked from the corresponding subclasses. In fact, the class sizes can be derived according to the following recursion:

$$\texttt{size}(\mathcal{C}, n) := \begin{cases} 1 & \mathcal{C} \text{ is neutral and } n = 0, \\ 0 & \mathcal{C} \text{ is neutral and } n \neq 0, \\ 1 & \mathcal{C} \text{ is atomic and } n = 1, \\ 0 & \mathcal{C} \text{ is atomic and } n \neq 1, \\ \sum_{i=1}^{k} \texttt{size}(\mathcal{A}_i, n) & \mathcal{C} = \mathcal{A}_1 + \ldots + \mathcal{A}_k, \\ \sum_{j=0}^{n} \texttt{size}(\mathcal{A}, j) \cdot \texttt{size}(\mathcal{B}, n-j) & \mathcal{C} = \mathcal{A} \times \mathcal{B}. \end{cases}$$

Note that when computing the sums for cartesian products, we can either consider the values for j in the *sequential* (also called *lexicographic*) order $(1, 2, 3, \ldots, n)$ or in the so-called *boustrophedon* order $(1, n, 2, n-1, \ldots, \lceil \frac{n}{2} \rceil)$. In either case, given a fix number of considered combinatorial (sub)classes (or corresponding nonterminal symbols), the precomputation of all class size tables up to size n requires $\mathcal{O}(n^2)$ operations on coefficients. One random generation step then needs $\mathcal{O}(n^2)$ arithmetic operations when using the *sequential method* and $\mathcal{O}(n \cdot \log(n))$ operations when using the *boustrophedon method* (for details, we refer to [FZC94]).

Obviously, using uniform unranking procedures to construct the i^{th} structure of size n for a randomly drawn number i, any structure of size n is equiprobably generated. Consequently, in order to make sure that, for given size n and a sample set of random numbers i, the corresponding structures are in accordance with an appropriate probability distribution (as for instance observed from real-life RNA data), it is mandatory to use a corresponding non-uniform unranking method or an alternative non-uniform random generation approach.

5.3.3. Non-Uniform Random Generation

Since decomposable objects (like for instance RNA secondary structures) can conveniently be modeled by appropriate SCFGs (see, for example [Neb02b, Neb04a]), it seems reasonable to further utilize this concept in connection with random sampling. For this reason, the problem of *non-uniform* random generation of combinatorial structures has been recently addressed in [WN10], where it is described how to get algorithms for the random generation of objects of a previously fixed size according to an arbitrary (non-uniform) distribution implied by a given SCFG. In principle, the construction scheme introduced in [WN10] extends on the recursive method for the (uniform) random generation [FZC94] and adapted it to the problem of unranking of [Mol05].

Essentially, in [WN10], a new admissible construction called *weighting* has been introduced in order to make non-uniform random generation possible. By weighting, we understand the generation of distinguishable copies of objects. Formally:

Definition 5.3.4 *If $\mathcal{A}$ is a combinatorial class and λ is an integer, the* weighting *of $\mathcal{A}$ by λ is defined as*

$$\lambda\mathcal{A} := \underbrace{\mathcal{A} + \ldots + \mathcal{A}}_{\lambda \text{ times}}.$$

We will call two objects from a combinatorial class copies of the same object *iff they only differ in the tags added by weighting operations.*

Example 5.3.1 *If we weight the class $\mathcal{A} = \{a\}$ by two, we assume the result to be the set $\{a, a\}$. Accordingly, weighting $\mathcal{B} = \{b\}$ by three generates $\{b, b, b\}$. Thus, $2\mathcal{A} + 3\mathcal{B} = \{a, a, b, b, b\}$ and within this class, a has relative frequency $2/5$, while b has relative frequency $3/5$. Hence, this way it becomes possible to regard non-uniformly distributed classes.*

As weighting a class can be replaced by a disjoint union, $\texttt{size}(\lambda\mathcal{A}, n) = \lambda \cdot \texttt{size}(\mathcal{A}, n)$ and the complexity results from [Mol05] also hold for weighted classes. Hence, the corresponding class size computations up to n need $\mathcal{O}(n^2)$ time.

5.3.4. Random Generation With SCFGs

SCFGs can easily be used for the random generation of combinatorial objects according to the probability distribution induced by a sample set of data, where the only problem is that they do not allow the user to fix the length of generated structures. In particular, given an SCFG $\mathcal{G}$ and the corresponding language (combinatorial class) $\mathcal{L}(\mathcal{G})$, a random word $w \in \mathcal{L}(\mathcal{G})$ can be generated in the following way:

- Start with the sentential form S (where S denotes the axiom of the grammar $\mathcal{G}$).
- While there are nonterminal symbols (in the currently considered sentential form), do the following:
  1) Let A denote the leftmost nonterminal symbol.
  2) Draw a random number r from the interval $(0, 1]$.
  3) Substitute symbol A by the right-hand side α of the production $A \to \alpha$ determined by the random number r. This means consider all $m \geqslant 1$ rules

$$p_1 : A \to \alpha_1, \ldots, p_m : A \to \alpha_m$$

having left-hand side A, where according to the definition of SCFGs,

$$\sum_{i=1}^{m} p_i = 1$$

must hold. Then, find $k \geqslant 1$ with

$$\sum_{i=1}^{k-1} p_i < r \leqslant \sum_{i=1}^{k} p_i,$$

that is determine $k \geqslant 1$ with

$$r \in \left(\sum_{i=1}^{k-1} p_i, \sum_{i=1}^{k} p_i \right].$$

The production corresponding to the randomly drawn number $r \in (0, 1]$ is then given by $A \to \alpha_k$ and hence, in the currently considered sentential form, the nonterminal symbol A is substituted by α_k.

- If there are no more nonterminal symbols, then the currently considered sentential form is equal to a word $w \in \mathcal{L}(\mathcal{G})$. Thus, w has been randomly generated.

Note that the choice of the production made in 3) according to the previously drawn random number is appropriate, since it is conform to the probability distribution on the grammar rules.

Example 5.3.2 *Consider the language generated by the SCFG with productions* $3/4 : S \to \epsilon$ *and* $1/4 : S \to (S)$. *Thus, we start with the sentential form* S, *then consider the leftmost nonterminal symbol, which is given by* S, *and draw a random number* $r \in (0, 1]$. *If* $0 < r \leqslant 3/4$, *the production determined by* r *is* $S \to \epsilon$ *and thus, we get the empty word and are finished. Otherwise,* $3/4 < r \leqslant 3/4 + 1/4 = 1$, *which means we have to consider* $A \to (S)$ *for the substitution in step 3) and thus obtain the sentential form* (S). *Afterwards, we must repeat the process, as there is still one nonterminal symbol left.*

Unfortunately, there is one major problem that comes with this approach for the (non-uniform) random generation of combinatorial objects: The underlying (consistent) SCFG $\mathcal{G}$ implies a probability distribution on the whole language $\mathcal{L}(\mathcal{G})$, such that we generate a word of arbitrary size. In order to fix the size, we can proceed along the following lines:

1) We translate the grammar $\mathcal{G}$ into a new framework which allows to consider fixed sizes for the random generation, such that

2) the distribution implied on $\mathcal{L}(\mathcal{G})$ conditioned on any fixed size n is kept within the new framework.

A well-known approach which allows for 1) is connected to the concept of admissible constructions used to describe a decomposable combinatorial class (see above).

5.3.5. Why Using Unranking?

Some might think that with an appropriate SCFG (modeling a given class of objects) at hand, it is not really necessary to use an unranking method that implies cumbersome formalities such as admissible constructions and decomposable classes if we want to generate random objects of a fixed size n. As a matter of principle, they are right – we could also use a *conditional sampling* method: If we need to generate a word of size n from nonterminal symbol A, where there are

$m \geqslant 1$ rules $f_i = A \to \alpha_i$, $1 \leqslant i \leqslant m$, having left-hand side A, then we just need to choose the next production f_i according to

$$\frac{\Pr(A \to \alpha_i \Rightarrow^* x \mid \text{size}(x) = n)}{\Pr(A \Rightarrow^* x \mid \text{size}(x) = n)},$$

which is the probability that we used production rule f_i under the condition that a word of size n is generated. Similarly, if the production rule is of the type $A \to BC$ (assuming the grammar is in CNF, which does not pose a problem, as an unambiguous SCFG can be efficiently transformed into CNF [HF71]), we can choose a way to split size n into sizes j and $n-j$ for the lengths generated from nonterminal symbols B and C.

This requires precomputing n *length-dependent* probabilities (that is, all probabilities for generating a word of any length up to n) for each nonterminal symbol, which might seem to be similar (with respect to complexity) to precomputing all class sizes up to n for all considered combinatorial (sub)classes as needs to be done for unranking. However, there are some striking differences between the two approaches:

- While conditional sampling makes heavy use of rather small floating point values – with all the well-known problems and discomforting details like underflows or using logarithms associated with it – an unranking approach builds on integer values only, which we assume a major advantage.

- Furthermore, length-dependent probabilities (actually yielding a LSCFG, see Section 3.3.8) require a very rich training set. In fact, if the RNA data set used for determining the distribution induced by the grammar is not rich enough, then the corresponding stochastic RNA model is underestimated and its quality decreases. This is especially a problem when considering comprehensive CFGs that distinguish between many different structural motifs in order to get a realistic picture of the molecules' behavior; such a grammar might in some cases however be preferred over simple lightweight grammars as basis for a non-uniform random generation method. Nevertheless, this problem does not surface when sticking to conventional probabilities and the corresponding traditional SCFG model. Actually, since we consider a huge CFG where all possible structural motifs are created by distinct productions, we generally obtain realistic probability distributions and RNA models (see [NS11a]).

Finally, note that of course we could make use of random sampling strategies originally designed to sample structures connected to a given sequence in order to generate a random secondary structure. However, such algorithms typically use a linear time to sample a single base pair (see, for example [DL03]), such that the time to sample a complete structure is quadratic in its length. This causes no problems for the original application of such algorithms, since the sequence-dependent preprocessing (which is of course part of their overall procedure) is at least quadratic in time and thus the dominating part. Here, our approach is of advantage, as it can actually be implemented to replace a factor n by $\log(n)$ for the generation of a complete structure of size n. Additionally, since our preprocessing only depends on the size of the structure to be generated, it is performed once and the results are stored to disk for later reuse. Last but not least, we are not sure if the different existing approaches just mentioned could easily be made as fast as ours by simple changes only.

Bottom line is that hooking up to unranking of combinatorial classes offers three significant benefit compared to conditional sampling, namely a fast sampling strategy, the usage of integers instead of floating point values and a greater independence of the richness of the training data (compared to length-dependent models). For this reason, we assume our unranking algorithm a valuable contribution, even though it requires a more cumbersome framework.

5.3.6. Unranking of Combinatorial Objects

The problem of unranking can easily be solved along the composition of the objects at hand, that is the operations used for its construction, once we know the number of possible choices for each substructure. For example, assume we want to unrank objects from a class $\mathcal{C} = \mathcal{A} + \mathcal{B}$. We will assume all elements of $\mathcal{A}$ to be of smaller order than those of $\mathcal{B}$ (this way we use the construction of the class to imply an ordering). Finding the i^{th} element of $\mathcal{C}$, that is unranking class $\mathcal{C}$, now becomes possible by deciding whether $i < \texttt{card}(\mathcal{A})$. In this case, we recursively call the unranking procedure for $\mathcal{A}$. Otherwise (that is, if $i \geqslant \texttt{card}(\mathcal{A})$), we consider $\mathcal{B}$, searching for its $(i - \texttt{card}(\mathcal{A}))^{th}$ element.

Formally, we first need to specify an order on all objects of the considered combinatorial class that have the same size. This can be done in a recursive way according to the admissible specification of the class:

Definition 5.3.5 ([Mol05]) *Neutral and atomic classes contain only one element, such that there is only one possible ordering. Furthermore, let $<_{\mathcal{C}^n}$ denote the ordering within the combinatorial class $\mathcal{C}^n$, then*

- *If $\mathcal{C} = \mathcal{A}_1 + \ldots + \mathcal{A}_k$ and $\gamma, \gamma' \in \mathcal{C}^n$, then $\gamma <_{\mathcal{C}^n} \gamma'$ iff*

$$[\gamma \in (\mathcal{A}_i)^n \text{ and } \gamma' \in (\mathcal{A}_j)^n \text{ and } i < j] \text{ or } [\gamma, \gamma' \in (\mathcal{A}_i)^n \text{ and } \gamma <_{(\mathcal{A}_i)^n} \gamma'].$$

- *If $\mathcal{C} = \mathcal{A} \times \mathcal{B}$ and $\gamma = (\alpha, \beta), \gamma' = (\alpha', \beta') \in \mathcal{C}^n$, then $\gamma <_{\mathcal{C}^n} \gamma'$ iff*

$$[\text{size}(\alpha) < \text{size}(\alpha')] \text{ or } [j = \text{size}(\alpha) = \text{size}(\alpha') \text{ and } \alpha <_{(\mathcal{A})^j} \alpha'] \text{ or } [\alpha = \alpha' \text{ and } \beta <_{(\mathcal{B})^{n-j}} \beta']$$

when considering the lexicographic order $(1, 2, 3, \ldots, n)$, which is induced by the specification

$$\mathcal{C}^n = \mathcal{A}^0 \times \mathcal{B}^n + \mathcal{A}^1 \times \mathcal{B}^{n-1} + \mathcal{A}^2 \times \mathcal{B}^{n-2} + \ldots + \mathcal{A}^n \times \mathcal{B}^0.$$

- *If $\mathcal{C} = \mathcal{A} \times \mathcal{B}$ and $\gamma = (\alpha, \beta), \gamma' = (\alpha', \beta') \in \mathcal{C}^n$, then $\gamma <_{\mathcal{C}^n} \gamma'$ iff*

$$\begin{aligned}
&[\min(\text{size}(\alpha), \text{size}(\beta)) < \min(\text{size}(\alpha'), \text{size}(\beta'))] \text{ or} \\
&[\min(\text{size}(\alpha), \text{size}(\beta)) = \min(\text{size}(\alpha'), \text{size}(\beta')) \text{ and } \text{size}(\alpha) < \text{size}(\alpha')] \text{ or} \\
&[j = \text{size}(\alpha) = \text{size}(\alpha') \text{ and } \alpha <_{(\mathcal{A})^j} \alpha'] \text{ or } [\alpha = \alpha' \text{ and } \beta <_{(\mathcal{B})^{n-j}} \beta']
\end{aligned}$$

when considering the boustrophedon order $(1, n, 2, n-1, \ldots, \lceil \frac{n}{2} \rceil)$, induced by the specification

$$\mathcal{C}^n = \mathcal{A}^0 \times \mathcal{B}^n + \mathcal{A}^n \times \mathcal{B}^0 + \mathcal{A}^1 \times \mathcal{B}^{n-1} + \mathcal{A}^{n-1} \times \mathcal{B}^1 + \ldots$$

Considering $<_{\mathcal{C}^n}$, the actual unranking algorithms are quite straightforward. Therefore, they will not be presented here and we refer to [MM01, WN10] for details.

Recall that in [WN10], the basic approach towards non-uniform random generation is weighting of combinatorial classes, as this makes it possible that the classes are non-uniformly distributed. If those combinatorial classes are to correspond to a considered SCFG, we have to face the problem that maximum likelihood training introduces rational weights for the production rules while weighting as an admissible construction needs integer arguments.

When translating rational probabilities into integral weights, we have to assure that the relative weight of each (unambiguously) generated word of some size remains unchanged. This can be reached by scaling all productions by the same factor (common denominator of all probabilities), while ensuring that derivations are of equal length for words of the same size (ensured by using grammars in CNF). However, a much more elegant way is to scale each production according

to its contribution to the length of the word generated, that is, productions lengthening the word by k will be scaled by c^k. Since we consider CFGs, the lengthening of a production of the form $A \to \alpha$ is given by $|\alpha| - 1$, but this rule leads to productions with a conclusion of length 1 not being reweighted. Hence, we have to assure that all those productions already have integral weights. Furthermore, ϵ-productions need a special treatment. We don't want to discuss full details here and conclude by noticing that the *reweighting normal form (RNF)* keeps track of all possible issues:

Definition 5.3.6 ([WN10]) *If $\mathcal{G} = (I, T, R, S, W)$ is a WCFG[1], $\mathcal{G}$ is said to be in* reweighting normal form (RNF) *iff*

1. *$\mathcal{G}$ is loop-free and ϵ-free.*
2. *For all $A \to \alpha \in R$ with $A = S$, we have $|\alpha| \leqslant 1$.*
3. *For all $A \to \alpha \in R$ with $A \neq S$, we have $|\alpha| > 1$ or $W(A \to \alpha) \in \mathbb{N}$.*
4. *For all $A \in I$ there exists $\alpha \in (I \cup T)^*$ such that $A \to \alpha \in R$.*

Note that the last condition (that any intermediate symbol occurs as premise of at least one production) is not required for reweighting, but necessary for the translation of a grammar into an admissible specification.

Definition 5.3.7 ([WN10]) *If $\mathcal{G}$ and $\mathcal{G}'$ are WCFGs, then $\mathcal{G}$ and $\mathcal{G}'$ are said to be* word-equivalent *iff $\mathcal{L}(\mathcal{G}) = \mathcal{L}(\mathcal{G}')$ and for each word $w \in \mathcal{L}(\mathcal{G})$, we have $W(w) = W'(w)$.*

In [WN10], it is shown how to transform an arbitrary SCFG to a word-equivalent, loop-free and ϵ-free grammar, that grammar to one in RNF and the latter to the corresponding admissible specification. Formally:

Theorem 5.3.1 ([HF71]) *If $\mathcal{G}$ is a SCFG, there exists a SCFG $\mathcal{G}'$ in CNF that is word-equivalent to $\mathcal{G}$, and $\mathcal{G}'$ can be effectively constructed from $\mathcal{G}$.*

The construction given in [HF71] assumes that $\mathcal{G}$ is ϵ-free. It can however be extended to non-ϵ-free grammars by adding an additional step after the intermediate grammar $\mathcal{G}$ has been created (see, for example [WN10]). Furthermore, it should be noted that an unambiguous grammar is inevitably loop-free.

Theorem 5.3.2 ([WN10]) *If $\mathcal{G}$ is a loop-free, ϵ-free WCFG, there exists a WCFG $\mathcal{G}'$ in RNF that is word-equivalent to $\mathcal{G}$ and $\mathcal{G}'$ can be effectively constructed from $\mathcal{G}$.*

Altogether, starting with an arbitrary unambiguous SCFG $\mathcal{G}_0$ that models the class of objects to be randomly generated, we have to proceed along the following lines:

- Transform $\mathcal{G}_0$ to a corresponding ϵ-free and loop-free SCFG $\mathcal{G}_1$.
- Transform $\mathcal{G}_1$ into $\mathcal{G}_2$ in RNF (where all production weights are rational).
- Reweight the production rules of $\mathcal{G}_2$ (such that all production weights are integral), yielding reweighted WCFG $\mathcal{G}_3$.
- Transform $\mathcal{G}_3$ (with integral weights) into the corresponding admissible specification.
- This specification (with weighted classes) can be translated directly

[1] This means $\mathcal{G}$ must only be a *weighted* CFG, that is not necessarily a proper SCFG, see Definition 3.3.3.

- into a recursion for the function `size` of all involved combinatorial (sub)classes (where class sizes are weighted) and
- into generating algorithms for the specified (weighted) classes,

yielding the desired weighted unranking algorithm for generating random elements of $\mathcal{L}(\mathcal{G}_0)$.

To complete this section, we decided to present the following (rather small) example that shows how to proceed from an arbitrary SCFG to RNF and then to the corresponding weighted combinatorial classes which allow for non-uniform generation by means of unranking:

Example 5.3.3 *Let us consider the SCFG $\mathcal{G}_{e,u}$, which generates a secondary structure related class and contains exactly the following rules:*

$$\begin{aligned} &w_1 : S \to B, \\ &w_2 : B \to (B), \quad w_3 : B \to \circ C, \\ &w_4 : C \to \epsilon, \quad w_5 : C \to \circ C. \end{aligned}$$

To apply the approach presented in [WN10] to transform a given SCFG to RNF, the grammar needs to be ϵ-free and loop-free. Thus, we first have to transform grammar $\mathcal{G}_{e,u}$ into the following one:

$$\begin{aligned} &\widehat{w}_1 : S \to B, \\ &\widehat{w}_2 : B \to (B), \quad \widehat{w}_3 : B \to C, \\ &\widehat{w}_4 : C \to \circ, \quad \widehat{w}_5 : C \to \circ C. \end{aligned}$$

The transformation of $\mathcal{G}_{e,u}$ into RNF now works as follows: First, we have to gather all possible chains $A \to A_1 \to A_2 \to \ldots \to \alpha$, where $A \neq S$ and $|\alpha| = 1$. These chains are $B \to C$, $B \to C \to \circ$ and $C \to \circ$. The rules $B \to C$ and $C \to \circ$ are then removed. Second, we have to replace each of these chains by a specific new rule. In fact, we have to add $B^{C,\epsilon} \to C$, $B^{\circ,C} \to \circ$ and $C^{\circ,\epsilon} \to \circ$ to the new set of productions. Consequently, our new rule set is now given by

$$\begin{aligned} &\widehat{w}_1 : S \to B, \\ &\widehat{w}_2 : B \to (B), \\ &\widehat{w}_5 : C \to \circ C, \\ &1 : B^{C,\epsilon} \to C, \quad 1 : B^{\circ,C} \to \circ, \quad 1 : C^{\circ,\epsilon} \to \circ. \end{aligned}$$

Third, for each occurrence of a nonterminal symbol A in the conclusion of a production and each previously added new rule $A^{\alpha,A_1A_2\cdots} \to \alpha$ corresponding to a chain $A \to A_1 \to A_2 \to \ldots \to \alpha$, add a specific new rule. This way, we obtain the following production set:

$$\begin{aligned} &\widehat{w}_1 : S \to B, \quad \widehat{w}_1 \cdot \widehat{w}_3 : S \to B^{C,\epsilon}, \quad \widehat{w}_1 \cdot \widehat{w}_3 \cdot \widehat{w}_4 : S \to B^{\circ,C}, \\ &\widehat{w}_2 : B \to (B), \quad \widehat{w}_2 \cdot \widehat{w}_3 : B \to (B^{C,\epsilon}), \quad \widehat{w}_2 \cdot \widehat{w}_3 \cdot \widehat{w}_4 : B \to (B^{\circ,C}), \\ &\widehat{w}_5 : C \to \circ C, \quad \widehat{w}_5 \cdot \widehat{w}_4 : C \to \circ C^{\circ,\epsilon}, \\ &1 : B^{C,\epsilon} \to C, \quad 1 : B^{\circ,C} \to \circ, \quad 1 : C^{\circ,\epsilon} \to \circ. \end{aligned}$$

Fourth, each intermediate symbol that no longer occurs as premise in any of the productions has to be removed and fifth, each production of the form $S \to \alpha$, where S is the axiom and $|\alpha| > 1$ has to be changed in a specific way. However, since in our case, there is obviously nothing left to do, the transformation of $\mathcal{G}_{e,u}$ into RNF is finished.

For $\mathcal{G}_{e,u}$ (in RNF), where all production weights are rational, we can determine the common denominator s of the weights of productions with premise S, as well as the common denominator c of the weights of the remaining productions (that is, of the productions with premise B or C). Then, the reweighting of the production rules of (the RNF of) $\mathcal{G}_{e,u}$ is done by multiplying the weights of productions with source S

by s, and the weights of the other productions $A \to \alpha$, where $A \neq S$, by the factor $c^{|\alpha|-1}$. After that, we obtain the following reweighted grammar $\mathcal{G}'_{e,u}$:

$$\begin{array}{lll} w'_1 : S \to B, & w'_2 : S \to B^{C,\epsilon}, & w'_3 : S \to B^{\circ,C}, \\ w'_4 : B \to (B), & w'_5 : B \to (B^{C,\epsilon}), & w'_6 : B \to (B^{\circ,C}), \\ w'_7 : C \to \circ C, & w'_8 : C \to \circ C^{\circ,\epsilon}, & \\ 1 : B^{C,\epsilon} \to C, & 1 : B^{\circ,C} \to \circ, & 1 : C^{\circ,\epsilon} \to \circ, \end{array}$$

where each w'_i, $1 \leqslant i \leqslant 8$, is integral. The (now weighted) grammar can easily be translated into a corresponding admissible specification, which includes the weighting of all involved combinatorial (sub)classes, as described earlier. For the reweighted grammar $\mathcal{G}'_{e,u}$, this specification is given by the following equations:

$$\begin{array}{lll} \mathcal{S}_1 = \mathcal{B}, & \mathcal{S}_2 = \mathcal{B}^{C,\epsilon}, & \mathcal{S}_3 = \mathcal{B}^{\circ,C}, \\ \mathcal{B}_1 = \mathcal{Z}_(\times \mathcal{B} \times \mathcal{Z}_), & \mathcal{B}_2 = \mathcal{Z}_(\times \mathcal{B}^{C,\epsilon} \times \mathcal{Z}_), & \mathcal{B}_3 = \mathcal{Z}_(\times \mathcal{B}^{\circ,C} \times \mathcal{Z}_), \\ \mathcal{C}_1 = \mathcal{Z}_\circ \times \mathcal{C}, & \mathcal{C}_2 = \mathcal{Z}_\circ \times \mathcal{C}^{\circ,\epsilon}, & \\ \mathcal{B}^{C,\epsilon} = \mathcal{C}, & \mathcal{B}^{\circ,C} = \mathcal{Z}_\circ, & \mathcal{C}^{\circ,\epsilon} = \mathcal{Z}_\circ, \end{array}$$

$$\begin{aligned} \mathcal{S} &= w'_1 \cdot \mathcal{S}_1 + w'_2 \cdot \mathcal{S}_2 + w'_3 \cdot \mathcal{S}_3, \\ \mathcal{B} &= w'_4 \cdot \mathcal{B}_1 + w'_5 \cdot \mathcal{B}_2 + w'_6 \cdot \mathcal{B}_3, \\ \mathcal{C} &= w'_7 \cdot \mathcal{C}_1 + w'_8 \cdot \mathcal{C}_2, \end{aligned}$$

which can be simplified in the following way:

$$\begin{array}{lll} \mathcal{B}_1 = \mathcal{Z}_(\times \mathcal{B} \times \mathcal{Z}_), & \mathcal{B}_2 = \mathcal{Z}_(\times \mathcal{C} \times \mathcal{Z}_), & \mathcal{B}_3 = \mathcal{Z}_(\times \mathcal{Z}_\circ \times \mathcal{Z}_), \\ \mathcal{C}_1 = \mathcal{Z}_\circ \times \mathcal{C}, & \mathcal{C}_2 = \mathcal{Z}_\circ \times \mathcal{Z}_\circ, & \end{array}$$

$$\begin{aligned} \mathcal{S} &= w'_1 \cdot \mathcal{B} + w'_2 \cdot \mathcal{C} + w'_3 \cdot \mathcal{Z}_\circ, \\ \mathcal{B} &= w'_4 \cdot \mathcal{B}_1 + w'_5 \cdot \mathcal{B}_2 + w'_6 \cdot \mathcal{B}_3, \\ \mathcal{C} &= w'_7 \cdot \mathcal{C}_1 + w'_8 \cdot \mathcal{C}_2. \end{aligned}$$

As described earlier, this specification (with weighted classes) derived from reweighted grammar $\mathcal{G}'_{e,u}$ transforms immediately into a recursion for the function size of all needed combinatorial classes. For $\mathcal{G}'_{e,u}$, the recursion for the function size has the following form:

$$\mathsf{size}(\mathcal{I}, n) := \begin{cases} \mathsf{size}(\mathcal{B}, n-2) & \mathcal{I} = \mathcal{B}_1, \\ \mathsf{size}(\mathcal{C}, n-2) & \mathcal{I} = \mathcal{B}_2, \\ 1 & \mathcal{I} = \mathcal{B}_3 \text{ and } n = 3, \\ \mathsf{size}(\mathcal{C}, n-1) & \mathcal{I} = \mathcal{C}_1, \\ 1 & \mathcal{I} = \mathcal{C}_2 \text{ and } n = 2, \\ w'_1 \cdot \mathsf{size}(\mathcal{B}, n) + w'_2 \cdot \mathsf{size}(\mathcal{C}, n) + w'_3 \cdot 1 & \mathcal{I} = \mathcal{S} \text{ and } n = 1, \\ w'_1 \cdot \mathsf{size}(\mathcal{B}, n) + w'_2 \cdot \mathsf{size}(\mathcal{C}, n) + w'_3 \cdot 0 & \mathcal{I} = \mathcal{S} \text{ and } n \neq 1, \\ w'_4 \cdot \mathsf{size}(\mathcal{B}_1, n) + w'_5 \cdot \mathsf{size}(\mathcal{B}_2, n) + w'_6 \cdot \mathsf{size}(\mathcal{B}_3, n) & \mathcal{I} = \mathcal{B}, \\ w'_7 \cdot \mathsf{size}(\mathcal{C}_1, n) + w'_8 \cdot \mathsf{size}(\mathcal{C}_2, n) & \mathcal{I} = \mathcal{C}, \\ 0 & \text{else}. \end{cases}$$

This recursive size *function (with weighted class sizes) can now be used for the straightforward construction of a corresponding algorithm for the non-uniform generation of elements of $\mathcal{L}(\mathcal{G}_{e,u})$ by means of unranking, as proposed in [WN10].*

5.4. Generating Random RNA Secondary Structures

We will now consider the previously discussed approach to construct a weighted unranking algorithm that generates random RNA secondary structures of a given size according to a realistic probability distribution. As for this chapter, the corresponding probability distribution will be induced by a set of sample (SSU and LSU r)RNA secondary structures from the databases [WRdP+01, WdPWW02], which will be referred to as *biological database* in the sequel. However, the presented algorithm can easily be used for any other distribution, which can be defined by a database of known RNA structures of a particular RNA type[2].

5.4.1. Considered Combinatorial Class

According to the most common definition of RNA secondary structure, we decided to consider the combinatorial class of all RNA secondary structures without pseudoknots that meet the stereochemical constraint of hairpin loops consisting of at least $\min_{HL} = 3$ unpaired nucleotides, where isolated base pairs are explicitly allowed. Furthermore, any secondary structure must contain at least one base paired structure, that is completely unpaired secondary structures are prohibited. A formal definition of the corresponding CFL $\mathcal{L}$ is given in Definition 3.3.2.

Thus, for a given size n and a given number $i \in \{0, \ldots, \mathtt{card}(\mathcal{L}^n) - 1\}$, the desired weighted unranking algorithm should generate the i^{th} secondary structure $s \in \mathcal{L}^n$, where $\mathtt{card}(\mathcal{L}^n) = \mathrm{size}(\mathcal{L}, n)$ is the number of elements in the weighted class $\mathcal{L}^n$.

5.4.2. Considered SCFG Model

In order to derive our algorithm, we initially have to find a suitable SCFG that generates $\mathcal{L}$ and models the distribution of the sample data as closely as possible. To reach this goal, it is important to appropriately specify the set of production rules in order to guarantee that all substructures that have to be distinguished are generated by different rules (see Section 3.3.6.1 and especially the illustrations by Example 3.3.7 for details).

For this reason, we decided that the basis for our weighted unranking algorithm should be the following ϵ-free, loop-free and unambiguous[3] SCFG, which has been derived from the sophisticated grammar presented in [NS11a] (that actually manages to distinguish between all known structural motifs that can be found in RNA secondary structure):

Definition 5.4.1 *The unambiguous ϵ-free SCFG $\widehat{\mathcal{G}}_u$ generating exactly the language $\mathcal{L}$ is given by $\widehat{\mathcal{G}}_u = (I_{\widehat{\mathcal{G}}_u}, \Sigma_{\widehat{\mathcal{G}}_u}, R_{\widehat{\mathcal{G}}_u}, S')$, where*

$$I_{\widehat{\mathcal{G}}_u} = \{S', E, S, T, C, A, L, G, D, B, F, H, P, Q, R, V, W, O, J, K, M, X, Y, Z, N, U\},$$

$\Sigma_{\widehat{\mathcal{G}}_u} = \{(,), \circ\}$ *and* $R_{\widehat{\mathcal{G}}_u}$ *contains exactly the following rules:*

$$\left.\begin{array}{ll} \widehat{p}_1 : S' \to E, & \\ \widehat{p}_2 : E \to S, & \widehat{p}_3 : E \to SC, \\ \widehat{p}_4 : S \to A, & \widehat{p}_5 : S \to TA, \\ \widehat{p}_6 : T \to E, & \widehat{p}_7 : T \to C, \end{array}\right\} \textit{shape of exterior loop}$$

$\widehat{p}_8 : C \to \circ, \quad \widehat{p}_9 : C \to C\circ, \quad \rightsquigarrow$ *strands in exterior loop*

$\widehat{p}_{10} : A \to (L), \quad \rightsquigarrow$ *initiate helix*

[2]Note that our web service implementation accessible at `http://wwwagak.cs.uni-kl.de/NonUniRandGen` is actually able to sample random secondary structures of any specified RNA type.

[3]Note that these are exactly the preliminary required conditions for the basis grammar according to [WN10].

$\widehat{p}_{11} : L \to A$, $\widehat{p}_{12} : L \to M$, $\rightsquigarrow$ *initiate stacked pair or multiple loop*

$\widehat{p}_{13} : L \to P$, $\widehat{p}_{14} : L \to Q$, $\widehat{p}_{15} : L \to R$, $\rightsquigarrow$ *initiate interior loop*

$\widehat{p}_{16} : L \to F$, $\widehat{p}_{17} : L \to G$, $\rightsquigarrow$ *initiate hairpin loop or bulge loop*

$\widehat{p}_{18} : G \to A\circ$, $\widehat{p}_{19} : G \to AD$, $\widehat{p}_{20} : G \to \circ A$, $\widehat{p}_{21} : G \to DA$, $\rightsquigarrow$ *shape of bulge loop*

$\left.\begin{array}{l}\widehat{p}_{22} : D \to B\circ,\\ \widehat{p}_{23} : B \to \circ, \quad \widehat{p}_{24} : B \to B\circ,\end{array}\right\}$ *strands in bulge loop*

$\left.\begin{array}{l}\widehat{p}_{25} : F \to \circ\circ\circ, \quad \widehat{p}_{26} : F \to \circ\circ\circ\circ, \quad \widehat{p}_{27} : F \to \circ\circ\circ\circ H,\\ \widehat{p}_{28} : H \to \circ, \quad \widehat{p}_{29} : H \to H\circ,\end{array}\right\}$ *hairpin loop*

$\widehat{p}_{30} : P \to \circ A\circ$, $\widehat{p}_{31} : P \to \circ A\circ\circ$, $\widehat{p}_{32} : P \to \circ\circ A\circ$, $\widehat{p}_{33} : P \to \circ\circ A\circ\circ$, $\rightsquigarrow$ *small interior loops*

$\left.\begin{array}{l}\widehat{p}_{34} : Q \to \circ\circ O\circ\circ, \quad \widehat{p}_{35} : Q \to \circ\circ V\circ,\\ \widehat{p}_{36} : R \to \circ O\circ\circ, \quad \widehat{p}_{37} : R \to \circ\circ W\circ,\\ \widehat{p}_{38} : V \to JO,\\ \widehat{p}_{39} : W \to JA,\\ \widehat{p}_{40} : O \to AK,\end{array}\right\}$ *other interior loops*

$\left.\begin{array}{l}\widehat{p}_{41} : J \to \circ, \quad \widehat{p}_{42} : J \to J\circ,\\ \widehat{p}_{43} : K \to \circ, \quad \widehat{p}_{44} : K \to K\circ,\end{array}\right\}$ *strands in interior loop*

$\left.\begin{array}{l}\widehat{p}_{45} : M \to XY,\\ \widehat{p}_{46} : X \to A, \quad \widehat{p}_{47} : X \to UA,\\ \widehat{p}_{48} : Y \to Z,\\ \widehat{p}_{49} : Z \to X, \quad \widehat{p}_{50} : Z \to XN,\\ \widehat{p}_{51} : N \to Z, \quad \widehat{p}_{52} : N \to U,\end{array}\right\}$ *multiple loop*

$\widehat{p}_{53} : U \to \circ$, $\widehat{p}_{54} : U \to U\circ$. $\rightsquigarrow$ *strands in multiple loop*

Figures 5.1 and 5.2 illustrate by examples how (parts of) secondary structures are generated by this SCFG, where we used $\overset{I}{\underset{x}{\triangle}}$ to denote the full parse tree for $I \Rightarrow^* x$ (that is, for consecutive applications of an arbitrary number of production rules that generate the subword x from the intermediate symbol I) in oder to obtain a more compact tree representation. In fact, it is easy to see that the overall structure is always produced by starting with the axiom S', while any particular substructure or structural motif that belongs to the combinatorial (sub)class $\mathcal{I}$ is created from the corresponding intermediate symbol I.

Remark 5.4.1 *For our application, it is crucial that $\widehat{\mathcal{G}}_u$ – as claimed its definition – is unambiguous. To prove this, we first note that (the CFG from [NS11a] and then) $\widehat{\mathcal{G}}_u$ has been constructed by starting from a simple grammar which generates $\mathcal{L}$ and then iteratively replacing one production by several more specialized ones (like we did in Example 3.3.7) in order to distinguish more and more structural motifs, while taking care not to change the generated language. Furthermore, a standard construction to make the grammar ϵ-free has been applied. That way, we can be sure that $\widehat{\mathcal{G}}_u$ generates $\mathcal{L}$ (formally this fact easily follows by obvious bi-simulation proofs for each substitution and by the proven correctness of the used construction to ensure ϵ-freeness).*

To prove unambiguity, we translated $\widehat{\mathcal{G}}_u$ into a system of equations for its structure generating function, which is given by

$$S[z] = \sum_{w \in \mathcal{L}} d(w) z^{|w|},$$

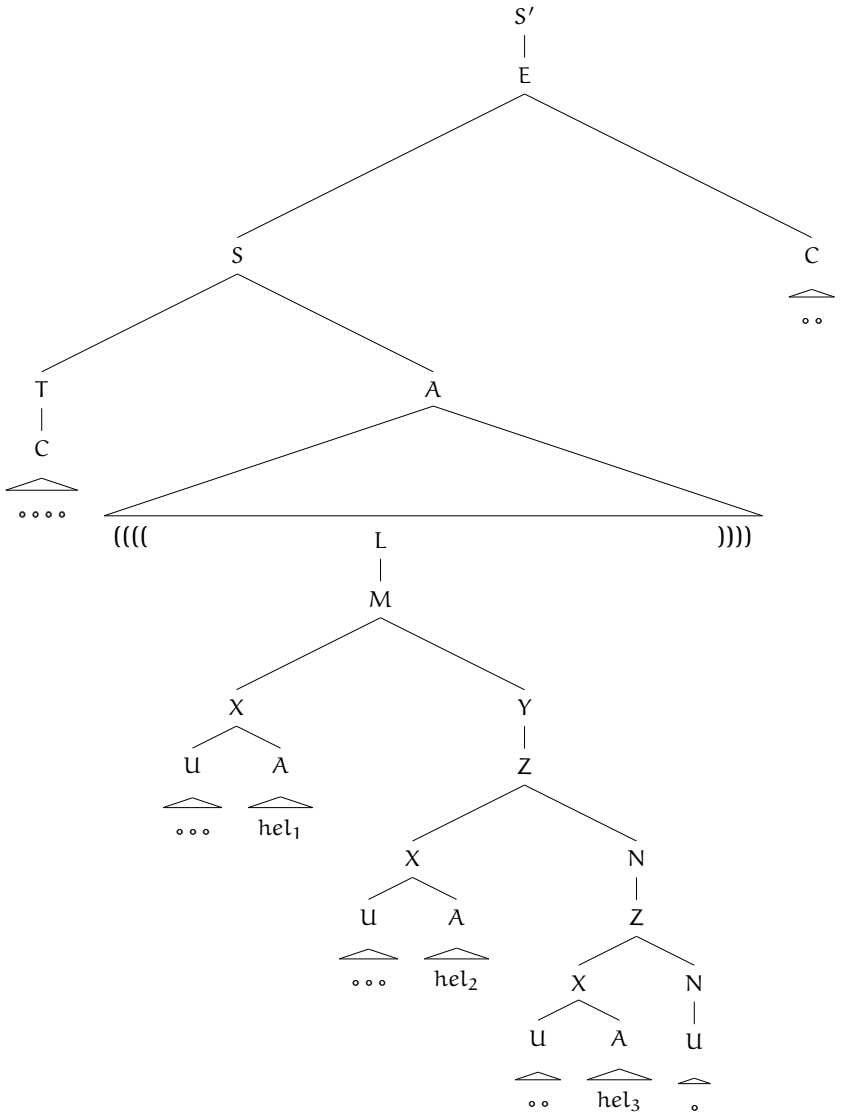


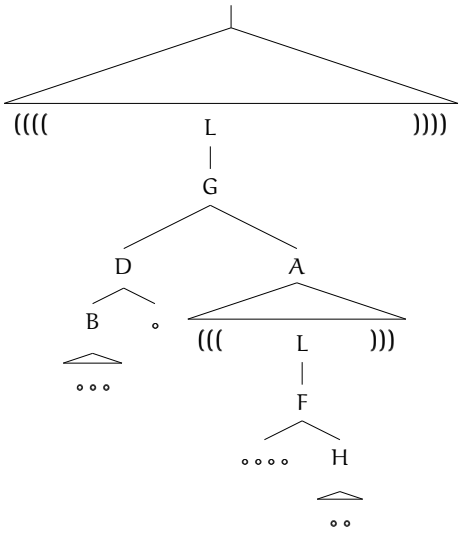
Figure 5.1.: **Exemplary parse tree using grammar $\widehat{\mathcal{G}}_u$.** Figure shows the unique parse tree for the dot-bracket word considered in Example 2.2.4 that corresponds to the planar secondary structure from Figure 2.7.

where $d(w)$ denotes the number of derivation trees $\widehat{\mathcal{G}}_u$ offers for w (for details, see [SS78]). Eliminating all but the variable associated with the axiom and simplifying (for this step, we made use of Mathematica) yields the single equation

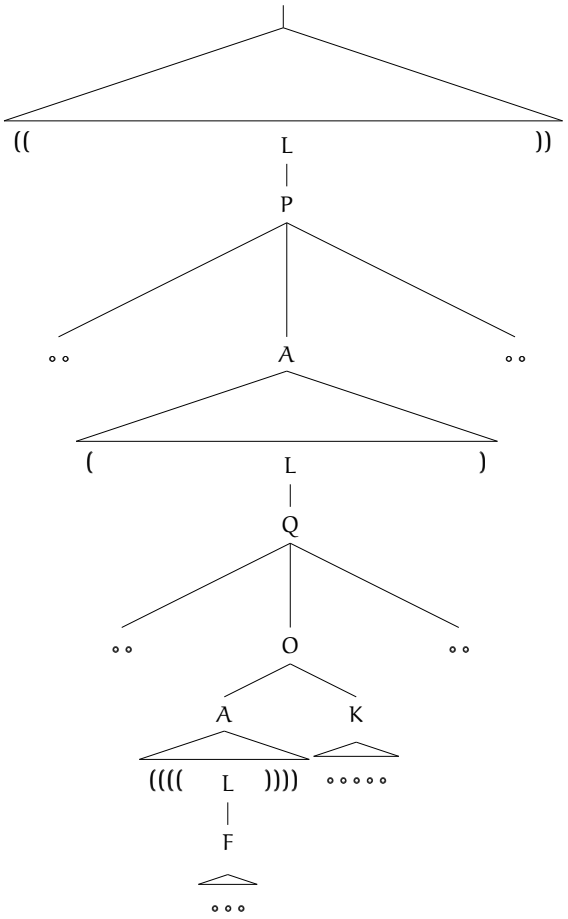
$$-z^5 + S[z]\,(-1+z)\left(-1+z\left(2-S[z]\,(-1+z)\,z+z^4\right)\right) = 0.$$

This equation is exactly the same one as obtained for the comparably much simpler grammar $\mathcal{G}_{e,3}$ from Example 3.3.7 (for which unambiguity has indeed been proven). This proves that for any size $n \geqslant 0$, both grammars have the same number of derivation trees for words of size n. Knowing that both grammars generate $\mathcal{L}$ and that $\mathcal{G}_{e,3}$ is unambiguous, the same can thus be concluded for the comprehensive grammar $\widehat{\mathcal{G}}_u$.

Note that $\widehat{\mathcal{G}}_u$ contains more production rules (and more different nonterminal symbols) than the SCFG considered in [NS11a], but it is ϵ-free and additionally, the right-hand side of every single production contains at most two nonterminal symbols, such that the resulting unranking algorithm has to consider less cases (that is, less "else if ( )" cases), which makes it significantly more efficient; for details, see [WN10] and the Appendix.



(a) Parse tree for subword $\mathtt{hel}_1$.



(b) Parse tree for subword $\mathtt{hel}_3$.

Figure 5.2.: **More exemplary parse trees using grammar $\widehat{\mathcal{G}}_u$.** Figures show particular subtrees of the tree presented in Figure 5.1.

Anyway, we trained the probabilities (relative frequencies) of $\widehat{\mathcal{G}}_u$ from the structures $s \in \mathcal{L}(\widehat{\mathcal{G}}_u)$ given in our biological database. The resulting probabilities are reported in Table 5.1. Their floating point approximations, rounded to the third decimal place, are collected in Table 5.2.

In oder to see if overfitting is an issue in connection with our rather complex grammar design and the resulting rich parameter set, that is, to see if our training set is large enough to derive reliable values for the rule probabilities, we performed the following experiments: We selected a random 90% (resp. 50%) portion of the original training set and re-estimated the probabilities of all the grammar rules. This process was iterated 40 times, resulting in a sample of 40 parameter sets. Finally, for each parameter we determined its variance along this sample of size 40. The corresponding values lay between 0 (resulting for intermediate symbols without alternatives; for whose productions a probability of 1 is predetermined) and 2.87652×10^{-6} (resp. 2.86242×10^{-5}). We can thus conclude that overfitting is no issue in connection with our sophisticated grammar and the training set used.

5.4.3. Derivation of the Algorithm

The elaborate SCFG $\widehat{\mathcal{G}}_u$ is appropriate for being used as the basis for the desired weighed unranking method: after having determined the RNF of this SCFG and the corresponding weighted combinatorial classes, we easily find a recursion for the `size` function (in the same ways as discussed in Example 5.3.3). Then, we can use the resulting weighted class sizes for the straightforward construction of the desired unranking algorithm.

Nonterminal Nt	Probabilities of Rules with Premise Nt			
S′	$\widehat{p}_1 := 1,$			
E	$\widehat{p}_2 := \frac{137}{6476},$	$\widehat{p}_3 := \frac{6339}{6476},$		
S	$\widehat{p}_4 := \frac{177}{12952},$	$\widehat{p}_5 := \frac{12775}{12952},$		
T	$\widehat{p}_6 := \frac{11086}{12775},$	$\widehat{p}_7 := \frac{1689}{12775},$		
C	$\widehat{p}_8 := \frac{14367}{148978},$	$\widehat{p}_9 := \frac{134611}{148978},$		
A	$\widehat{p}_{10} := 1,$			
L	$\widehat{p}_{11} := \frac{605069}{792975},$	$\widehat{p}_{12} := \frac{31912}{792975},$	$\widehat{p}_{13} := \frac{4912}{264325},$	$\widehat{p}_{14} := \frac{5821}{158595},$
	$\widehat{p}_{15} := \frac{1893}{264325},$	$\widehat{p}_{16} := \frac{2723}{31719},$	$\widehat{p}_{17} := \frac{38399}{792975},$	
G	$\widehat{p}_{18} := \frac{11667}{38399},$	$\widehat{p}_{19} := \frac{7235}{38399},$	$\widehat{p}_{20} := \frac{11831}{38399},$	$\widehat{p}_{21} := \frac{7666}{38399},$
D	$\widehat{p}_{22} := 1,$			
B	$\widehat{p}_{23} := \frac{4967}{12748},$	$\widehat{p}_{24} := \frac{7781}{12748},$		
F	$\widehat{p}_{25} := \frac{3912}{68075},$	$\widehat{p}_{26} := \frac{23208}{68075},$	$\widehat{p}_{27} := \frac{8191}{13615},$	
H	$\widehat{p}_{28} := \frac{8191}{40700},$	$\widehat{p}_{29} := \frac{32509}{40700},$		
P	$\widehat{p}_{30} := \frac{533}{4912},$	$\widehat{p}_{31} := \frac{1053}{4912},$	$\widehat{p}_{32} := \frac{2963}{14736},$	$\widehat{p}_{33} := \frac{7015}{14736},$
Q	$\widehat{p}_{34} := \frac{4986}{29105},$	$\widehat{p}_{35} := \frac{24119}{29105},$		
R	$\widehat{p}_{36} := \frac{2357}{5679},$	$\widehat{p}_{37} := \frac{3322}{5679},$		
V	$\widehat{p}_{38} := 1,$			
W	$\widehat{p}_{39} := 1,$			
O	$\widehat{p}_{40} := 1,$			
J	$\widehat{p}_{41} := \frac{27441}{84620},$	$\widehat{p}_{42} := \frac{57179}{84620},$		
K	$\widehat{p}_{43} := \frac{15731}{53725},$	$\widehat{p}_{44} := \frac{37994}{53725},$		
M	$\widehat{p}_{45} := 1,$			
X	$\widehat{p}_{46} := \frac{6196}{87035},$	$\widehat{p}_{47} := \frac{80839}{87035},$		
Y	$\widehat{p}_{48} := 1,$			
Z	$\widehat{p}_{49} := \frac{2812}{55123},$	$\widehat{p}_{50} := \frac{52311}{55123},$		
N	$\widehat{p}_{51} := \frac{7737}{17437},$	$\widehat{p}_{52} := \frac{9700}{17437},$		
U	$\widehat{p}_{53} := \frac{109939}{518817},$	$\widehat{p}_{54} := \frac{408878}{518817}.$		

Table 5.1.: **Trained probabilities for $\widehat{\mathcal{G}}_u$.** Table contains the probabilities (relative frequencies) for the production rules of the SCFG $\widehat{\mathcal{G}}_u$, obtained by training it using our biological database.

Nonterminal Nt	Probabilities of Rules with Premise Nt			
S′	$\widehat{p}_1 := 1.000,$			
E	$\widehat{p}_2 := 0.021,$	$\widehat{p}_3 := 0.979,$		
S	$\widehat{p}_4 := 0.014,$	$\widehat{p}_5 := 0.986,$		
T	$\widehat{p}_6 := 0.868,$	$\widehat{p}_7 := 0.132,$		
C	$\widehat{p}_8 := 0.096,$	$\widehat{p}_9 := 0.904,$		
A	$\widehat{p}_{10} := 1.000$			
L	$\widehat{p}_{11} := 0.763,$	$\widehat{p}_{12} := 0.040,$	$\widehat{p}_{13} := 0.019,$	$\widehat{p}_{14} := 0.037,$
	$\widehat{p}_{15} := 0.007,$	$\widehat{p}_{16} := 0.086,$	$\widehat{p}_{17} := 0.048,$	
G	$\widehat{p}_{18} := 0.304,$	$\widehat{p}_{19} := 0.188,$	$\widehat{p}_{20} := 0.308,$	$\widehat{p}_{21} := 0.200,$
D	$\widehat{p}_{22} := 1.000$			
B	$\widehat{p}_{23} := 0.390,$	$\widehat{p}_{24} := 0.610,$		
F	$\widehat{p}_{25} := 0.057,$	$\widehat{p}_{26} := 0.341,$	$\widehat{p}_{27} := 0.602,$	
H	$\widehat{p}_{28} := 0.201,$	$\widehat{p}_{29} := 0.799,$		
P	$\widehat{p}_{30} := 0.109,$	$\widehat{p}_{31} := 0.214,$	$\widehat{p}_{32} := 0.201,$	$\widehat{p}_{33} := 0.476,$
Q	$\widehat{p}_{34} := 0.171,$	$\widehat{p}_{35} := 0.829,$		
R	$\widehat{p}_{36} := 0.415,$	$\widehat{p}_{37} := 0.585,$		
V	$\widehat{p}_{38} := 1.000$			
W	$\widehat{p}_{39} := 1.000$			
O	$\widehat{p}_{40} := 1.000$			
J	$\widehat{p}_{41} := 0.324,$	$\widehat{p}_{42} := 0.676,$		
K	$\widehat{p}_{43} := 0.293,$	$\widehat{p}_{44} := 0.707,$		
M	$\widehat{p}_{45} := 1.0000$			
X	$\widehat{p}_{46} := 0.071,$	$\widehat{p}_{47} := 0.929,$		
Y	$\widehat{p}_{48} := 1.0000$			
Z	$\widehat{p}_{49} := 0.051,$	$\widehat{p}_{50} := 0.949,$		
N	$\widehat{p}_{51} := 0.444,$	$\widehat{p}_{52} := 0.556,$		
U	$\widehat{p}_{53} := 0.212,$	$\widehat{p}_{54} := 0.788.$		

Table 5.2.: **Rounded probabilities for $\widehat{\mathcal{G}}_u$.** Table contains floating point approximations of the probabilities (relative frequencies) for the production rules of the SCFG $\widehat{\mathcal{G}}_u$ (rounded to three decimal places).

In fact, for the construction of the complete algorithm, we simply have to use Algorithms 1 to 4 (Unranking of neutral classes, atomic classes, disjoint unions and cartesian products, respectively) and Algorithm 6 (Unranking of weighted classes) given in [WN10] as subroutines. However, to improve the worst-case complexity of the resulting unranking procedure from $\mathcal{O}(n^2)$ to $\mathcal{O}(n \cdot \log(n))$ by using the *boustrophedonic order* instead of the *sequential order*, a simple change in Algorithm 4 (Unranking of cartesian products) is necessary (for details, we refer to [Pon08]).

Anyway, a random RNA secondary structure of size n can easily be computed by drawing a random number $i \in \{0, \ldots, \text{size}(\mathcal{L}, n) - 1\}$ and then unranking the i^{th} structure of size n. The worst-case runtime complexity of this procedure is equal to that of unranking and is thus given by $\mathcal{O}(n \cdot \log(n))$ when using the boustrophedonic order. By repeating this procedure m times, a set of m (not necessarily distinct) random RNA secondary structures of size n can be generated in time $\mathcal{O}(m \cdot n \cdot \log(n))$, where a preprocessing time of $\mathcal{O}(n^2)$ is required for the computation of all (weighted) class sizes up to input length n.

A complete and detailed description of the derivation of our weighted unranking algorithm for (SSU and LSU r)RNA secondary structures can be found in Section A, since it is rather comprehensive and does not need to be presented here in detail, as the different steps for its generation simply correspond to those described in [WN10].

5.5. Discussion

The purpose of this section is to analyze the quality of randomly generated structures by considering some experimental results.

5.5.1. Parameters for Structural Motifs

As a first step, we decided to consider several important parameters related to particular structural motifs of RNA secondary structure and compare the observed statistical values derived from a native sample (here our biological database, that is the set of real-life RNA data that we used for deriving the distribution and thus the weights for the unranking algorithm) to those derived from a corresponding random sample (that is, a set of random structures generated by our algorithm). In order to obtain an appropriate random sample, we have generated exactly one random structure of size n for each native RNA structure of size n given in our database, such that for each occurring size n, the random sample and the native sample contain the same number of structures having this size.

The determined results are presented in Table 5.3. Comparing the specific values of all different parameters, we can see that our algorithm produces random RNA secondary structures that are, related to the different structural motifs and thus related to the expected shape of such structures, in most cases realistic. Obviously, this is a major improvement over existing approaches for the random generation of secondary structures of a given input size n (where the corresponding specific RNA sequence is *not* known, but only its length n), as those (sequence-independent) methods are only capable of generating structures uniformly at random for input size n. Furthermore, with the SCFG model used here, we have an new model for RNA secondary structures at hand which realistically reflects the structure of an RNA molecule and its basic structural motifs.

Parameter	Expected Value		Variance	
	Random	Native	Random	Native
num_{unp}	848.179	839.956	98964.7	103426.
num_{bps}	420.848	424.96	27785.3	31310.9
num_{urs}	179.73	181.822	4959.96	5117.47
num_{e}	1.	1.	0.	0.
num_{h}	36.6983	36.4818	196.935	185.596
num_{s}	321.18	324.26	16538.8	19343.4
num_{b}	20.6061	20.5782	87.1894	50.3103
num_{i}	26.1442	26.538	125.66	194.769
num_{m}	16.2197	17.1018	57.8874	41.0261
num_{hel}	99.6683	100.7	1549.24	1492.84
unp_{e}	106.014	79.8382	4039.69	3897.61
unp_{h}	6.93534	6.93188	18.4264	77.464
unp_{s}	—	—	—	—
unp_{b}	1.9948	1.99596	3.10283	6.87868
unp_{i}	7.14617	7.08869	16.5725	31.1197
unp_{m}	16.0122	16.2577	87.4906	195.497
unp_{hel}	—	—	—	—
bps_{e}	9.41479	6.94105	29.1956	6.30949
bps_{h}	—	—	—	—
bps_{s}	1.	1.	0.	0.
bps_{b}	1.	1.	0.	0.
bps_{i}	1.	1.	0.	0.
bps_{m}	2.68212	2.72734	1.12921	1.21643
bps_{hel}	4.22249	4.22006	13.6266	5.52299

Table 5.3.: **Expectation and variance of important parameters related to particular structural motifs of RNA secondary structure.** Values are derived from a native sample (our biological database) and from a random sample, respectively. num_x denotes the number of occurrences of motif x in one secondary structure and unp_x (bps_x) denotes the number of accessible unpaired bases (base pairs) in one substructure of type x. unp, bps, urs denote unpaired bases, base pairs and unpaired regions, whereas e, h, s, b, i, m, hel denote exterior loop, hairpin loop, stacked pair, bulge loop, interior loop, multiloop and helix, respectively.

5.5.2. Related Free Energies

For further investigation on the accuracy of our random generator, we take on a completely different point of view and consider thermodynamics. The reason behind this idea is that if an RNA secondary structure model induced by a SCFG shows a realistic behavior (expectation and variance) with respect to minimum free energy, then it is rather likely that our grammar also shows a realistic picture for all the different structural motifs of a molecule's folding (as the free energy of a molecule's structure is defined as the sum of the energy contributions of all its substructures).

Since we do not know the corresponding RNA sequences for the randomly generated structures, we can not use one of the common sequence-dependent thermodynamic models for RNAs. Therefore, we decided to consider both the *static* and *dynamic* free energy models defined in [NS11a] for RNA secondary structures with unknown sequence. Briefly, these models are based on the well-known Turner energy model [XSB+98, MSZT99] and model parameters can easily be derived from an arbitrary set of training data (by sequence counting). However, they basically differ in the model parameters used:

- In the static model, averaged free energy contributions for the distinguished structural motifs are considered. These averaged values actually represent the free energy contributions that have to be added for the respective whole substructures.
- For the dynamic model, corresponding average values for length-dependent free energy contributions (that depend on the number of unpaired or paired bases within particular substructures) are added for each component (unpaired base or base pair) in the respective motifs, such that in contrast to the static model, substructures of different lengths are assigned different free energy values).

Notably, parameters for any of these two models have (amongst others) been reported for the same biological database (of SSU and LSU rRNAs) that we consider in this chapter. In fact, both induced models have turned out to show a realistic behavior (with respect to free energies) and can therefore be used to judge the quality of random structures generated by our algorithm.

5.5.2.1. Unquantified Results

Similar to [NS11a], we denote the free energy of a given secondary structure $s \in \mathcal{L}$ according to the static and dynamic model by $g_{stat}(s)$ and $g_{dyn}(s)$, respectively. Moreover, the expected free energy and corresponding variance that have been analytically derived in that paper for any $n > 0$ are denoted by

$$\mu_{energy,n} := \mathbb{E}\left[energy(s) \mid \text{size}(s) = n\right] \quad \text{and} \quad \sigma^2_{energy,n} := \mathbb{V}\left[energy(s) \mid \text{size}(s) = n\right],$$

respectively, where $energy \in \{g_{stat}, g_{dyn}\}$. The corresponding confidence interval for $n > 0$ and $k > 1$, which contains at least $(100 - 100/k^2)$ percent of the energies in $\{energy(s) \mid s \in \mathcal{L}^n\}$ is denoted by

$$I_{energy,n}(k) := (\mu_{energy,n} - k\sigma_{energy,n}, \mu_{energy,n} + k\sigma_{energy,n}).$$

Before we start with our comparisons, note that for any sample set $\mathcal{S}$ of secondary structures, we can calculate the corresponding energy points

$$EP(\mathcal{S}, energy) := \{(\text{size}(s), energy(s)) \mid s \in \mathcal{S}\},$$

where $energy \in \{g_{stat}, g_{dyn}\}$. Obviously, we can also compute the corresponding "average energy points"

$$AvEP(\mathcal{S}, energy) := \left\{ \left(n, \mu_n := \frac{1}{\mathtt{card}(\mathcal{S}^n)} \sum_{s \in \mathcal{S}^n} energy(s) \right) \middle| \mathcal{S}^n \neq \emptyset \right\}$$

and the corresponding "energy variance points"

$$VarEP(\mathcal{S}, energy) := \left\{ \left(n, \sigma_n^2 := \frac{1}{\mathtt{card}(\mathcal{S}^n)} \sum_{s \in \mathcal{S}^n} (\mu_n - energy(s))^2 \right) \middle| \mathcal{S}^n \neq \emptyset \right\},$$

respectively. In the sequel, we will denote a random sample generated by our algorithm by $\mathcal{R}$ and a native sample (biological database) by $\mathcal{N}$.

In order to obtain an appropriate random sample for our energy comparisons, we derived a large set of random structures by generating 1000 RNA secondary structures for each of the sizes $n \in \{500, 1000, 1500, \ldots, 5000, 5500\}$ with our weighted unranking algorithm. To compare the energies of our randomly generated structures to the corresponding confidence interval(s), we decided to consider any $k \in \{\sqrt{2}, 2, \sqrt{10}, \sqrt{20}\}$, meaning the probability that the free energy of a random RNA secondary structure of size n lies within the corresponding interval is greater than 0.5, 0.75, 0.9, and 0.95, respectively.

Figure 5.3 shows a plot of the corresponding four confidence intervals (analytically derived, related to our biological data) along with the energy points for our random sample and for our native database, respectively, under the assumption of the static energy model. The corresponding plots for the dynamic energy model are shown in Figure 5.4. Looking at both figures, we immediately see that the energies for our set of randomly generated RNA secondary structures seem to fit to the ones for the considered RNA database and also to the corresponding analytically obtained energy results from [NS11a]. This observation becomes even more clear by considering Figures 5.5 and 5.6. There, we compare the previously introduced "average energy points" and "energy variance points" to the analytically determined expected free energy and corresponding variance from [NS11a], respectively.

5.5.2.2. Quantified Results

The previously considered energy comparisons have been presented only by unquantified plots. This may not be very satisfying, since it is obvious that the free energy would decrease with structure size and aside from this, it could have been expected that for large randomly generated sets of structures of a given size, the average energy and corresponding variance fit the analytically obtained energy results derived under the assumption of a basically equivalent SCFG model for secondary structures. Therefore, there is a need to consider some sort of quantification and additionally present corresponding quantified comparison results.

What really matters is the degree to which the energy ranges of the random structures agree, in distribution, with our biological database. This means we have to find out if the energies related to a random sample (generated by our unranking method) and those related to a native sample (given by the structures in our biological database) come from a common distribution. Consequently, we have to consider the energies of a random sample and those of a native one as two independent sets of values and determine the extend to which their distributions coincide, or in other words to test for significant differences between these two sets.

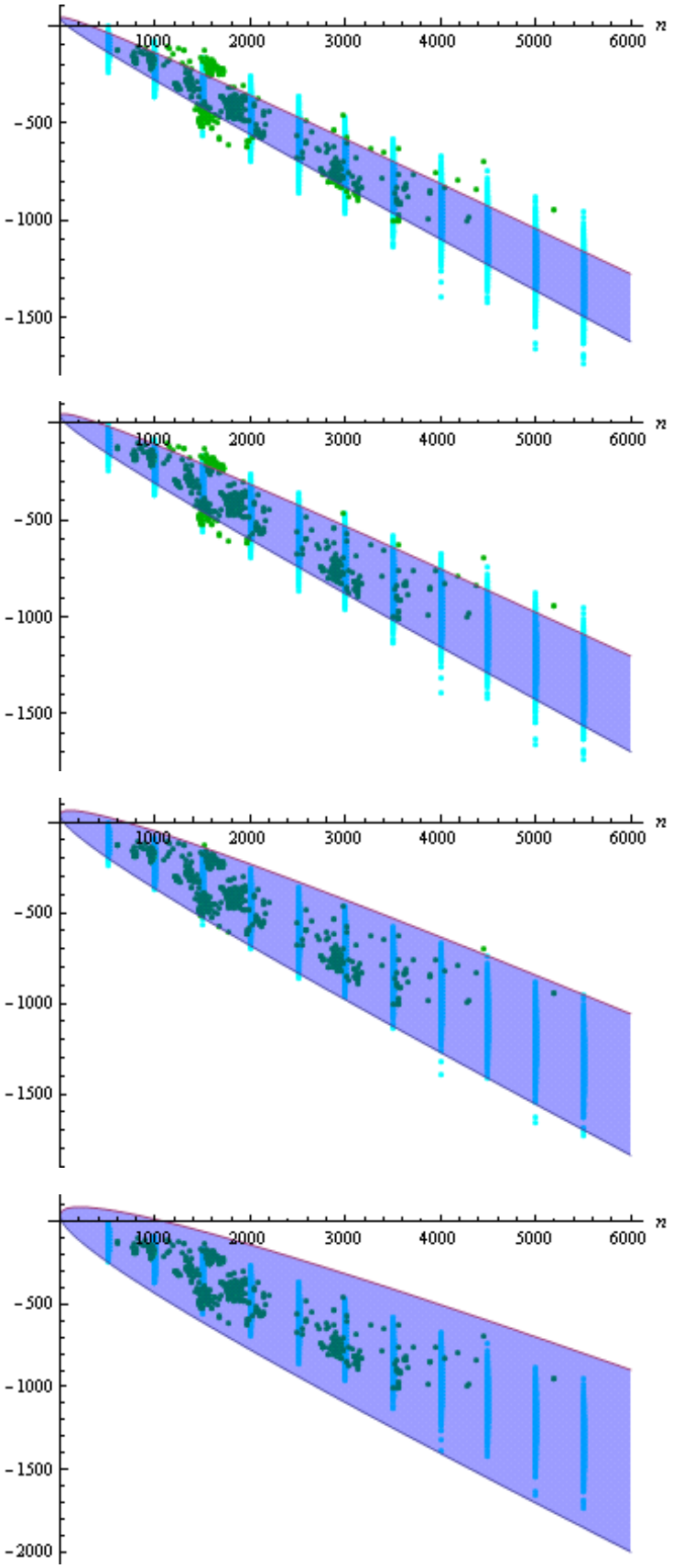


Figure 5.3.: **Confidence intervals and energy points under assumption of the static model.** Plots of the confidence intervals $I_{g_{stat},n}(k)$ for the static energy model (blue), for $k \in \{\sqrt{2}, 2, \sqrt{10}, \sqrt{20}\}$ (top to bottom), together with the corresponding energy points $EP(\mathcal{R}, g_{stat})$ for the random sample (cyan) and $EP(\mathcal{N}, g_{stat})$ for the native sample (green).

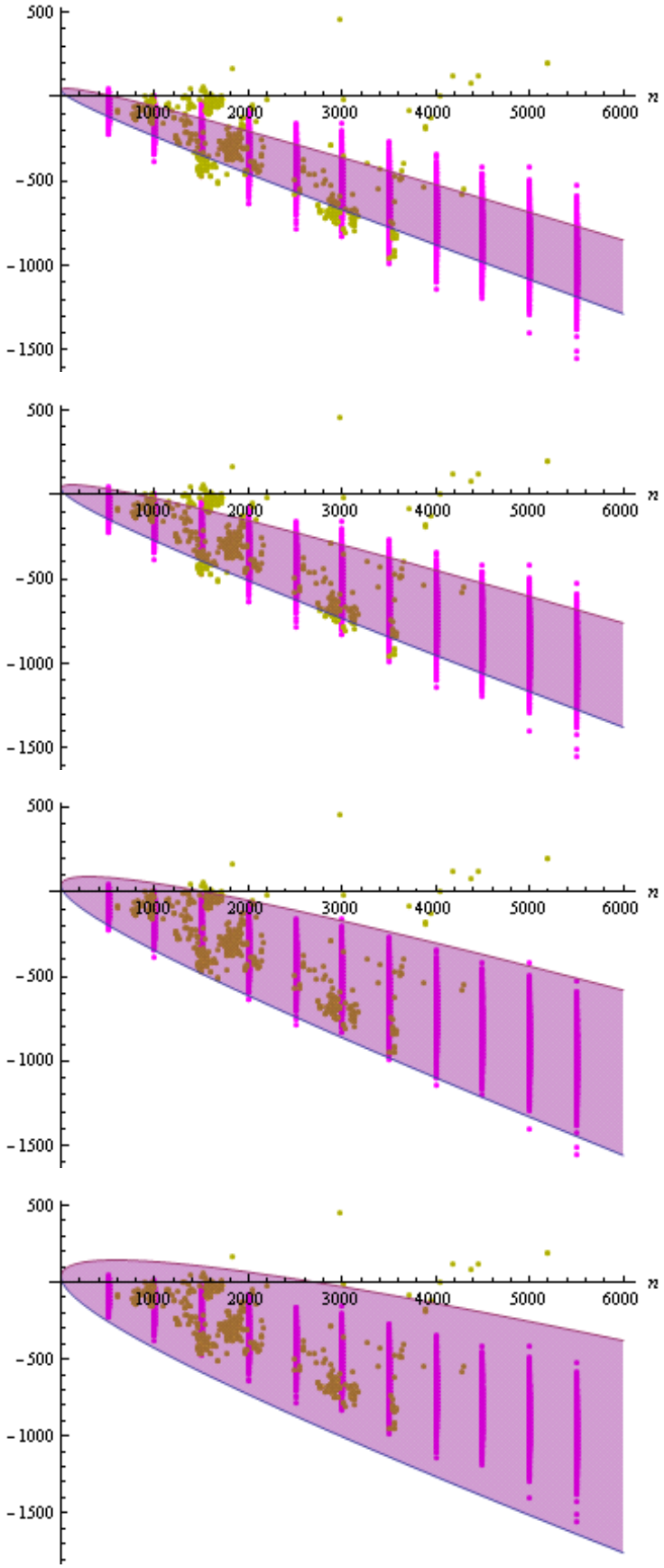


Figure 5.4.: **Confidence intervals and energy points under assumption of the dynamic model.** Plots of the confidence intervals $I_{g_{dyn},n}(k)$ for the dynamic energy model (purple), for $k \in \{\sqrt{2}, 2, \sqrt{10}, \sqrt{20}\}$ (top to bottom), together with the corresponding energy points $EP(\mathcal{R}, g_{dyn})$ for the random sample (magenta) and $EP(\mathcal{N}, g_{dyn})$ for the native sample (yellow).

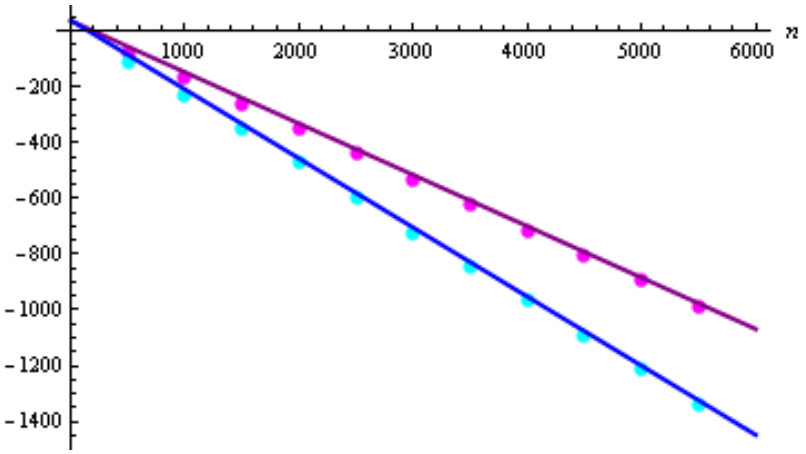


Figure 5.5.: **Expectations of the free energy.** Plots of the expected free energy $\mu_{g_{stat},n}$ (blue) and $\mu_{g_{dyn},n}$ (purple) of a random RNA secondary structure of size n, together with the "average energy points" $AvEP(\mathcal{R}, g_{stat})$ (cyan) and $AvEP(\mathcal{R}, g_{dyn})$ (magenta) for the random sample.

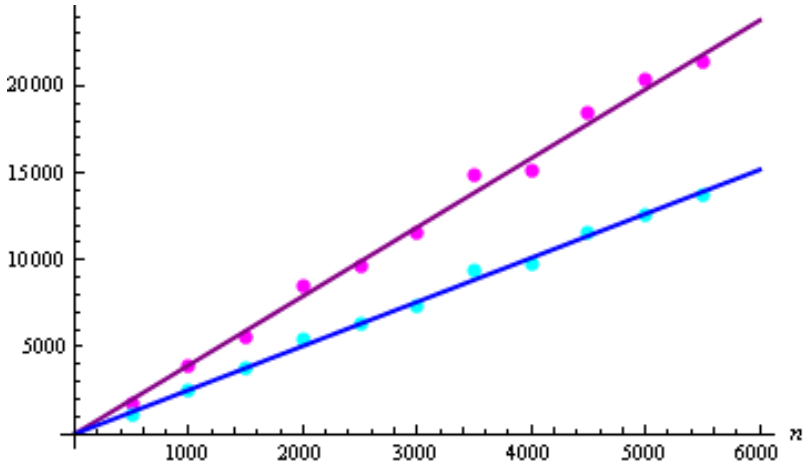


Figure 5.6.: **Variances of the free energy.** Plots of the variance of the expected free energy $\sigma^2_{g_{stat},n}$ (blue) and $\sigma^2_{g_{dyn},n}$ (purple) of a random RNA secondary structure of size n, together with the "energy variance points" $VarEP(\mathcal{R}, g_{stat})$ (cyan) and $VarEP(\mathcal{R}, g_{dyn})$ (magenta) for the random sample.

For this reason, we decided to apply one of the most common (non-parametric) significance tests known from statistics, the so-called *Mann-Whitney U-test* [MW47], which is widely used as statistical hypothesis test for assessing whether two independent samples of observations (with arbitrary sample sizes) come from the same distribution. It is also known as the *Wilcoxon rank-sum test* [Wil45] which however can only be applied for equal sample sizes.

Formally, this test is used to check whether the *null hypothesis* N_0 – which states that the two independent samples X and Y are identically distributed (that is, $F(X) = F(Y)$) – can be accepted or else, has to be rejected. More specifically, the result of such a test, the so-called p-*value*, is a probability answering the following question: If the two samples really have the same distribution, what is the probability that the observed difference is due to chance alone? In other words, were the deviations (differences between the two samples) the result of chance, or were they due to other factors and how much deviation can occur before one must conclude that something other than chance causes the differences? The p-value is called *statistically significant* if it is unlikely that the differences occurred by chance alone, according to a preliminary chosen threshold probability, the *significance level* α (common choices are, for example $\alpha \in \{0.10, 0.05, 0.01\}$). If $p \geqslant \alpha$, the deviation is small enough that chance alone

accounts for it; this is within the range of acceptable deviation. If $p < \alpha$, we must conclude that some factor other than chance causes the deviation to be so great, this will lead us to decide that the two sets come from different distributions.

For our analysis, we again decided to generate the same numbers of random structures for any size as are given for this size in our biological database, such that random and native sample contain the same numbers of structures for any occurring size (and hence the sample sizes are equal). Moreover, note that the unquantified results presented in Figures 5.3 and 5.4 might yield the assumption that for any structure size, some energy values of randomly generated structures are scattered too widely around the corresponding expected value, such that those randomly drawn secondary structures can not be considered realistic (neither with respect to thermodynamics nor with respect to structural composition and expected shape). In an attempt to disprove that assumption, we decided to perform a series of Wilcoxon tests by considering a number of different random samples. These samples are created by obeying a specified energy-based rejection scheme: Do not add a randomly generated structure of a given size to the sample if its free energy (according to the static or dynamic model or according to both models) lies outside the corresponding confidence interval(s). Formally, for any preliminary chosen value $k > 1$, a generated structure $s \in \mathcal{L}^n$ is added to the random sample iff

$$
\begin{aligned}
&[g_{stat} \in I_{g_{stat},n}(k) \text{ (variant “static”)}] \text{ or}\\
&[g_{dyn} \in I_{g_{dyn},n}(k) \text{ (variant “dynamic”)}] \text{ or}\\
&[g_{stat} \in I_{g_{stat},n}(k) \text{ and } g_{dyn} \in I_{g_{dyn},n}(k) \text{ (variant “both”)}];
\end{aligned}
$$

otherwise it is rejected. This means we accept only a specified deviation of the energy $\mathit{energy}(s)$ of the random structure s from the corresponding expected free energy $\mu_{\mathit{energy},n}$ and reject structures whose energy differs too much from the expected value. Note that for $k = \infty$ (confidence interval $I_{\mathit{energy},n}(k)$ contains 100 percent of the energies $\mathit{energy}(s)$ of all $s \in \mathcal{L}^n$), no structures are rejected. Hence, in this case, the corresponding random sample corresponds to the usual (unrestricted) output of our algorithm.

The Wilcoxon test results for our native sample together with any of a number of random sample sets generated in the previously described restricted manner, respectively, can be found in Table 5.4. As we can see, the best results are achieved for the unrestricted sample sets, where all free energies of randomly generated structures were allowed during the sample creation process. Moreover, these two results (for the unrestricted case $k = \infty$) are not statistically significant when considering the common significance level $\alpha = 0.05$, that is in both cases, we can assume that the energies of the random structures and those of the biological data follow a common distribution. These observations indicate that our weighted unranking algorithm produces random RNA secondary structures that are – related to the free energy of such structures (in expectation and variation) – in most cases realistic.

Besides that, it is obvious that the computed p-values are much better for the dynamic energy model than for the static one. This underlines the suggestion made in [NS11a] that, although both energy models have been proven to be realistic, due to the more realistic variation of free energies connected to varying loop length, the dynamic model should be used for possible applications. Since at least for the dynamic model, the random data fit very nicely with the native data, we can conclude that structures generated by our non-uniform random generation algorithm behave realistic with respect to free energies and – as the energy of the overall structure is assumed to be equal to the sum of the substructure energies – rather likely also with respect to appearance of the different structural motifs of RNA molecules.

Chosen Value of k	Percent Within Corr. Interval	Models Used for Rejection	Model for Native Energies	Model for Random Energies	Resulting Wilcoxon p-Value (approx.)
$\frac{10}{3\sqrt{11}} \approx 1.00504$	1	Dynamic	Dynamic	Dynamic	0.0008438
		Static	Static	Static	$1.872 \cdot 10^{-9}$
		Both	Dynamic	Dynamic	0.000507
		Both	Static	Static	$1.851 \cdot 10^{-10}$
$\frac{2\sqrt{5}}{\sqrt{19}} \approx 1.02598$	5	Dynamic	Dynamic	Dynamic	0.001567
		Static	Static	Static	$1.454 \cdot 10^{-10}$
		Both	Dynamic	Dynamic	0.0002654
		Both	Static	Static	$1.009 \cdot 10^{-9}$
$\frac{\sqrt{10}}{3} \approx 1.05409$	10	Dynamic	Dynamic	Dynamic	0.001374
		Static	Static	Static	$3.526 \cdot 10^{-9}$
		Both	Dynamic	Dynamic	0.0004116
		Both	Static	Static	$9.018 \cdot 10^{-10}$
$\frac{2}{\sqrt{3}} \approx 1.15470$	25	Dynamic	Dynamic	Dynamic	0.003618
		Static	Static	Static	$2.530 \cdot 10^{-7}$
		Both	Dynamic	Dynamic	0.001228
		Both	Static	Static	$1.162 \cdot 10^{-7}$
$\sqrt{2} \approx 1.41421$	50	Dynamic	Dynamic	Dynamic	0.02394
		Static	Static	Static	$1.278 \cdot 10^{-6}$
		Both	Dynamic	Dynamic	0.001389
		Both	Static	Static	$1.515 \cdot 10^{-7}$
2	75	Dynamic	Dynamic	Dynamic	0.1184
		Static	Static	Static	0.001034
		Both	Dynamic	Dynamic	0.0495
		Both	Static	Static	0.0009445
∞	100	—	Dynamic	Dynamic	0.4007
		—	Static	Static	0.08961

Table 5.4.: **Significance results for statistical hypothesis testing.** All values were computed by the Wilcoxon rank-sum method.

5.6. Conclusions

Altogether, we can finally conclude that the non-uniform random generation method proposed in this chapter is capable of producing appropriate output and may thus be used (for research issues as well as for practical applications) to randomly sample RNA secondary structures according to arbitrary probability distributions that can be learned from structural data sets. In fact, for any arbitrary type of (pseudoknot-free) RNA, a corresponding random sampler can be derived in the presented way. Due to the importance of efficient and accurate random sampling methods in practice and the proven efficiency of the discussed algorithm (reached by using the boustrophedon order rather than the sequential one), the results of this chapter are obviously valuable in their own right.

However, with respect to computational prediction of RNA secondary structures, this chapter can only be seen as an indicator for the power of complex stochastic grammar models to capture the expected shapes of RNA molecules. In fact, we completely abstracted from sequence information and considered only the structure size as input for our sampling method, such that the derived algorithm can not immediately be applied for RNA folding. However, due to the observation that the considered elaborate SCFG manages to realistically model the different canonical features of RNA secondary structure, where despite the heavyweight (by means of number of productions rules) grammar design, overfitting seems to be no problem in general, the development of new competitive stochastic RNA secondary structure prediction methods realized by statistical random sampling seems to be a solvable task. Basically, we only need to find a way to extend the presented realistic SCFG (or a similar) model to additionally deal with a given RNA sequence and sample structures connected to that sequence – in proportion to the induced distribution on all feasible foldings for it. Unfortunately, it is not obvious how this could be realized by means of unranking, such that the presented algorithm can not directly be upgraded in order to obtain an efficient and reliable new method for single sequence structure prediction.

Chapter 6

Evaluation of a Sophisticated SCFG Design for RNA Secondary Structure Prediction

In this chapter, we introduce and evaluate a sophisticated SCFG design (similar to that considered in Chapter 5) that mirrors state-of-the-art physics-based RNA structure prediction procedures by distinguishing between all canonical features of RNA that imply different energy rules. This SCFG actually serves as the foundation for a statistical sampling algorithm for RNA secondary structures of a single sequence that represents a probabilistic counterpart to the sampling extension of the PF approach. Furthermore, some new ways to derive meaningful structure predictions from generated sample sets are presented. They are used to compare the predictive accuracy of our model to that of other probabilistic and energy-based prediction methods.

Particularly, comparisons to lightweight SCFGs and corresponding CLLMs for RNA structure prediction indicate that more complex SCFG designs might yield higher accuracy but eventually require more comprehensive and pure training sets. Investigations on both the accuracies of predicted foldings and the overall quality of generated sample sets (especially on the abstraction level of shapes of generated structures that is relevant for biologists) yield the conclusion that the Boltzmann distribution of the PF sampling approach is more centered than the ensemble distribution induced by the sophisticated SCFG model, which implies a greater structural diversity within generated samples. In general, neither of the two distinct ensemble distributions is more adequate than the other and the corresponding results obtained by statistical sampling can be expected to bare fundamental differences, such that the method to be preferred for a particular input sequence strongly depends on the considered RNA type.

6.1. Motivation and Objectives

As regards computational structure prediction, early probabilistic approaches such as [KH99] seem to have chosen the structure of their SCFG rather arbitrarily; at least, there is almost no discussion about the motivation for the choice of the productions. This problem has first been addressed in [DE04], where nine different SCFGs have been evaluated in connection with RNA secondary structure prediction. Aiming at an exploration on how different SCFG designs affect the accuracy of single sequence RNA secondary structure prediction methods, the authors observed that fairly simple SCFGs achieve respectable prediction accuracies, but – despite the uncertainties in Turner's energy model – the best physics-based methods still generally perform significantly better than the best SCFGs. Therefore, the authors of [DE04] raised the following questions, which will be addressed by this chapter:

1) Could an appropriately designed sophisticated SCFG be able to outperform the existing MFE methods for single sequence prediction?

2) How would an (unambiguous)[1] SCFG mirroring state-of-the-art physics-based algorithms (that is, a grammar with specific productions for all structural motifs for which there are different thermodynamic parameters or energy rules) perform?

As already noted, in order to improve the predictive accuracy of energy-based algorithms, (some of) the corresponding thermodynamic parameters might be estimated or improved via statistical inference methods, by taking advantage of a particular RNA database. This obviously strongly relates to the estimation of the grammar parameters of a sophisticated SCFG design as described in question 2). Actually, if a certain energy parameter value for a specific structural motif can be statistically estimated from a given set of real-world RNA data, then the corresponding grammar parameter for the production that generates this motif can effectively be trained from the same data set, yielding a one-to-one correspondence between estimated thermodynamic and grammar parameter values. Hence, it might be assumed that a sophisticated SCFG satisfying the conditions formulated in question 2) has a similar predictive power than modern physics-based algorithms that employ elaborate free energy models.

Moreover, due to the benefit caused by departing from the common MFE approach to considering the sampling extension of the PF approach (see the discussion in Section 3.2), it seems reasonable to rely on Boltzmann samples rather than on single MFE structures in order to address question 1). Accordingly, we decided to oppose the Boltzmann samples to corresponding samples obtained by a SCFG version of `Sfold`'s statistical sampling strategy based on an appropriately designed grammar that actually meets the requirements raised in question 2). This means we will employ an efficient statistical sampling algorithm that incorporates comprehensive structural features and – instead of the recent thermodynamic Turner parameters – additional information obtained from trusted databases of real-world RNA structures in order to generate probabilistic counterparts of the Boltzmann samples. Actually, just like in the PF variant, secondary structures are sampled rigorously from the ensemble distribution of all feasible foldings for a given input sequence, but the distribution will be induced by the parameter values of the underlying SCFG.

Altogether, due to the before mentioned connection of thermodynamic parameters and probabilities of a sophisticated grammar (especially if both are estimated statistically), it seems adequate to put the following hypothesis which will be examined within this chapter:

H_0: The Boltzmann distribution implied by a thermodynamic PF approach and the ensemble distribution induced by a corresponding (sophisticated) SCFG are similar and thus yield comparable statistical sampling results (that is, no significant differences of the generated sample sets can be expected).

[1] A structurally ambiguous SCFG mirror of modern energy-based algorithms for single sequence structure prediction has already been described in [RE00].

According to the preceding explanations, the main objectives of this chapter are given as follows: We will answer the two important questions 1) and 2) already raised in [DE04] (according to the previously mentioned aspects) and essentially check whether hypothesis H_0 can be verified. Therefore, we will first define a sophisticated SCFG that represents a probabilistic mirror to the optimization schemes applied in modern MFE based DP routines and statistical sampling approaches based on free energies and PFs. Contrary to the grammar considered in Chapter 5, this SCFG is actually designed to represent an exact probabilistic mirror to the diverse recursions and formulae for calculating all equilibrium PFs and sampling probabilities that are needed for the elaborate statistical sampling procedure applied in the `Sfold` software.

Another take on the same kind of problems but with slightly different intensions can be found in [RLE12]. There, in order to explore a range of probabilistic models of increasing complexity, and to directly compare probabilistic, thermodynamic, and discriminative approaches, a computational tool is created that can parse a wide spectrum of RNA grammar architectures (including the standard nearest-neighbor model and more) using a generalized super-grammar that can be parameterized with probabilities, energies, or arbitrary scores. The authors put forward that discriminative training is not required, simple ML learning is enough. Therefore, their tool uses only generative training, not discriminative. Parameters can, however, be imported from other sources. Using their tool, the authors show that probabilistic nearest-neighbor models perform comparably to (but not significantly better than) discriminative methods and that complex statistical models are prone to overfitting RNA structure.

6.2. Outline

The rest of the present chapter is organized as follows: Section 6.3 describes the SCFG model for secondary structures that will be used as the foundation for the probabilistic sampling approach. The complete sampling strategy is introduced in Section 6.4 and Section 6.5 proposes several appropriate ways for deriving particular predictions from generated structure samples. Notably, some of them deal with a new mechanism for controlling the prediction accuracy (by a sensitivity/PPV trade-off parameter γ_{t-o}) similar to the one implemented in the `CONTRAfold` software (details will follow).

Section 6.6 examines the benefits and potential drawbacks of using a sophisticated SCFG like ours compared to lightweight SCFGs and corresponding CLLMs for RNA structure prediction. We find that using a more complex SCFG design might actually yield a higher prediction accuracy but requires a more comprehensive and pure training set to ensure that all parameters are appropriately estimated. To address hypothesis H_0, Section 6.6 additionally discusses the potentials and pitfalls of the SCFG based sampling method compared to the sampling extension of the PF approach as implemented in the `Sfold` software, where both the quality of generated sample sets and their applicability to the problem of RNA structure prediction are investigated. These comparisons include results connected to abstract shapes of sampled structures as introduced in [JRG08] (see Section 2.2.3 for details), as this abstraction level is of great interest and relevance for biologists. One of the prime observations is that the SCFG induced distribution implies a greater structural diversity within generated samples, as it seems to be less centered than the Boltzmann energy distribution. Moreover, the distinct comparisons indicate that using a lean database of mixed RNA classes results in improper estimators of the needed grammar parameters, such that in these cases the PF approach usually generates more realistic samples. The SCFG approach generally produces more accurate sample sets if a rich and pure training set is available.

In summary, free energy based samplers are proven to have stronger abilities for generalization or vice versa, approaches based on a sophisticated SCFG can be fitted to a specific class of RNA (where they show high predictive accuracy possibly implied by *non-energetic* effects which find

their way into the parameter set) without generalization to other biological classes (maybe because there those effects behave differently). Finally, Section 6.7 summarizes our findings and hints at some interesting matters for further research.

6.3. Used SCFG Model

According to our objectives motivated in Section 6.1, our SCFG should be constructed to represent a mirror to the free energy model employed in `Sfold`'s sampling procedure, which means we have to take care of the fact that all distinct structural features of RNA for which there are different energy rules and free energy parameters according to the underlying thermodynamic model have to be modeled by corresponding distinct production rules. Briefly, at any point, the desired SCFG must be capable of distinguishing between exactly the same mutually exclusive and exhaustive cases that have to be considered in the recursions for calculating the equilibrium PFs as defined in [DL03]. Then, the inside and outside values derived for a given sequence on the basis of that SCFG (see Section 3.3.7.2 for details) can be used in a straightforward fashion – along with the corresponding SCFG parameters (rule probabilities) – in order to define the needed conditional sampling probabilities that directly correspond to those applied in `Sfold`'s elaborate PF based sampling algorithm.

Note that in order to facilitate the consideration of the different classes of secondary structures usually described in literature, our SCFG should additionally be parameterized to impose the two relevant restrictions on the class of all feasible foldings: first, a minimum length of $\min_{HL}$ for hairpin loops and second, a minimum number of $\min_{hel}$ consecutive base pairs for helices, such that we can easily use our grammar to generate the language of all RNA secondary structures according to any possible combination of the common choices $\min_{HL} \in \{1,3\}$ and $\min_{hel} \in \{1,2\}$. By using different intermediate symbols for the distinct loop types and their respective substructures, we obtain the following sophisticated SCFG design for modeling the formal language of all RNA secondary structures that obey to both structural parameters $\min_{HL}$ and $\min_{hel}$:

Definition 6.3.1 *The (unambiguous) SCFG $\mathcal{G}_s$ generating exactly all feasible[2] secondary structures is given by $\mathcal{G}_s = (\mathcal{I}_{\mathcal{G}_s}, \Sigma_{\mathcal{G}_s}, \mathcal{R}_{\mathcal{G}_s}, S)$, where*

$$\mathcal{I}_{\mathcal{G}_s} = \{S, T, C, A, P, L, F, H, G, B, M, O, N, U, Z\},$$

$\Sigma_{\mathcal{G}_s} = \{(,), \circ\}$ *and for* $m_h := \min_{HL} \geqslant 1$ *and* $m_s := \min_{hel} \geqslant 1$, $\mathcal{R}_{\mathcal{G}_s}$ *contains exactly the following rules:*

$p_1 : S \to T$, $\rightsquigarrow$ *initiate exterior loop*

$p_2 : T \to C$, $p_3 : T \to A$, $p_4 : T \to CA$, $p_5 : T \to AT$, $p_6 : T \to CAT$, $\rightsquigarrow$ *exterior loop*

$p_7 : C \to ZC$, $p_8 : C \to Z$, $\rightsquigarrow$ *strands in exterior loop*

$p_9 : A \to ({}^{m_s}L)^{m_s}$, $\rightsquigarrow$ *initiate helix*

$p_{10} : P \to (L)$, $\rightsquigarrow$ *extend helix*

$p_{11} : L \to F$, $p_{12} : L \to P$, $p_{13} : L \to G$, $p_{14} : L \to M$, $\rightsquigarrow$ *initiate any loop*

$p_{15} : F \to Z^{m_h - 1}H$, $\rightsquigarrow$ *start hairpin loop*

$p_{16} : H \to ZH$, $p_{17} : H \to Z$, $\rightsquigarrow$ *extend hairpin loop*

$p_{18} : G \to BA$, $p_{19} : G \to AB$, $p_{20} : G \to BAB$, $\rightsquigarrow$ *type of bulge/interior loop*

$p_{21} : B \to ZB$, $p_{22} : B \to Z$, $\rightsquigarrow$ *strands in bulge/interior loop*

[2] Note that we allow only such structures that do neither contain hairpin loops of less than $\min_{HL}$ unpaired bases nor helices of less than $\min_{hel}$ consecutive base pairs.

$p_{23} : M \to UAO$, $\rightsquigarrow$ *first substructure of multiple loop*

$p_{24} : O \to UAN$, $\rightsquigarrow$ *second substructure of multiple loop*

$p_{25} : N \to UAN$, $p_{26} : N \to U$, $\rightsquigarrow$ k^{th} *substructure of multiple loop,* $k \geqslant 3$

$p_{27} : U \to ZU$, $p_{28} : U \to \epsilon$, $\rightsquigarrow$ *strands in multiple loop*

$p_{29} : Z \to \circ.$ $\rightsquigarrow$ *unpaired base*

The unambiguity of that grammar can be proven along the lines of Remark 5.4.1. Note that the productions $F \to Z^{m_h-1}H$ and $A \to (^{m_s}L)^{m_s}$ ensure that neither hairpin loops of less than m_h unpaired nucleotides nor helices of less than m_s consecutive base pairs are generated.

Obviously, the (unambiguous) grammar $\mathcal{G}_s$ can immediately be transformed into a corresponding (ambiguous) SCFG $\mathcal{G}_r$ that models the language of all RNA sequences: we only have to replace $\Sigma_{\mathcal{G}_s} = \{(,), \circ\}$ by $\Sigma_{\mathcal{G}_r} := \{a, c, g, u\}$ and the three rules $A \to (^{m_s}L)^{m_s}$, $P \to (L)$ and $Z \to \circ$ by corresponding new productions generating *valid*[3] base pairs and unpaired bases, respectively (as described by the end of Section 3.3.6.1). Finally, in order to guarantee that appropriate probabilities are used for the production rules of the SCFG $\mathcal{G}_r$, we can assign relative frequencies (which can be derived from an arbitrary training set of known RNA sequences with corresponding secondary structures) to the elements in $\mathcal{R}_{\mathcal{G}_r}$, yielding a consistent SCFG (see Section 3.3.6.3).

However, we can equivalently only consider the initial grammar $\mathcal{G}_s$ with *transition probabilities* for the productions in $\mathcal{R}_{\mathcal{G}_s}$ and – in order to be able to model structures on RNA sequences – two additional sets of *emission probabilities* for unpaired bases (that is, for each $x \in \Sigma_{\mathcal{G}_r}$) and for base pairs (that is, for every $x_1x_2 \in \Sigma^2_{\mathcal{G}_r}$), as discussed in Section 3.3.6.5. Accordingly, the probability of each production rule in $\mathcal{G}_r$ that generates one or more individual base pairs or an individual unpaired base is given by the product of the corresponding transition probability (for $A \to (^{m_s}L)^{m_s}$, $P \to (L)$ or $Z \to \circ$ in $\mathcal{R}_{\mathcal{G}_s}$) and the respective emission probabilities (for base pairs or unpaired bases). For example, if $m_s = 2$, then $\Pr(A \to acLgu \in \mathcal{R}_{\mathcal{G}_r}) = \Pr_{tr}(A \to ((L)) \in \mathcal{R}_{\mathcal{G}_s}) \cdot \Pr_{em}(au) \cdot \Pr_{em}(cg)$. In the sequel, we will always use this separation into emission and transition probabilities in order to perform our evaluations.

It should be mentioned that the trained transition and emission probabilities are obviously linked in the straightforward mathematical sense, that is the probabilities of the different transitions with same left-hand side, as well as the emissions for unpaired and paired bases, respectively, must sum up to unity. Moreover, all emission probabilities come from the same distribution, that is for any considered loop type, we use the same emission probabilities for unpaired bases located within and base pairs closing a corresponding loop. Consequently, the number of free parameters that have to be trained is given by

$$\mathtt{card}(\mathcal{R}_{\mathcal{G}_s}) - \mathtt{card}(\mathcal{I}_{\mathcal{G}_s}) + \mathtt{card}(\Sigma_{\mathcal{G}_r})^2 + \mathtt{card}(\Sigma_{\mathcal{G}_r}) = 29 - 15 + 16 + 4 = 34.$$

Note that this rather moderate number (compared to the heavyweight grammar design) effectively results from linking together the emissions of base pairs generated with different rules instead of going strictly with the grammar definition which implies using different trained distributions for any such rule (here $p_9 : A \to (^{m_s}L)^{m_s}$ and $p_{10} : P \to (L)$). This simplification obviously reduces the dimensionality of the parameter space in a significant way (especially for $\min_{hel} > 1$), and is also justified due to observations made from considering trusted RNA

[3] Here, we decided to consider any possible pair as valid base pair, where non-canonical ones are mostly prohibited due to small probabilities. Thus, in contrast to the thermodynamics based PF approach which can only handle canonical base pairs, our algorithm is able to deal with arbitrary base pairs, in a convenient way: when using appropriate probabilities, canonical base pairs will be very likely and non-canonical ones will be very unprobable (but not necessarily impossible) to be formed. However, since non-canonical base pairs are usually not permitted in secondary structure models (to limit the number of possible foldings), it would also be adequate to allow only canonical ones. The probabilities for non-canonical base pairs would then be equal to zero.

databases (trained distributions usually are very similar) and having a closer look at the Turner energy parameters (many tables, excluding the stacking table and some others, contain only a few different values in total).

6.4. Algorithm

In this section, we give a complete derivation of all results needed for a probabilistic statistical sampling algorithm for RNA secondary structures according to the SCFG model defined in the last section. Just like the PF variant, the sampling algorithm has two basic steps: Its first step (preprocessing) computes the inside and outside probabilities for all substrings of an RNA sequence based on the considered SCFG. These inside and outside values are used for calculating *conditional* sampling probabilities for all considered cases. The second step (structure generation) is basically the same as with PFs, which means it takes the form of a recursive sampling algorithm to randomly draw secondary structures according to the sampling probabilities derived in step one. By applying the algorithm to a biological RNA sequence, a statistically representative sample of secondary structures can quickly be generated once the preprocessing step for deriving the inside and outside values is completed.

6.4.1. Computing Inside and Outside Probabilities

In order the determine all inside and outside variables for a given sequence r, we decided to use the SCFG $\mathcal{G}_r$ as the basis for a special version of Earley's algorithm. In particular, we chose to rely on the formalism presented in [Goo98, Goo99] for describing parsers, which is called *semiring parsing* and has been found to be especially useful for deriving grammar specific parsers that compute outside values (see Section 3.3.7.2). As already mentioned in Section 3.3.7.1, the advantage of using an Earley-style parser description is that the corresponding semiring parser can handle general grammars, which means we do not have to transform the grammar $\mathcal{G}_r$ into CNF. This is especially useful in our case, since the number of productions of the CNF of grammar $\mathcal{G}_r$ would be huge. For this reason, computing the needed inside and outside values with the traditional inside-outside algorithm for grammars in CNF would be significantly less efficient.

6.4.1.1. Notations

In accordance with Section 3.3.7.2, for A an intermediate symbol of the considered grammar $\mathcal{G}_r$, let $\alpha_A(i,j)$ denote the inside variables and $\beta_A(i,j)$ denote the outside variables for a given sequence r of size n, $1 \leqslant i,j \leqslant n$.

Furthermore, recall that Earley's algorithm considers the so-called *dotted rules* that correspond to the productions of the used grammar (see Section 3.3.7.1). Formally, when using the grammar $\mathcal{G}_r$, a symbol $\bullet \notin \Sigma_{\mathcal{G}_r} \cup \mathcal{I}_{\mathcal{G}_r}$ is used to mark the current position up to which the parsing has proceeded; according to the fact that Earley's algorithm parses input words from left to right, this symbols must thus be "shifted" from the leftmost position to the rightmost one in each production rule of the grammar used for parsing. Accordingly, for each production $rule \in \mathcal{R}_{\mathcal{G}_r}$ of the form

$$rule = A \to \alpha_1 \dots \alpha_k$$

with $\alpha_i \in \mathcal{I}_{\mathcal{G}_r} \cup \Sigma_{\mathcal{G}_r}$, $1 \leqslant i \leqslant k$, we need to consider the $k+1$ dotted rules

$$\begin{aligned} rule_0 &= A \to \bullet\alpha_1 \dots \alpha_k, \\ rule_1 &= A \to \alpha_1 \bullet \dots \alpha_k, \\ &\dots, \end{aligned}$$

$$rule_{k-1} = A \to \alpha_1 \ldots \bullet\, \alpha_k,$$
$$rule_k = A \to \alpha_1 \ldots \alpha_k \bullet;$$

if $rule = A \to \epsilon$, we only consider $rule_0 = A \to \epsilon\bullet$. The set of all dotted rules for grammar $\mathcal{G}_r$ will be denoted by $\mathcal{R}_{\mathcal{G}_r,\bullet}$ in the sequel. Moreover, each set of $k+1$ dotted rules corresponding to a production $rule = A \to \alpha_1 \ldots \alpha_k \in \mathcal{R}_{\mathcal{G}_r}$ will be denoted by $\mathcal{R}_{\mathcal{G}_r,\bullet}(rule)$, such that

$$\bigcup_{rule \in \mathcal{R}_{\mathcal{G}_r}} \mathcal{R}_{\mathcal{G}_r,\bullet}(rule) = \mathcal{R}_{\mathcal{G}_r,\bullet}.$$

Last but not least, recall that for defining the desired Earley-based semiring parser, we use an *item-based* parser description (see also Section 3.3.7.1). Therefore, in contrast to the usual inside-outside algorithm for computing the inside values $\alpha_A(i,j)$ and outside values $\beta_A(i,j)$, $1 \leqslant i,j \leqslant n$, for A an intermediate symbol of the considered grammar and n the length of the input word, the corresponding semiring parser used here computes inside and outside values for *items*. Here, the items slightly differ from the ones introduced in Section 3.3.7.1: they are defined by three components, having the form

$$[i, \mathrm{ind}(rule), j],$$

where for a given input word r of length n, i and j, $1 \leqslant i,j \leqslant n+1$, define positions in r (that is, in front of the first character, in between two characters or after the last character). However, $\mathrm{ind}(rule)$ denotes the *index* of production $rule \in \mathcal{R}_{\mathcal{G}_r,\bullet}$ in an appropriate ordering (details will follow later) of production set $\mathcal{R}_{\mathcal{G}_r,\bullet}$. Anyway, an item of the form

$$[i, \mathrm{ind}(A \to \alpha \bullet \beta), j]$$

also asserts that

$$A \Rightarrow \alpha\beta \Rightarrow^* r_i \ldots r_{j-1}\beta.$$

Consequently, the inside and outside values are computed for each production $rule \in \mathcal{R}_{\mathcal{G}_r,\bullet}$ and not as needed for each nonterminal symbol $A \in \mathcal{I}_{\mathcal{G}_r}$. However, the needed inside and outside values $\alpha_A(i,j)$ and $\beta_A(i,j)$ can easily be derived from the corresponding inside and outside results for items $[i, \mathrm{ind}(A \to \gamma\bullet), j]$, as we will see later.

6.4.1.2. Deriving the Inside and Outside Values of Items

First, we want to describe how to compute the inside and outside values of all items, using a corresponding item-based description of an Earley-style parser.

Inside Computation To obtain the inside values of all items $[i, \mathrm{ind}(rule), j]$, $1 \leqslant i,j \leqslant n+1$ (for an RNA sequence r of size n) and $rule \in \mathcal{R}_{\mathcal{G}_r,\bullet}$, by semiring parsing based on Earley's algorithm, we can use the following formulae, which we derived according to [Goo98, Goo99]:

- Scanning:

$$\mathrm{inside}[i, \mathrm{ind}(A \to \alpha w_j \bullet \beta), j+1] = \delta_{w_j,r_j} \cdot \mathrm{inside}[i, \mathrm{ind}(A \to \alpha \bullet w_j\beta), j]$$

  where for w_j an arbitrary terminal symbol of the underlying grammar $\mathcal{G}_r$ and r_j the (terminal) symbol read at position j of the input string r,

$$\delta_{w_j,r_j} = \begin{cases} 1, & \text{if } w_j = r_j, \\ 0, & \text{if } w_j \neq r_j, \end{cases}$$

according to the definition of Kronecker's delta.

- Prediction:

$$\text{inside}[j, \text{ind}(B \to \bullet\gamma), j] = \begin{cases} \Pr(B \to \gamma), & \text{if } S \Rightarrow^* r_1 \dots r_{j-1} B\delta \text{ for some } \delta, \\ 0, & \text{else,} \end{cases}$$

where $\Pr(\mathit{rule})$ denotes the probability of production $\mathit{rule} \in \mathcal{R}_{\mathcal{G}_r}$ as given by the SCFG $\mathcal{G}_r$. Note that this top-down filtering is usually made by Earley's algorithm to ensure that only such items can be predicted that might later be used by the completion rule. However, this is not needed here, since for any superfluously predicted item, the resulting probability will later be set to 0 by a scan. Thus, we can simply predict all items[4] by

$$\text{inside}[j, \text{ind}(B \to \bullet\gamma), j] = \Pr(B \to \gamma).$$

- Completion:

$$\begin{aligned} &\text{inside}[i, \text{ind}(A \to \alpha B \bullet \beta), j] \\ &= \sum_{i \leqslant k \leqslant j} \text{inside}[i, \text{ind}(A \to \alpha \bullet B\beta), k] \cdot \sum_{\mathit{rule}_B \in \mathcal{R}_B} \text{inside}[k, \text{ind}(\mathit{rule}_B), j], \end{aligned}$$

where

$$\mathcal{R}_B = \{\mathit{rule} \in \mathcal{R}_{\mathcal{G}_r, \bullet} \mid \mathit{rule} = B \to \gamma\bullet\}.$$

Moreover, the desired semiring parser algorithm for the correct computation of all inside values additionally requires the definition of a convenient ordering of the considered items $[i, \text{ind}(\mathit{rule}), j]$, for $1 \leqslant i, j \leqslant n+1$ and $\mathit{rule} \in \mathcal{R}_{\mathcal{G}_r, \bullet}$, such that no item precedes any other item on which it depends. Details on how we derived the corresponding ordering used here will follow. In principle, we can define an ordering by first and last parameters $i, j \in \{1, \dots, n+1\}$ that matches the order of consideration of items induced by Earley's algorithm and especially an appropriate ordering of the considered rule set $\mathcal{R}_{\mathcal{G}_r, \bullet}$ by indices (p, q), for $p \in \{1, \dots, \texttt{card}(\mathcal{R}_{\mathcal{G}_r})\}$ and $q \in \{0, \dots, k(p)\}$, where $k(p)$ denotes the conclusion length of the production $\mathit{rule} \in \mathcal{R}_{\mathcal{G}_r}$ indexed by p.

Based on the previously introduced formulæ and the appropriate ordering that will be formally defined hereafter, we finally obtain Algorithm 1 that shows how to perform the complete inside computation.

Ordering of Items In [Goo98, Goo99], each item x is associated with a "bucket" B, formally $\text{bucket}(x) = B$. The buckets have to be ordered as follows: If item y depends on item x, then $\text{bucket}(x) \leqslant \text{bucket}(y)$. There are two types of buckets: looping buckets and non-looping buckets. In fact, if items x and y depend (directly or indirectly) on each other, then they are both associated with a special looping bucket B, such that $\text{bucket}(x) = B = \text{bucket}(y)$. A bucket is also called looping bucket if an item in it depends on itself. In all other cases, the bucket is called non-looping.

If item x is associated with a non-looping bucket, then its value can easily be computed, as this value depends only on the values of items in earlier buckets. However, in the case of item x being associated with a looping bucket, the computation is much more complex, which is due to the fact that the value of x then depends potentially on the values of other items in the same bucket. In fact, this means that infinite loops may occur, for two different reasons: First, if the

[4] Note that this simplification leads to a larger number of items to be considered. Since the efficiency of Earley's algorithm depends on the cardinality of the item lists, this approach usually requires more runtime in practice. However, the implementation of the parser becomes much easier by simply predicting all items.

Algorithm 1 Computation of inside values

Input: RNA sequence r of length $n \geqslant 1$,
set $\mathcal{R}_{\mathcal{G}_r,\bullet}$ of production rules used by Earley's algorithm for parsing r with $\mathcal{G}_r$, and
probabilities $\Pr(\mathit{rule})$ of the productions $\mathit{rule} \in \mathcal{R}_{\mathcal{G}_r}$, trained on RNA structure data.

for $j = 1 \ldots n+1$ **do**
  for $i = j \ldots 1$ **do**
    for $p = 1 \ldots \mathtt{card}(\mathcal{R}_{\mathcal{G}_r})$ **do**
      for $q = 0 \ldots k(p)$ **do**
        $\mathit{rule} = \mathrm{ind}^{-1}(p, q)$ /*$\mathit{rule} \in \mathcal{R}_{\mathcal{G}_r,\bullet}$ is the rule having index (p, q) in our ordering.*/
        if $\mathit{rule} = A \to \alpha w_{j-1} \bullet \beta$ **then**
          /* Scanning: */
          $\mathrm{inside}[i, (p, q), j] = \delta_{w_{j-1}, r_{j-1}} \cdot \mathrm{inside}[i, (p, q-1), j-1]$
        else if $\mathit{rule} = B \to \bullet\gamma$ **then**
          /* Prediction: */
          $\mathrm{inside}[j, (p, q), j] = \Pr(B \to \gamma)$
        else if $\mathit{rule} = A \to \alpha B \bullet \beta$ **then**
          /* Completion: */
          $\mathrm{inside}[i, (p, q), j] = \sum_{i \leqslant k \leqslant j} \left(\mathrm{inside}[i, (p, q-1), k] \cdot \sum_{\mathit{rule}_B \in \mathcal{R}_B} \mathrm{inside}[k, \mathrm{ind}(\mathit{rule}_B), j]\right)$
        end if
      end for
    end for
  end for
end for

values of two items in the same bucket are mutually dependent, or second if an item depends on its own value. Although such infinite loops may require computation of infinite sums, there exists a way to efficiently compute or approximate them, as shown in [Goo98, Goo99].

Fortunately, as the SCFG $\mathcal{G}_r$ considered here is loop-free, each item $[i, \mathrm{ind}(\mathit{rule} \in \mathcal{R}_{\mathcal{G}_r,\bullet}), j]$ can be associated with a non-looping bucket B (of size one). Thus, considering the restriction that no item precedes any item on which it depends, an ordering on the items $[i, \mathrm{ind}(\mathit{rule}), j]$ can be defined by appropriately iterating over positions i and j, respectively, as well as by using a suitable ordering (indexing) of the elements in $\mathcal{R}_{\mathcal{G}_r,\bullet}$.

Since we use an Earley-style parser, it is obvious that in order to calculate all values of items $[i, \mathrm{ind}(\mathit{rule}), j]$, $1 \leqslant i, j \leqslant n+1$ and $\mathit{rule} \in \mathcal{R}_{\mathcal{G}_r,\bullet}$, we first have to iterate over all values j from 1 to $n+1$. This means we "shift" the symbol $\bullet$[5] from left to right. For each value of $j \in \{1, \ldots, n+1\}$, we then have to iterate over all values i from j down to 1. Thus, we can first make a prediction for $i = j$ and then scanning or completion steps for $i < j$. However, the problem of finding an appropriate ordering of $\mathcal{R}_{\mathcal{G}_r,\bullet}$ that has to be applied for every pair of fixed positions i and j in order to derive the values for items $[i, \mathrm{ind}(\mathit{rule}), j]$, $\mathit{rule} \in \mathcal{R}_{\mathcal{G}_r,\bullet}$, is more complicated.

Here, the ordering of the rules in $\mathcal{R}_{\mathcal{G}_r,\bullet}$ is defined by index values (p, q), given as follows:

- The first index $p \in \{1, \ldots, \mathtt{card}(\mathcal{R}_{\mathcal{G}_r})\}$ corresponds to a set of productions $\mathcal{R}_{\mathcal{G}_r,\bullet}(\mathit{rule}) \subset \mathcal{R}_{\mathcal{G}_r,\bullet}$ (the one that was derived from production $\mathit{rule} \in \mathcal{R}_{\mathcal{G}_r}$) and

- the corresponding second index $q \in \{0, \ldots, \mathtt{card}(\mathcal{R}_{\mathcal{G}_r,\bullet}(\mathit{rule}))\}$ corresponds to a single production $\mathit{rule}_q \in \mathcal{R}_{\mathcal{G}_r,\bullet}(\mathit{rule})$ (the one in which symbol $\bullet$ occurs after the q^{th} symbol in the conclusion, see above).

[5] Recall that symbol $\bullet$ is used to mark the current position j, $1 \leqslant j \leqslant n+1$, in the input word up to which the parsing has proceeded.

Obviously, this ordering within the sets $\mathcal{R}_{\mathcal{G}_r,\bullet}(\mathit{rule})$ is appropriate, since if $\mathit{rule} = A \to \alpha B \beta$ is indexed by $p \in \{1, \ldots, \mathtt{card}(\mathcal{R}_{\mathcal{G}_r})\}$, then item

$$[i, (p, q) = \mathrm{ind}(A \to \alpha B \bullet \beta), j]$$

depends on item

$$[i, (p, q-1) = \mathrm{ind}(A \to \alpha \bullet B \beta), j']$$

for $j' \leqslant j$. Consequently, it remains to find a suitable distinct index $p \in \{1, \ldots, \mathtt{card}(\mathcal{R}_{\mathcal{G}_r})\}$ for any set $\mathcal{R}_{\mathcal{G}_r,\bullet}(\mathit{rule})$ corresponding to the original production $\mathit{rule} \in \mathcal{R}_{\mathcal{G}_r}$, such that the resulting ordering ensures that no item precedes any item on which it depends.

It is easy to see that for predictions and scanning steps, no problems can occur due to our ordering (implied by index q) within any set $\mathcal{R}_{\mathcal{G}_r,\bullet}(\mathit{rule})$. Thus, the center of attention has to be laid on the completion steps. In fact, suppose the value of an item

$$[i, (p, q) = \mathrm{ind}(A \to \alpha B \bullet \beta), j]$$

has to be computed by completion. Then, this value depends on the values of items

$$[i, (p, q-1) = \mathrm{ind}(A \to \alpha \bullet B \beta), k] \quad \text{and} \quad [k, \mathrm{ind}(\mathit{rule}_B \in \mathcal{R}_B), j],$$

for $i \leqslant k \leqslant j$. Whereas in all cases, the value of $[i, (p, q-1), k]$ has been computed at this point (due to our ordering of $\mathcal{R}_{\mathcal{G}_r,\bullet}(A \to \alpha B \beta)$ and since $k \leqslant j$), problems may arise for $[k, \mathrm{ind}(\mathit{rule}_B \in \mathcal{R}_B), j]$. Particularly, if α can not be the empty word, that is if $|\alpha| \geqslant 1$ holds, then we only have to consider $i+1 \leqslant k \leqslant j$, in which cases values of items $[k, \mathrm{ind}(\mathit{rule}_B \in \mathcal{R}_B), j]$ have already been determined in previous iterations, since $k > i$. However, if α can be empty, then

$$[i, (p, q) = \mathrm{ind}(A \to \alpha B \bullet \beta), j]$$

also depends on

$$[i, (p', q') = \mathrm{ind}(\mathit{rule}_B = B \to \gamma \bullet \in \mathcal{R}_B), j].$$

Thus, $B \to \gamma$ has to be considered before $A \to \alpha B \beta$, which implies $p' < p$ must hold. In fact, as this holds for any $\mathit{rule}_B \in \mathcal{R}_B$, we can conclude that if α can be empty, then in an appropriate ordering, $A \to \alpha B \beta \in \mathcal{R}_{\mathcal{G}_r}$ has to be placed after all productions $B \to \gamma \in \mathcal{R}_{\mathcal{G}_r}$ that have premise B.

According to these observations, the desired ordering can easily be constructed in the following way:

- Start by assigning the smallest indices $p \in \{1, \ldots, \mathtt{card}(\mathcal{R}_{\mathcal{G}_r})\}$ to productions of the form $\mathit{rule} = I \to t\delta$, where the first symbol t of the conclusion is any terminal symbol from $\Sigma_{\mathcal{G}_r}$.

- Then, assign the remaining indices to the other sets $\mathcal{R}_{\mathcal{G}_r,\bullet}(\mathit{rule})$, for $\mathit{rule} \in \mathcal{R}_{\mathcal{G}_r}$, taking into account the previously discussed restrictions.

For the sake of simplicity, let us first consider the grammar $\mathcal{G}_s$ that models the language of all secondary structures. For this grammar, we could for example use the following ordering of the corresponding rule set $\mathcal{R}_{\mathcal{G}_s}$, that is the following ordering by first indices $p \in \{1, \ldots, \mathtt{card}(\mathcal{R}_{\mathcal{G}_s})\}$:

p	*rule*	p	*rule*	p	*rule*	p	*rule*
1	$Z \to \circ$,	2	$A \to ({}^{m_s}L)^{m_s}$,	3	$P \to (L)$,		
4	$C \to ZC$,	5	$C \to Z$,	6	$H \to ZH$,	7	$H \to Z$,
8	$B \to ZB$,	9	$B \to Z$,	10	$U \to ZU$,	11	$U \to \epsilon$,
12	$T \to C$,	13	$T \to A$,	14	$T \to CA$,		
15	$T \to AT$,	16	$T \to CAT$,	17	$F \to Z^{m_h-1}H$,		
18	$G \to BA$,	19	$G \to AB$,	20	$G \to BAB$,		
21	$M \to UAO$,	22	$O \to UAN$,	23	$N \to UAN$,	24	$N \to U$,
25	$L \to F$,	26	$L \to P$,	27	$L \to G$,	28	$L \to M$,
29	$S \to T$.						

The derivation of a corresponding ordering for the considered SCFG $\mathcal{G}_r$ generating all RNA sequences is straightforward. Thus, we have defined an appropriate ordering of $\mathcal{R}_{\mathcal{G}_r,\bullet}$ by indices (p, q), for $p \in \{1, \ldots, \mathtt{card}(\mathcal{R}_{\mathcal{G}_r})\}$ and $q \in \{0, \ldots, k(p)\}$, where $k(p) = \mathtt{card}(\mathcal{R}_{\mathcal{G}_r,\bullet}(\mathit{rule}))$ if $\mathcal{R}_{\mathcal{G}_r,\bullet}(\mathit{rule})$ can be found under index p.

Outside Computation Once the inside values have been computed, the corresponding outside values of all items $[i, \mathrm{ind}(\mathit{rule}), j]$, $1 \leqslant i, j \leqslant n+1$ (for an RNA sequence r of size n) and $\mathit{rule} \in \mathcal{R}_{\mathcal{G}_r,\bullet}$ can be calculated with Algorithm 2.

Algorithm 2 Computation of outside values

Input: RNA sequence r of length $n \geqslant 1$,
set $\mathcal{R}_{\mathcal{G}_r,\bullet}$ of production rules used by Earley's algorithm for parsing r with $\mathcal{G}_r$, and
the corresponding inside values (computed by Algorithm 1).

```
outside[1, ind(S → T•), n + 1] = 1
for j = n + 1 ... 1 do
  for i = 1 ... j do
    for p = card(R_{G_r}) ... 1 do
      for q = k(p) ... 0 do
        rule = ind^{-1}(p, q) /*rule ∈ R_{G_r,•} is the rule having index (p, q) in our ordering.*/
        if rule = A → α w_j • β then
          /* Scanning (reverse): */
          outside[i, (p, q − 1), j] = δ_{w_j, r_j} · outside[i, (p, q), j + 1]
        else if rule = B → •γ then
          /* Prediction (reverse): */
          do nothing
        else if rule = A → αB • β then
          /* Completion (reverse): */
          for k = i ... j do
            outside[i, (p, q − 1), k] =
              outside[i, (p, q − 1), k] + outside[i, (p, q), j] · (Σ_{rule_B ∈ R_B} inside[k, ind(rule_B), j])
            for rule_B ∈ R_B do
              outside[k, ind(rule_B), j] =
                outside[k, ind(rule_B), j] + outside[i, (p, q), j] · inside[i, (p, q − 1), k]
            end for
          end for
        end if
      end for
    end for
  end for
end for
```

This Earley-based semiring parsing algorithm uses the reversed previously introduced ordering of items and makes use of the following formulæ for the outside computations (for details, we refer to [Goo98, Goo99]):

- Scanning (reverse):

$$\text{outside}[i, \text{ind}(A \to \alpha \bullet w_j \beta), j] = \delta_{w_j, r_j} \cdot \text{outside}[i, \text{ind}(A \to \alpha w_j \bullet \beta), j+1].$$

- Prediction (reverse):

  There is nothing to do, since this value is obtained while performing a (reverse) completion computation.

- Completion (reverse):

$$\text{outside}[i, \text{ind}(A \to \alpha \bullet B\beta), k] = \text{outside}[i, \text{ind}(A \to \alpha \bullet B\beta), k] + \text{outside}[i, \text{ind}(A \to \alpha B \bullet \beta), j] \cdot \sum_{rule_B \in \mathcal{R}_B} \text{inside}[k, \text{ind}(rule_B), j]$$

  and

$$\text{outside}[k, \text{ind}(rule_B), j] = \text{outside}[k, \text{ind}(rule_B), j] + \text{outside}[i, \text{ind}(A \to \alpha B \bullet \beta), j] \cdot \text{inside}[i, \text{ind}(A \to \alpha \bullet B\beta), k],$$

  for $i \leqslant k \leqslant j$ and $rule_B \in \mathcal{R}_B$.

Since the number of production rules considered for the inside and outside computations is given by $\texttt{card}(\mathcal{R}_{\mathcal{G}_r, \bullet})$ and is thus not dependent on the input size, Algorithms 1 and 2 need cubic time and quadratic space in the worst-case.

6.4.1.3. Deriving the Needed Inside and Outside Probabilities

Finally, since for a given sequence r of length n, an item of the form $[i, \text{ind}(A \to \alpha\bullet), j+1]$, $1 \leqslant i, j \leqslant n+1$, asserts that $A \Rightarrow \alpha \Rightarrow^* r_i \dots r_j$, it is easy to see that

$$\sum_{rule = A \to \alpha\bullet} \text{inside}[i, \text{ind}(rule), j+1] = \sum_{rule \in \mathcal{R}_A} \text{inside}[i, \text{ind}(rule), j+1] = \alpha_A(i, j)$$

is the probability of a leftmost derivation that generates the subword $r_i \dots r_j$ of r from the intermediate symbol A.

Furthermore, recall that $\beta_A(i, j)$ is defined as the probability of a derivation which, starting with the intermediate symbol S (the axiom of the grammar $\mathcal{G}_r$), generates the expression $r_1 \dots r_{i-1}\ A\ r_{j+1} \dots r_n$. For this outside probability, it obviously does not matter what subword $r_i \dots r_j$ of r is derived from intermediate symbol A, that is it is independent on which rule $A \to \alpha\bullet \in \mathcal{R}_A$ generates subword $r_i \dots r_j$. Consequently, for $rule = A \to \alpha\bullet \in \mathcal{R}_A$, the outside value for item $[i, \text{ind}(rule), j+1]$ is either equal to zero (if $r_i \dots r_j$ can not be derived from nonterminal A using production *rule*), or it is equal to the outside value for any items $[i, \text{ind}(rule'), j+1]$, where $rule' \in \mathcal{R}_A$ and $\text{outside}[i, \text{ind}(rule'), j+1] \neq 0$ (which means that production $rule'$ can be used to generate subword $r_i \dots r_j$ of r from A). Accordingly, the needed outside probability for symbol A is equal to one of the non-zero values (if any) of the corresponding production rules with premise A, which can be written as:

$$\max_{rule = A \to \alpha\bullet} \text{outside}[i, \text{ind}(rule), j+1] = \max_{rule \in \mathcal{R}_A} \text{outside}[i, \text{ind}(rule), j+1] = \beta_A(i, j).$$

Thus, for any given RNA sequence r of size n, we can derive the desired inside and outside probabilities $\alpha_A(i,j)$ and $\beta_A(i,j)$, for each $A \in \mathcal{I}_{\mathcal{G}_r}$ and $1 \leqslant i,j \leqslant n$, by computing the inside and outside values of all items by semiring parsing based on an Earley-style parser for the SCFG $\mathcal{G}_r$ and afterwards using the results for each *rule* $\in \mathcal{R}_{\mathcal{G}_r,\bullet}$ of the form *rule* $= A \to \alpha\bullet$ to obtain the corresponding probabilities for each $A \in \mathcal{I}_{\mathcal{G}_r}$ (in the previously described way). Consequently, for sequence r of size n, there result cubic time complexity and quadratic memory requirements for the computation of all probabilities $\alpha_A(i,j)$ and $\beta_A(i,j)$, $A \in \mathcal{I}_{\mathcal{G}_r}$ and $1 \leqslant i,j \leqslant n$.

6.4.2. Sampling Structures According to SCFG Model

In this section, we first present equations for computing the needed sampling probabilities for all considered cases. Afterwards, we give a detailed description of the corresponding sampling algorithm, including detailed information on how to use the respective sampling probabilities. Note that these parts are written in a similar way as the corresponding section in [DL03], in order to illustrate the similarities and differences that arise when computing the sampling probabilities according to either approach.

Before we will define sampling probabilities for mutually exclusive and exhaustive cases that correspond to those derived in [DL03] with the PF approach, recall that when using the PF method, one has to choose a constant value for the parameter $\max_{bulge}$ which defines the maximum allowed size of single-stranded regions in bulge and interior loops (for applications, $\max_{bulge} = 30$ is a common choice) in order to ensure that the worst-case time complexity remains cubic. However, such restrictions are not necessary to improve the performance of an SCFG based sampling algorithm (see Section 6.4.2.1), but the corresponding sampling strategy can easily be implemented to deal with $\max_{bulge}$, such that (the default value) $\max_{bulge} = \infty$ has to be chosen to avoid restrictions on bulge and interior loops.

Nevertheless, we decided to make use of the two structural parameters $\min_{HL}$ and $\min_{hel}$ to be able to avoid hairpin loops of less than $\min_{HL}$ nucleotides and helices of less than $\min_{hel}$ consecutive base pairs, such that each paired substructure actually consists of at least

$$\min_{ps} := 2 \cdot \min_{hel} + \min_{HL}$$

bases. This enables us to compare the different results obtained for each combination of the commonly used values $\min_{HL} \in \{1,3\}$ and $\min_{hel} \in \{1,2\}$ to the corresponding results derived with the PF approach (which always implicitly uses $\min_{hel} = 1$ and $\min_{HL} = 3$).

6.4.2.1. Equations for Computation of Sampling Probabilities

Basically, the definitions of the needed sampling probabilities for all regular loop types can be derived in a similar fashion as the respective ones considered for the PF approach, where we eventually use only the corresponding inside and outside values, together with the probabilities of the production rules of the considered SCFG. Therefore, the aim of this section is to present details concerning the formal definitions of all relevant conditional sampling probabilities (in accordance with the ones presented in [DL03], that is using similar notions etc.) and their usages, respectively, separately for each loop type.

Sampling Probabilities for Exterior Loops We start by considering a fragment $R_{i,j}$ that does not lie within any regular loop, meaning that consists only of free bases of the exterior loop. Obviously, we can either leave the whole fragment unfolded or else, we can choose a first free base pair $r_h.r_l$ of the exterior loop (that starts a paired substructure on $R_{i,j}$). As we have to take into account all possible cases for choosing and combining r_h and r_l on the considered fragment, we define

- $P_0^E(i,j)$ as the sampling probability for leaving $R_{i,j}$ single-stranded,
- $P_{ij}^E(i,j)$ as that for pairing r_i with r_j (that is, case $h = i$ and $l = j$),
- $\{P_{hj}^E(i,j,h)\}$ as those for cases where $i < h < l = j$ and
- $\{P_{il}^E(i,j,l)\}$ as those for cases $h = i < l < j$.

Moreover, let

- $\{P_{hl}^E(i,j,h)\}$ be the probabilities for first sampling h for cases where $i < h < l < j$ and
- $\{\widehat{P}_{hl}^E(j,h,l)\}$ be those for sampling l after h is sampled (in any case $i < h < l < j$).

Notably, these probabilities directly correspond to the ones defined in Section 3.2.4.1 on the basis of PFs and thus distinguish between the same mutually exclusive and exhaustive cases, namely those illustrated by Figure 3.1. However, making use of the convenient relationship between inside and outside values (as described in Section 3.3.7.2 and pictured by Figure 3.7c), we find:

$$
\begin{aligned}
P_0^E(i,j) &= \frac{1}{p^E(i,j)} \cdot \beta_T(i,j) \cdot (\alpha_C(i,j) \cdot \Pr(T \to C)),\\
P_{ij}^E(i,j) &= \frac{1}{p^E(i,j)} \cdot \beta_T(i,j) \cdot (\alpha_A(i,j) \cdot \Pr(T \to A)),\\
P_{hj}^E(i,j,h) &= \frac{1}{p^E(i,j)} \cdot \beta_T(i,j) \cdot (\alpha_C(i,h-1) \cdot \alpha_A(h,j) \cdot \Pr(T \to CA)),\\
P_{il}^E(i,j,l) &= \frac{1}{p^E(i,j)} \cdot \beta_T(i,j) \cdot (\alpha_A(i,l) \cdot \alpha_T(l+1,j) \cdot \Pr(T \to AT)),\\
P_{hl}^E(i,j,h) &= \frac{1}{p^E(i,j)} \cdot \beta_T(i,j) \cdot (\alpha_C(i,h-1) \cdot \alpha_{AT}(h,j) \cdot \Pr(T \to CAT)),\\
\widehat{P}_{hl}^E(j,h,l) &= \frac{1}{\alpha_{AT}(h,j)} \cdot (\alpha_A(h,l) \cdot \alpha_T(l+1,j)),
\end{aligned}
$$

where

$$
\alpha_{AT}(i,j) = \sum_{l=(i-1)+\min_{ps}}^{(j-1)} (\alpha_A(i,l) \cdot \alpha_T(l+1,j))
$$

and

$$
p^E(i,j) = \beta_T(i,j) \cdot \alpha_T(i,j).
$$

Since the probabilities of all mutually exclusive and exhaustive cases sum up to 1, we have

$$
P_0^E(i,j) + P_{ij}^E(i,j) + \sum_{h=(i+1)}^{(j+1)-\min_{ps}} P_{hj}^E(i,j,h) + \sum_{l=(i-1)+\min_{ps}}^{(j-1)} P_{il}^E(i,j,l) + \sum_{h=(i+1)}^{j-\min_{ps}} P_{hl}^E(i,j,h) = 1,
$$

and, under the condition that $P_{hl}^E(i,j,h) > 0$, also

$$
\sum_{l=(h-1)+\min_{ps}}^{(j-1)} \widehat{P}_{hl}^E(j,h,l) = 1.
$$

Sampling Probabilities for Substructures Between a Given Base Pair Given a base pair $r_i.r_j$, then this pair can either be the closing base pair of a hairpin loop, the exterior pair of a base

pair stack, the closing pair of a bulge or an interior loop, or close a multi-branched loop. For all of these cases, the corresponding probabilities are given as follows:

$$\begin{aligned}
Q_{ij}^{HL}(i,j) &= \frac{1}{q_{ij}(i,j)} \cdot \beta_L(i+1,j-1) \cdot (\alpha_F(i+1,j-1) \cdot \Pr(L \to F)), \\
Q_{ij}^{SP}(i,j) &= \frac{1}{q_{ij}(i,j)} \cdot \beta_L(i+1,j-1) \cdot (\alpha_P(i+1,j-1) \cdot \Pr(L \to P)), \\
Q_{ij}^{BI}(i,j) &= \frac{1}{q_{ij}(i,j)} \cdot \beta_L(i+1,j-1) \cdot (\alpha_G(i+1,j-1) \cdot \Pr(L \to G)), \\
Q_{ij}^{ML}(i,j) &= \frac{1}{q_{ij}(i,j)} \cdot \beta_L(i+1,j-1) \cdot (\alpha_M(i+1,j-1) \cdot \Pr(L \to M)).
\end{aligned}$$

Here, we have to use the normalizing factor

$$q_{ij}(i,j) = \beta_L(i+1,j-1) \cdot \alpha_L(i+1,j-1).$$

Thus, $Q_{ij}^{HL}(i,j)$, $Q_{ij}^{SP}(i,j)$, $Q_{ij}^{BI}(i,j)$ and $Q_{ij}^{ML}(i,j)$ is the sampling probability for a hairpin loop, base pair stack, bulge or interior loop and multi-branched loop, respectively, where for mutually exclusive and exhaustive cases,

$$Q_{ij}^{HL}(i,j) + Q_{ij}^{SP}(i,j) + Q_{ij}^{BI}(i,j) + Q_{ij}^{ML}(i,j) = 1$$

holds. Note that these sampling probabilities obviously also directly correspond to the ones presented in Section 3.2.4.1 for the PF approach.

Sampling Probabilities for Bulge and Interior Loops For sampling bulge and interior loops in analogy to the PF approach, we would have to use the following probabilities:

$$P_{hl}^{BIL}(i,j,h,l) = \begin{cases} P_{hl}^{B1}(i,j,h), & \text{if } h > i+1 \text{ and } l = j-1, \\ P_{hl}^{B2}(i,j,l), & \text{if } h = i+1 \text{ and } l < j-1, \\ P_{hl}^{IL}(i,j,h), & \text{if } h > i+1 \text{ and } l < j-1, \\ 0, & \text{else,} \end{cases}$$

where

$$\begin{aligned}
P_{hl}^{B1}(i,j,h) &= \frac{1}{Q_{ij}^{BI}(i,j) \cdot q_{ij}(i,j)} \cdot \beta_L(i+1,j-1) \cdot \Pr(L \to G) \\
&\quad \times (\alpha_B(i+1,h-1) \cdot \alpha_A(h,j-1) \cdot \Pr(G \to BA)), \\
P_{hl}^{B2}(i,j,l) &= \frac{1}{Q_{ij}^{BI}(i,j) \cdot q_{ij}(i,j)} \cdot \beta_L(i+1,j-1) \cdot \Pr(L \to G) \\
&\quad \times (\alpha_A(i+1,l) \cdot \alpha_B(l+1,j-1) \cdot \Pr(G \to AB)), \\
P_{hl}^{IL}(i,j,h,l) &= \frac{1}{Q_{ij}^{BI}(i,j) \cdot q_{ij}(i,j)} \cdot \beta_L(i+1,j-1) \cdot \Pr(L \to G) \\
&\quad \times (\alpha_B(i+1,h-1) \cdot \alpha_A(h,l) \cdot \alpha_B(l+1,j-1) \cdot \Pr(G \to BAB)).
\end{aligned}$$

After the case of bulge or interior loop was sampled, $\{P_{hl}^{BIL}(i,j,h,l)\}$ would then be used for sampling h and l (together in one single sampling step) and for mutually exclusive and exhaustive cases, this would imply

$$\sum_{h=(i+1)}^{j-\min_{ps}} \sum_{l=(h-1)+\min_{ps}}^{(j-1)} P_{hl}^{BIL}(i,j,h,l) = 1$$

(under the condition that $Q_{ij}^{BI}(i,j) > 0$).

However, to ensure that the sampling algorithm runs in cubic time, we would then have to disregard long bulge and interior loops by using a constant $\max_{bulge}$ – just like with PFs[6]. Nevertheless, we do *not* need to apply this restriction if we sample h and l one after the other with the following probabilities:

$$
\begin{aligned}
P^{BI}_{hj}(i,j,h) &= \frac{1}{p^{BI}(i,j)} \cdot \beta_G(i+1,j-1) \cdot (\alpha_B(i+1,h-1) \cdot \alpha_A(h,j-1) \cdot \Pr(G \to BA)),\\
P^{BI}_{il}(i,j,l) &= \frac{1}{p^{BI}(i,j)} \cdot \beta_G(i+1,j-1) \cdot (\alpha_A(i+1,l) \cdot \alpha_B(l+1,j-1) \cdot \Pr(G \to AB)),\\
P^{BI}_{hl}(i,j,h) &= \frac{1}{p^{BI}(i,j)} \cdot \beta_G(i+1,j-1) \cdot (\alpha_B(i+1,h-1) \cdot \alpha_{AB}(h-1,j) \cdot \Pr(G \to BAB)),\\
\widehat{P}^{BI}_{hl}(j,h,l) &= \frac{1}{\alpha_{AB}(h-1,j)} \cdot (\alpha_A(h,l) \cdot \alpha_B(l+1,j-1)),
\end{aligned}
$$

where

$$
\alpha_{AB}(i,j) = \sum_{l=i+\min_{ps}}^{(j-2)} (\alpha_A(i+1,l) \cdot \alpha_B(l+1,j-1))
$$

and

$$
p^{BI}(i,j) = \beta_G(i+1,j-1) \cdot \alpha_G(i+1,j-1).
$$

Obviously, $\{P^{BI}_{hj}(i,j,h)\}$ and $\{P^{BI}_{il}(i,j,l)\}$ are the sampling probabilities for bulges on the left and bulges on the right, respectively. Furthermore, $\{P^{BI}_{hl}(i,j,h)\}$ are the probabilities for first sampling h for interior loops and $\{\widehat{P}^{BI}_{hl}(j,h,l)\}$ are the probabilities for sampling l after h is sampled (for interior loops).

Since the probabilities of all mutually exclusive and exhaustive cases sum up to 1, we have

$$
\sum_{h=(i+2)}^{j-\min_{ps}} P^{BI}_{hj}(i,j,h) + \sum_{l=i+\min_{ps}}^{(j-2)} P^{BI}_{il}(i,j,l) + \sum_{h=(i+2)}^{j-\min_{ps}-1} P^{BI}_{hl}(i,j,h) = 1,
$$

and, under the condition that $P^{BI}_{hl}(i,j,h) > 0$, also

$$
\sum_{l=(h-1)+\min_{ps}}^{(j-2)} \widehat{P}^{BI}_{hl}(j,h,l) = 1.
$$

Sampling Probabilities for Multiloops In the case of a multi-branched loop, the probabilities for sampling the first accessible base pair $r_{h_1}.r_{l_1}$ within this loop can be obtained by considering the intermediate symbols of $\mathcal{G}_r$ that generate (parts of) multiloops. More specifically, we first sample h and l according to the following conditional probabilities:

$$
\begin{aligned}
P^{M_1}_{hl}(i,j,h) &= \frac{1}{p^{M_1}(i,j)} \cdot \beta_M(i+1,j-1) \cdot (\alpha^*_U(i+1,h-1) \cdot \alpha_{AO}(h,j) \cdot \Pr(M \to UAO)),\\
\widehat{P}^{M_1}_{hl}(j,h,l) &= \frac{1}{\alpha_{AO}(h,j)} \cdot (\alpha_A(h,l) \cdot \alpha_O(l+1,j-1)),
\end{aligned}
$$

where

$$
\alpha_{AO}(h,j) = \sum_{l=(h-1)+\min_{ps}}^{(j-1)-\min_{ps}} (\alpha_A(h,l) \cdot \alpha_O(l+1,j-1))
$$

[6]Note that when using the PF approach based on thermodynamics, h and l have to be sampled at once, since the free energy of a bulge or interior loops strongly depends on both the closing pair $r_i.r_j$ and the accessible pair $r_h.r_l$ for any $1 \leqslant i \leqslant h < l \leqslant j \leqslant n$.

and

$$p^{M_1}(i,j) = \beta_M(i+1,j-1)\cdot\alpha_M(i+1,j-1).$$

Note that we have to take care of the ϵ-rule $U \to \epsilon$, which implies that symbol U may generate words of size zero. For this reason, $h = i+1$ could be chosen, implying $h-1 < i+1$. However, $\alpha_U(i+1,h-1)$ is only defined for $i+1 \leqslant h-1$. To fix this problem, we have used the term $\alpha^*_U(i+1,h-1)$ instead of $\alpha_U(i+1,h-1)$ in the previous two definitions, which is given as follows:

$$\alpha^*_U(i+1,h-1) = \begin{cases} \alpha_U(i+1,h-1), & \text{if } i+1 \leqslant h-1, \\ \Pr(U\to\epsilon), & \text{if } i+1 > h-1. \end{cases}$$

Thus, $\{\widehat{P}^{M_1}_{hl}(j,h,l)\}$ are probabilities for sampling l after $h \geqslant i+1$ is sampled with probabilities $\{P^{M_1}_{hl}(i,j,h)\}$. For mutually exclusive and exhaustive cases, we have

$$\sum_{h=(i+1)}^{j-2\cdot\min_{ps}} P^{M_1}_{hl}(i,j,h) = 1,$$

and accordingly,

$$\sum_{l=(h-1)+\min_{ps}}^{(j-1)-\min_{ps}} \widehat{P}^{M_1}_{hl}(j,h,l) = 1.$$

Sampling both h and l yields the first accessible base pair $r_{h_1}.r_{l_1} := r_h.r_l$ (which closes the first helix radiating out from this multiloop).

In order to sample the second accessible base pair $r_{h_2}.r_{l_2}$, we consider the remaining structure fragment $R_{(l_1+1),(j-1)}$ (between the $3'$ base r_{l_1} of the first accessible base pair $r_{h_1}.r_{l_1}$ and the $3'$ base r_j of the closing base pair $r_i.r_j$ of the considered multiloop). In fact, for any $k \geqslant 1$, the probabilities for sampling the $(k+1)^{th}$ accessible base pair $r_{h_{k+1}}.r_{l_{k+1}}$ within this multi-branched loop are computed by considering the structure fragment $R_{(l_k+1),(j-1)}$ and using the corresponding inside and outside variables for some specific multiloop generating intermediate symbols of the grammar $\mathcal{G}_r$. More specifically, we first sample h and l according to conditional probabilities, which are defined as follows:

$$P^{M_{k+1}}_{hl}(l_k,j,h) = \frac{1}{p^{M_{k+1}}(l_k,j)}\cdot\beta_X(l_k+1,j-1)\cdot(\alpha^*_U(l_k+1,h-1)\cdot\alpha_{AN}(h,j)\cdot\Pr(X\to UAN)),$$
$$\widehat{P}^{M_{k+1}}_{hl}(j,h,l) = \frac{1}{\alpha_{AN}(h,j)}\cdot(\alpha_A(h,l)\cdot\alpha^*_N(l+1,j-1)),$$

where

$$\alpha_{AN}(h,j) = \sum_{l=(h-1)+\min_{ps}}^{(j-1)} (\alpha_A(h,l)\cdot\alpha^*_N(l+1,j-1))$$

and

$$p^{M_{k+1}}(l_k,j) = \begin{cases} \beta_O(l_k+1,j-1)\cdot\alpha_O(l_k+1,j-1), & \text{if } (k+1)=2, \\ \beta_N(l_k+1,j-1)\cdot\alpha_N(l_k+1,j-1) - & \\ \beta_N(l_k+1,j-1)\cdot(\alpha_U(l+1,j-1)\cdot\Pr(N\to U)), & \text{if } (k+1)\geqslant 3, \end{cases}$$

as well as

$$X = \begin{cases} O, & \text{if } (k+1)=2, \\ N, & \text{if } (k+1)\geqslant 3. \end{cases}$$

Again, we have used α^*_U instead of α_U and α^*_N instead of α_N, which is defined as

$$\alpha^*_N(l+1,j-1) = \begin{cases} \alpha_N(l+1,j-1), & \text{if } l+1 \leqslant j-1, \\ \Pr(N \to U) \cdot \Pr(U \to \epsilon), & \text{if } l+1 > j-1, \end{cases}$$

in order to take care of possible cases where U and/or N generate words of size zero.

According to these definitions, $\{\widehat{P}^{M_{k+1}}_{hl}(j,h,l)\}$ are probabilities for sampling l after $h \geqslant l_k + 1$ is sampled with probabilities $\{P^{M_{k+1}}_{hl}(l_k,j,h)\}$ and again, for mutually exclusive and exhaustive cases, we have

$$\sum_{h=(l_k+1)}^{j-\min_{ps}} P^{M_{k+1}}_{hl}(l_k,j,h) = 1,$$

and

$$\sum_{l=(h-1)+\min_{ps}}^{(j-1)} \widehat{P}^{M_{k+1}}_{hl}(j,h,l) = 1.$$

By sampling both h and l, we obtain the desired $(k+1)^{th}$ accessible base pair $r_{h_{k+1}}.r_{l_{k+1}} := r_h.r_l$ (which closes the $(k+1)^{th}$ helix radiating out from this multiloop).

According to the definition of multi-branched loops, we now have to address two different cases: either the considered multiloop contains no additional accessible base pair, or there is at least one more base pair accessible from the closing pair $r_i.r_j$. These two mutually exclusive cases are addressed by the following two probabilities: Conditional on the sampled values for h_k and l_k (for the k^{th} accessible base pair $r_{h_k}.r_{l_k}$ in the considered multiloop), $k \geqslant 2$, we consider the following "decision" probability for no additional accessible base pairs on the structure fragment $R_{(l_k+1),(j-1)}$ (that is, between the $3'$ base r_{l_k} of the k^{th} accessible base pair $r_{h_k}.r_{l_k}$ and the $3'$ base r_j of the closing base pair $r_i.r_j$):

$$P^{M_{k+1}}_{01}(l_k,j) = \frac{1}{p_{01}(l_k,j)} \cdot \beta_N(l_k+1,j-1) \cdot (\alpha_U(l_k+1,j-1) \cdot \Pr(N \to U)),$$

where

$$p_{01}(l_k,j) = \begin{cases} \beta_N(l_k+1,j-1) \cdot (\alpha_U(l_k+1,j-1) \cdot \Pr(N \to U)), & \text{if } (j-l_k-1) < \min_{ps}, \\ \beta_N(l_k+1,j-1) \cdot \alpha_N(l_k+1,j-1), & \text{if } (j-l_k-1) \geqslant \min_{ps}. \end{cases}$$

Accordingly, the probability that there is at least one more accessible base pair in the considered multiloop (that is, on the structure fragment $R_{(l_k+1),(j-1)}$) is given by $1 - P^{M_{k+1}}_{01}(l_k,j)$.

If no additional accessible base pair is sampled, the sampling process for the considered multi-branched loop (closed by pair $r_i.r_j$) is terminated; the resulting loop is thus a $(k+1)$-loop, with k internal helices closed by the k sampled base pairs $r_{h_p}.r_{l_p}$, $1 \leqslant p \leqslant k$, accessible from the closing pair $r_i.r_j$. Otherwise, the next accessible base pair $r_{h_{k+1}}.r_{l_{k+1}}$ is sampled and afterwards, it has yet again to be decided whether the loop contains additional accessible base pairs or not (by another "decision" sampling). This process is then repeated until no additional base pair is sampled.

6.4.2.2. Formal Description of the Sampling Process

According to the previous discussion, it should be clear that a secondary structure for a given RNA sequence r of length n can be sampled in the following recursive way: Start with the entire RNA sequence $R_{1,n}$ and consecutively compute the adjacent substructures (single-stranded regions and paired substructures) of the exterior loop (from left to right). Any paired

substructure, say the k^{th} substructure of the exterior loops, has to be completed by successively folding nested substructures, that is other loops (here hairpins, stacked pairs, bulges, interior and multi-branched loops), before the $(k+1)^{th}$ adjacent substructure is computed[7].

A schematic overview of the overall sampling process is given in Figure 3.2. Actually, the same strategy can be employed for statistical sampling based on PFs and our SCFG model, but the conditional sampling probabilities utilized for drawing the random choices are derived in different ways. Notably, just like in case of PFs, the sampling proceeds from left to right, which in our case means that any structure is sampled in accordance with the generation of its unique leftmost derivation using the underlying SCFG. For a formal description on how the sampling algorithm works and explicit information on where each of the previously defined sampling probabilities has to be considered in order to perform the needed random choices, see Algorithms 3 to 6.

Algorithm 3 Sampling an entire secondary structure

Input: RNA sequence r of length $n \geqslant 1$, and
all previously defined sampling probabilities computed for r (as global variables).
Output: Secondary structure sec of size n.

procedure computeRandomExteriorLoop (n)
$sec = \emptyset$
Set $i = 1$, $j = n$ and $k = 0$
while $(j - i + 1) \neq 0$ **do**
/*Create $(k+1)^{th}$ helix, starting with free base pair $h.l$, $i < h < l < j$, or leave $R_{i,j}$ unpaired:*/
$extLoopType$ = Sample exterior loop substructure type for $R_{i,j}$
if $extLoopType \mathrel{\widehat{=}} P^E_0(i,j)$ /*case (a): $R_{i,j}$ is single-stranded:*/ **then**
return sec
else if $extLoopType \mathrel{\widehat{=}} P^E_{ij}(i,j)$ /*case (b): $h = i$ and $l = j$:*/ **then**
Set $h = i$ and $l = j$
else if $extLoopType \mathrel{\widehat{=}} \sum_{h=(i+1)}^{(j+1)-\min_{ps}} P^E_{hj}(i,j,h)$ /*case (c): $i < h < l = j$:*/ **then**
Sample $h \in [(i+1), (j+1) - \min_{ps}]$ according to probabilities $\{P^E_{hj}(i,j,h)\}$
Set $l = j$
else if $extLoopType \mathrel{\widehat{=}} \sum_{l=(i-1)+\min_{ps}}^{(j-1)} P^E_{il}(i,j,l)$ /*case (d): $i = h < l < j$:*/ **then**
Set $h = i$
Sample $l \in [(i-1) + \min_{ps}, (j-1)]$ according to probabilities $\{P^E_{il}(i,j,l)\}$
else if $extLoopType \mathrel{\widehat{=}} \sum_{h=(i+1)}^{j-\min_{ps}} P^E_{hl}(i,j,h)$ /*case (e): $i < h < l < j$:*/ **then**
Sample $h \in [(i+1), j - \min_{ps}]$ according to probabilities $\{P^E_{hl}(i,j,h)\}$
Sample $l \in [(h-1) + \min_{ps}, (j-1)]$ according to probabilities $\{\widehat{P}^E_{hl}(j,h,l)\}$
end if
/*Collect base pairs for $(k+1)^{th}$ substructure and add them to the entire structure:*/
$sub = \{h.l, (h+1).(l-1), \ldots, (h + (\min_{hel} - 1)).(l - (\min_{hel} - 1))\}$
$sub = sub \cup$ computeRandomLoop $(h + (\min_{hel} - 1), l - (\min_{hel} - 1))$
$sec = sec \cup sub$
/*Consider the remaining fragment $R_{(l+1),j}$:*/
Set $i = l + 1$ /*=next unpaired base after free base pair $h.l$*/ and $k = k + 1$
end while
return sec
end procedure

[7]This means that the folding process performed by the sampling algorithm corresponds to the native folding procedures of RNA molecules (from left to right, due to the aspects of co-transcriptional folding).

Algorithm 4 Sampling any substructure of an entire secondary structure

procedure computeRandomLoop (i, j)
Set $sec = \emptyset$
$randLoopType$ = Sample loop type closed by i.j
if $randLoopType \mathrel{\widehat{=}} Q_{ij}^{HL}(i,j)$ /*i.j closes hairpin loop:*/ **then**
  return sec
else if $randLoopType \mathrel{\widehat{=}} Q_{ij}^{SP}(i,j)$ /*i.j closes stacked pair:*/ **then**
  $sec = sec \cup \{(i+1).(j-1)\}$
  $sec = sec \cup$ computeRandomLoop $(i+1, j-1)$
else if $randLoopType \mathrel{\widehat{=}} Q_{ij}^{BI}(i,j)$ /*i.j closes bulge or interior loop:*/ **then**
  $sec = sec \cup$ computeRandomBulgeInteriorLoop (i, j)
else if $randLoopType \mathrel{\widehat{=}} Q_{ij}^{ML}(i,j)$ /*i.j closes multiloop:*/ **then**
  $sec = sec \cup$ computeRandomMultiLoop (i, j)
end if
return sec
end procedure

Algorithm 5 Sampling a bulge or interior loop within a secondary structure

procedure computeRandomBulgeInteriorLoop (i, j)
if Sample strictly corresponding to PF approach **then**
  /*This requires to use a constant max_{bulge}:*/
  Sample h and l according to probabilities $\{P_{hl}^{BIL}(i,j,h,l)\}$
else
  /*This allows $max_{bulge} = \infty$ (then no restrictions are applied):*/
  $loopType$ = Sample bulge or interior loop type for $R_{i,j}$
  if $loopType \mathrel{\widehat{=}} \sum_{h=(i+2)}^{j-min_{ps}} P_{hj}^{BI}(i,j,h)$ /*bulge on the left:*/ **then**
    Sample $h \in [(i+2), j-min_{ps}]$ according to probabilities $\{P_{hj}^{BI}(i,j,h)\}$
    Set $l = j$
  else if $loopType \mathrel{\widehat{=}} \sum_{l=i+min_{ps}}^{(j-2)} P_{il}^{BI}(i,j,l)$ /*bulge on the right:*/ **then**
    Set $h = i$
    Sample $l \in [i+min_{ps}, (j-2)]$ according to probabilities $\{P_{il}^{BI}(i,j,l)\}$
  else if $loopType \mathrel{\widehat{=}} \sum_{h=(i+2)}^{j-min_{ps}-1} P_{hl}^{BI}(i,j,h)$ /*interior loop:*/ **then**
    Sample $h \in [(i+2), j-min_{ps}-1]$ according to probabilities $\{P_{hl}^{BI}(i,j,h)\}$
    Sample $l \in [(h-1)+min_{ps}, (j-2)]$ according to probabilities $\{\widehat{P}_{hl}^{BI}(j,h,l)\}$
  end if
end if
$sec = \{h.l, (h+1).(l-1), \ldots, (h+(min_{hel}-1)).(l-(min_{hel}-1))\}$
$sec = sec \cup$ computeRandomLoop $(h+(min_{hel}-1), l-(min_{hel}-1))$
return sec
end procedure

Algorithm 6 Sampling a multiloop within a secondary structure

procedure computeRandomMultiLoop (i, j)
Set $sec = \emptyset$, $k = 0$ and $l_k = i$
while $(j - l_k - 1) \geqslant \min_{ps}$ **do**
/*Create $(k+1)^{th}$ helix, starting with accessible pair $h_{k+1}.l_{k+1}$, $l_k < h_{k+1} < l_{k+1} < j$:*/
if $(k+1) = 1$ **then**
Sample $h \in [(i+1), j - 2 \cdot \min_{ps}]$ according to probabilities $\{P_{hl}^{M_1}(i,j,h)\}$
Sample $l \in [(h-1) + \min_{ps}, (j-1) - \min_{ps}]$ according to probabilities $\{\widehat{P}_{hl}^{M_1}(j,h,l)\}$
Set $h_1 = h$ and $l_1 = l$
else
Sample $h \in [(i+1), j - \min_{ps}]$ according to probabilities $\{P_{hl}^{M_{k+1}}(i,j,h)\}$
Sample $l \in [(h-1) + \min_{ps}, (j-1)]$ according to probabilities $\{\widehat{P}_{hl}^{M_{k+1}}(j,h,l)\}$
Set $h_{k+1} = h$ and $l_{k+1} = l$
end if
/*Collect base pairs for $(k+1)^{th}$ substructure and add them to the entire structure:*/
$sub = \{h.l, (h+1).(l-1), \ldots, (h + (\min_{hel} - 1)).(l - (\min_{hel} - 1))\}$
$sub = sub \cup$ computeRandomLoop $(h + (\min_{hel} - 1), l - (\min_{hel} - 1))$
$sec = sec \cup sub$
/*Decide whether to leave the remaining fragment $R_{(l_{k+1}+1),(j-1)}$ unpaired or not:*/
if $(k+1) \geqslant 2$ **then**
Sample "decision" according to $P_{01}^{M_{k+1}}(l_{k+1}, j)$ and $1 - P_{01}^{M_{k+1}}(l_{k+1}, j)$
if $P_{01}^{M_{k+1}}(l_{k+1}, j)$ /*no additional base pairs on $R_{(l_{k+1}+1),(j-1)}$:*/ **then**
return sec
else
Set $k = k + 1$
end if
end if
end while
return sec
end procedure

Note that this stochastic traceback step for sampling a secondary structure in proportion to a particular distribution for some input sequence r is similar to the traceback algorithm employed in MFE based DP algorithms. Actually, the main difference is that in those algorithms, base pairings are selected by the minimum energy principle for the fragments $R_{i,j}$, whereas here, base pairs are randomly sampled according to conditional probability distributions for the corresponding fragments, defined by the precomputed inside and outside probabilities and the probabilities of the grammar rules (in contrast to PF approaches where conditional sampling probabilities are derived from precomputed equilibrium PFs and energy values, see Section 3.2.4).

It remains to mention that when the probabilities $\alpha_x(i,j)$, $x \in \{AT, AB, AO, AN\}$, $1 \leqslant i, j \leqslant n$, are also precomputed, each of the needed sampling probabilities can be derived in constant time. Thus, after a preprocessing of the given RNA sequence (which includes the complete inside outside computation and takes cubic time and requires quadratic storage), corresponding secondary structures can be quickly generated. In fact, the time complexity of the sampling algorithm is bounded by $\mathcal{O}(n^2)$, since any structure of size n can have at most $\lfloor \frac{n - \min_{HL}}{2} \rfloor$ base pairs and any base pair can be sampled in linear time.

We can hence conclude that the considered SCFG based approach and the corresponding PF variant can produce a statistical sample for a given input sequence with similar time and space requirements, but the SCFG method can be used with less restrictions: one can allow $\min_{HL} < 3$, non-canonical base pairs and bulge/interior loops of arbitrary length, due to

the departure from thermodynamic models. However, when comparing the results of both sampling strategies, significant differences can be observed, as we will see in Section 6.6.

6.5. Extension to Structure Prediction

The sampling algorithm described in the last sections can easily be extended to a prediction algorithm for RNA secondary structures of a single sequence. In principle, after a sample set of possible secondary structures for a given RNA sequence has been constructed, we can derive a corresponding prediction from those (more or less) different candidate structures. Obviously, we can either pick one particular structure from the generated sample as prediction (according to a preliminary defined selection procedure) or we can compute a new structure as predicted folding (according to a preliminary defined construction scheme), where the predicted structure itself must not necessarily be contained in the considered sample.

Notably, for the latter variant, there exist elegant ways to incorporate a trade-off parameter γ_{t-o} in order to provide the user with a mechanism for controlling the *sensitivity* (Sens.) and the *positive predictive value* (PPV)[8] of the predicted foldings. These two measures were introduced in order to quantify the accuracy of RNA secondary structure prediction methods and are usually defined as follows (see, for instance [BBC+00]):

- Sens. is the relative frequency of correctly predicted pairs among all position pairs that are actually paired in a stem of native foldings, whereas
- PPV is defined as the relative frequency of correctly predicted pairs among all position pairs that were predicted to be paired with each other.

Formally, they are given by

$$\text{Sens.} = \frac{\mathsf{TP}}{\mathsf{TP} + \mathsf{FN}} \quad \text{and} \quad \text{PPV} = \frac{\mathsf{TP}}{\mathsf{TP} + \mathsf{FP}},$$

where

- TP is the number of correctly predicted base pairs (*true positives*),
- FN is the number of base pairs in the native structure that were not predicted (*false negatives*) and
- FP is the number of incorrectly predicted base pairs (*false positives*).

Note that in [DWB06], the idea of a parameter γ_{t-o} to control the sensitivity/PPV tradeoff has been used in connection with a DP optimization scheme. According to its value, the algorithm either tends to predict only those base pairs with rather strong signals for them to belong to the native folding or it is encouraged to predict more pairings even if they might be no part of the native structure. Here, we will show how γ_{t-o} can be incorporated in connection with sampling algorithms.

6.5.1. Most Frequent Structure

Since for a sufficiently large sample size, the generated samples are statistically representative, the most frequently observed structure within a given sample set can be assumed to be equal to the most probable folding for the given input sequence (under the considered model, that is according to the corresponding distribution on the entire ensemble of feasible structures for that sequence). Consequently, for an adequate prediction choice, we simply have to sample a sufficiently large number of possible foldings and choose the most frequently sampled one

[8] Note that the positive predictive value is often called *specificity*, like for example in [DWB06], which will be extensively referenced in the sequel.

as prediction. If there are more than one structures sampled with the same highest observed frequency, then the one with the highest probability among all of them should be chosen. This can be considered the standard selection method, as it intuitively yields the "best" sampled structure, which will be denoted by *most frequent (MF)* structure in the sequel.

Obviously, this selection inevitably corresponds to and is thus effectively comparable to the outputs of conventional SCFG based prediction methods for RNA secondary structures from a single sequence. In fact, these methods traditionally determine the most likely parse tree for a given input sequence (under the considered stochastic model) and for structurally unambiguous SCFGs, the most likely parse tree is actually equal to the most probable secondary structure for the given sequence (see Section 3.3.7.1).

6.5.2. Maximum Expected Accuracy Structures

For so-called *maximum expected accuracy (MEA)* structures as discussed in Section 3.3.7.3, we employ a rather simple procedure for constructing a particular prediction from a given sample set. As proposed in [DWB06], we will use a trade-off parameter γ_{t-o} for controlling the sensitivity and PPV of predicted foldings. Briefly, the MEA structures for a given sequence are then defined as the ones among all candidate structures that maximize the number of correctly unpaired positions plus γ_{t-o} times the number of correctly paired positions with respect to the true folding of that sequence.

In our case, γ_{t-o} may take on any positive real value and the choice of $\gamma_{t-o} = 1$ serves as the neutral element with respect to the prediction, that is the prediction is neither biased towards a better sensitivity nor to a better PPV. More precisely, γ_{t-o} may take on values in $[0, \infty)$, where for the considered sequence fragment $R_{i,j}$, $1 \leqslant i, j \leqslant n$,

- $\gamma_{t-o} < 1$ restricts the procedure to produce pair $i.j$ only if it is extremely confident,
- $\gamma_{t-o} = 1$ has no impact on the decision whether $i.j$ should be paired or not,
- $\gamma_{t-o} > 1$ encourages the algorithm to produce pair $i.j$, even if it is not confident,

that this pair belongs to the native folding.

In [DWB06], this parameter is actually used in the DP algorithm for computing the predicted folding – the MEA structure. More precisely, the corresponding DP matrix M is computed according to the following recurrence:

$$M_{i,i} = q_i, \text{ for } 1 \leqslant i \leqslant n, \text{ and}$$

$$M_{i,j} = \max \begin{cases} q_i + M_{i+1,j}, \\ M_{i,j-1} + q_j, \\ \gamma_{t-o} \cdot 2 \cdot p_{i,j} + M_{i+1,j-1}, \\ \max\limits_{i<k<j-1} M_{i,k} + M_{k+1,j}, \end{cases} \quad \text{for } 1 \leqslant i \leqslant j-1 \text{ and } 1 \leqslant j \leqslant n,$$

where $p_{i,j}$ denotes the probability that i pairs with j and q_i denotes the probability that i remains unpaired. The traceback step of the corresponding DP algorithm can thus be employed to identify the MEA structures of the input sequence according to the given setting of γ_{t-o}. If only one MEA structure is recovered in the traceback step, the complete algorithm obviously requires $\mathcal{O}(n^3)$ time and $\mathcal{O}(n^2)$ space.

Note that the authors of [DWB06] claim that for the default setting $\gamma_{t-o} = 1$, the algorithm only maximizes the expected number of correct (unpaired and paired) positions and is actually identical to the DP technique used in `Pfold`. This, however, is not quite right, since in the unparameterized recurrence of `Pfold` as introduced in [KH03] (supplemental material) and discussed in Section 3.3.7.3, the pairing probability $p_{i,j}$ is actually not multiplied by

the factor 2. Consequently, if this factor is indeed considered, the default setting would be $\gamma_{t-o} = \frac{1}{2}$. Therefore, we decided to ignore this multiplicative factor 2 in the above presented parameterized recurrence.

Anyway, according to [KH03] (supplemental material), a corresponding MEA parser for our sophisticated SCFG and $m_s = 1$ could actually precompute the pairing probabilities $p_{i,j}$, $1 \leqslant i, j \leqslant n$, based on the formula[9]

$$\begin{aligned} p_{i,j} = \; & \beta_A(i,j) \cdot \Pr(A \to ({}^{m_s}L)^{m_s}) \cdot \alpha_L(i+m_s, j-m_s) + \\ & \beta_P(i,j) \cdot \Pr(P \to (L)) \cdot \alpha_L(i+1, j-1), \end{aligned} \tag{6.1}$$

as $A \to ({}^{m_s}L)^{m_s}$ and $P \to (L)$ are the only rules that can create paired bases at positions i and j, see Section 3.3.7.3 for details. The respective q_i values, $1 \leqslant i \leqslant n$, can then immediately be derived according to

$$q_i = 1 - \sum_{j \neq i} p_{i,j}.$$

However, contrary to the `Pfold` [KH03] and `CONTRAfold` [DWB06] programs, we don't need to derive the $p_{i,j}$'s and q_i's explicitly from our grammar and therefore it is not necessary to find a corresponding formula valid for any possible choice of m_s. In our case, we can easily deduce the needed probabilities from the sample set. This way, we consider the distribution implied by the sample instead of the distribution of the entire structure ensemble of the given input sequence. For a representative sample set, however, this will make no difference.

Accordingly, we will compute the probabilities $p_{i,j}$ by counting the frequencies of all observed base pairs in the particular sample set generated by the sampling algorithm instead of considering the corresponding inside outside values and grammar parameters as done in formula (6.1). Consequently, the sole difference of our way to compute a MEA structure compared to the `CONTRAfold` approach lies in the precomputation step, where we will calculate the pairing probabilities according to

$$p_{i,j} = \frac{\text{Number of occurrences of pair } i.j \text{ within sample}}{\text{Sample size}},$$

such that they depend only on the sampled structures rather than on the entire structure ensemble for the given input sequence.

Note that in a clever implementation, $p_{i,j}$ can be determined while constructing the sample. That way, our approach gets rid of the computational overhead needed in cases where (6.1) is used. Note further that we can make use of this idea in connection with arbitrary sample sets, especially those generated by a PF based approach. MEA structures derived from particular sample sets of candidate foldings for a given setting of the sensitivity/PPV trade-off parameter γ_{t-o} will be called γ_{t-o}-*MEA structures* (of the respective sample) in the sequel.

6.5.3. Centroid Structures

The previously proposed selection procedures are especially adequate if one attempts to compare the results to that of other probabilistic prediction methods like the one employed for lightweight SCFGs in [DE04] or those implemented in `Pfold` and `CONTRAfold`. This is due to the fact that for a given input sequence, all these algorithms propose only one folding (the one that is assumed to be the "best" under the corresponding model, that is the most likely or the MEA structure for the sequence) instead of producing a statistically representative set of candidate structures.

[9] It should be clear that this formula would only be correct for $m_s = 1$. For choices of $m_s > 1$, however, it would only yield approximate results.

Nevertheless, one benefit of taking on a sampling approach that draws a number of possible foldings from the considered structure ensemble is that we can easily consider alternative schemes for constructing corresponding predictions. Particularly, we can make use of the fact that many (more or less) different secondary structures have been generated by the repeated execution of the sampling procedure and compute a suitable single prediction from the entire sample set. This can be done for example by constructing a particular *consensus* structure like the *centroid* [DCL05] structure of the sample which can be considered as the single structure that best represents the central tendency of the generated sample set.

As the centroid reflects the overall behavior of the structures in the sample, this choice possibly represents an appropriate alternative to the best sampled structure, that is the most probable structure according to the considered ensemble distribution implied by the used probabilistic or energy-based[10] approach. Therefore, computing centroids has become custom for applying sampling approaches to single sequence structure prediction. In analogy to the `Sfold` software, we could derive both the *ensemble centroid* (that is, the centroid computed from the entire set of sampled structures) and the *cluster centroid* (that is, the centroid derived only from a subset of structurally similar samples). However, in order to have a single prediction to be compared to the native folding resp. to the output of other tools, we decided to make only use of the first. Furthermore, the ensemble centroid characterizes the central tendency of the entire (representative) sample set and thus is the right choice for what we have in mind, namely for studying the distribution implied by our SCFG.

Formally, the centroid for a given sample set is the structure in the entire structure ensemble that has the minimum total base pair distance to the structures in the set. It can efficiently be computed as the unique consensus structure formed by all base pairs with a frequency of more than 50% within the sample, where the essential matter of fact is that any two base pairs with frequencies $> 50\%$ can not form a pseudoknot. For details, we refer to [DCL05].

In accordance with γ_{t-o}-MEA structures as defined above, we now introduce centroid structures (constructed from sample sets of secondary structures) according to particular settings of the sensitivity/PPV trade-off parameter γ_{t-o}, which will be named γ_{t-o}-*centroids* (of the respective sample) in the sequel. Note that this generalized version of the centroid is very similar to the concept of γ_{t-o}-*centroid estimators* proposed in [HKS+09], which predict the secondary structure maximizing the expected weighted true predictions of base pairs in the predicted structure on the basis of a particular ensemble distribution for a given RNA sequence (such as for example the Boltzmann distribution or the one implied by a considered SCFG model). In fact, both versions are equivalent to the unique centroid proposed in [DCL05] for $\gamma_{t-o} = 1$, but the one introduced in [HKS+09] determines the structure that optimizes the *expected numbers* of base pairs of TP, TN[11], FP and FN with respect to the entire ensemble distribution, whereas we only consider the generated sample set for deriving a corresponding γ_{t-o}-centroid.

Formally, a γ_{t-o}-centroid for a given set of m structures that all have length n is calculated by determining all base pairs $i.j$, $1 \leqslant i, j \leqslant n$, which satisfy

$$c_{i,j} = (\text{Number of occurrences of pair } i.j \text{ within sample}) \cdot \gamma_{t-o} > \frac{m}{2}. \tag{6.2}$$

These pairs are then used for constructing a corresponding consensus structure, where we have to take care of the fact that the inclusion of any of these pairs into the consensus could eventually result in a pseudoknot or a base triplet which are both prohibited according to our definition of RNA secondary structure. Therefore, we define the γ_{t-o}-centroid as the consensus structure that is formed by successively including base pairs $i.j$ with $c_{i,j} > m/2$ according to their observed frequencies in the sample set (in decreasing order), where $i.j$ is included if and

[10] Note that the most probable structure is assumed to be (nearly) the MFE structure when sampling is realized via PFs.

[11] TN is the number of base pairs which were correctly predicted as non-matching (*true negatives*).

only if it yields a compatible combination (that is, it causes neither a pseudoknot nor a base triplet in the partially formed consensus).

Remark 6.5.1 *An alternative interpretation of the centroid estimator as introduced in [HKS+09] is the following: The predicted secondary structure maximizes the sum of base pairing probabilities larger than*

$$\frac{1}{\gamma_{t-o}+1}.$$

According to equation (6.2) and the strategy just described, this is quite similar to our prediction, since $c_{i,j} > m/2$ *can be rewritten as*

$$\widehat{c}_{i,j} = \frac{\textit{(Number of occurrences of pair } i.j \textit{ within sample)}}{m} > \frac{1}{2 \cdot \gamma_{t-o}},$$

where $\widehat{c}_{i,j}$ *corresponds to a base pairing probability. By choosing the base pairs according to their decreasing observed frequencies, then to construct a* γ_{t-o}*-centroid our strategy aims for maximizing the sum of the*

$$\widehat{c}_{i,j} > \frac{1}{2 \cdot \gamma_{t-o}}.$$

Anyway, the time complexity for computing one possible γ_{t-o}-centroid is bounded by $\mathcal{O}(n^3)$, since any (partially formed) structure of size n can have $\mathcal{O}(n)$ base pairs and we potentially have to check for any of the $\mathcal{O}(n^2)$ possible base pairs whether it can be added to the partially formed centroid or not (that is, whether it yields a compatible or incompatible combination).

It should be noted that contrary to γ_{t-o}-MEA structures, where reasonable values are

$$\gamma_{t-o} \in [0, \infty),$$

γ_{t-o}-centroids by definition might only yield meaningful predictions for

$$\gamma_{t-o} \in \left(\frac{1}{2}, \frac{m}{2}\right).$$

Particularly,

- $\gamma_{t-o} \leqslant 1/2$ leads to $c_{i,j} \leqslant m \cdot \gamma_{t-o} \leqslant m/2$, for any $1 \leqslant i, j \leqslant n$, such that the corresponding centroid contains no base pairs at all,
- $1/2 < \gamma_{t-o} < 1$ results in a unique centroid formed by pairs that have been sampled very often,
- $\gamma_{t-o} = 1$ produces the unique centroid structure formed by all pairs with a frequency greater than 50%,
- $1 < \gamma_{t-o} < m/2$ might produce distinct centroids containing even such pairs that have rarely been sampled,
- $\gamma_{t-o} > m/2$ implies $c_{i,j} \geqslant 1 \cdot \gamma_{t-o} > m/2$ for any pair $i.j$ occurring in the sample, such that the centroid might entirely consist of pairs which have been sampled only once.

However, just like the MF structure, both the γ_{t-o}-MEA and γ_{t-o}-centroid structures can be calculated from any given set of secondary structures. This means they can not only be employed for obtaining predictions from samples generated with a (sophisticated) SCFG approach, but also from sets of possible foldings created with a corresponding statistical sampling strategy based on PFs. Consequently, this allows for a direct and well-defined comparison of the produced samples with respect to prediction accuracy.

Finally, it might be important to mention that with any of the previously proposed distinct selection processes, the predicted structure can be recovered in $\mathcal{O}(n^3)$ time and with $\mathcal{O}(n^2)$ space

requirements, such that the worst-case complexities of the corresponding overall prediction algorithms are equal to those of the respective sampling procedures.

6.6. Evaluation and Discussion

The main objective of this section is to find answers to the two questions from Section 6.1, and especially to prove or disprove hypothesis H_0. To reach this goal, we will compare our sophisticated SCFG to lightweight SCFGs and corresponding CLLMs for RNA structure prediction. Furthermore, we will discuss the potentials and pitfalls of the corresponding SCFG based sampling method and compare it to the sampling extension of the PF approach as implemented in the `Sfold` software.

Note that the (purposive) implementation of the statistical sampling strategy sketched in Section 6.4 (including the corresponding routines for extracting structure predictions as described in Section 6.5) used for deriving the results of this chapter has been incorporated into a web service, which is accessible to the scientific community at `http://wwwagak.cs.uni-kl.de/ProbStatSample`.

6.6.1. Comparison to Lightweight Grammars and Leading Prediction Methods

In order to see if our sophisticated SCFG can close the performance gap between probabilistic and MFE based approaches and furthermore whether its rich structure and parameter set allows to compensate the powerful scoring schemes of CLLMs (outperforming leading prediction methods) derived from lightweight grammars, we decided to perform a series of cross-validation experiments. Actually, we will compare our grammar to the nine different lightweight SCFGs proposed in [DE04] (to see if its sophisticated design is of any advantage), as well as to the corresponding nine CLLMs and a number of leading prediction methods such as `Mfold` or `ViennaRNA` considered in the `CONTRAfold` paper [DWB06].

It should be mentioned that the nine lightweight SCFGs from [DE04] can be categorized into three groups:

- First, two structurally ambiguous grammars: G1 is the most simple one (only 5 rules with same left-hand side) and G2 extends it to include base pair stacking parameters.
- Second, four unambiguous ones: G3 (with 3 intermediates and a total of 8 rules), the smaller G4 (with 2 intermediates and 6 rules), the ultra compact G5 (only one intermediate symbol with 3 alternatives) and G6 (the one utilized in `Pfold`, with 3 intermediates and 6 rules), where each grammar describes a slightly different class of structures (mainly according to different minimum allowed hairpin lengths).
- Third, three unambiguous grammars capable of including stacking parameters (and thus prohibiting isolated base pairs): G6s (extension of G6), as well as G7 and G8 (more complex versions of the simple backbones G3 and G4).

Generally, G1 and G5 perform badly, which might be due to the presence of only one non-terminal symbol. Notably, G5 is an extremely bad choice for RNA secondary structure, but a (very) good choice for CMs[12]. The reason, overloading of symbols, leads to this behavior, as for CMs one extends the grammar (by adding rules modeling insertions, deletions and matches), thereby removing the overloading problem (see G5M in [GzS11] for a corresponding specialization of G5).

[12]That is, covariance models, as mentioned in Section 3.3.8.

Sampling Parameters	MF struct.	
	Sens.	PPV
$\min_{HL} = 1, \min_{hel} = 1$	0.4433	0.5447
$\min_{HL} = 1, \min_{hel} = 2$	0.4895	0.5551
$\min_{HL} = 3, \min_{hel} = 1$	0.4852	0.5948
$\min_{HL} = 3, \min_{hel} = 2$	0.5171	0.5661

(a) Sensitivity and PPV derived by applying the SCFG based statistical sampling algorithm and choosing the most frequently sampled structure as predicted folding. Notably, all results were computed by two-fold cross-validation procedures, using the same folds of the S-151Rfam database as in [DWB06] and a sample size of 1000 structures.

Grammar	Generative Viterbi		Discriminative Viterbi	
	Sens.	PPV	Sens.	PPV
G1	0.41	0.27	0.40	0.28
G2	0.53	0.36	0.63	0.48
G3	0.46	0.48	0.45	0.46
G4	0.21	0.17	0.21	0.17
G5	0.03	0.04	0.02	0.03
G6	0.60	0.61	0.61	0.62
G6s	0.60	0.62	0.62	0.63
G7	0.58	0.63	0.58	0.62
G8	0.58	0.60	0.58	0.61

(b) Corresponding results from [DWB06].

Table 6.1.: **Comparison of Viterbi prediction accuracies.** Results were obtained by computing the most likely secondary structure for a given sequence by distinct approaches.

Nevertheless, since in [DWB06], for each of the nine original lightweight SCFGs from [DE04], an equivalent CLLM has been constructed and two-fold cross-validation procedures[13] have been applied to compare the performances of the respective SCFG and CLLM, we decided to consider the same partition of the structural data set collected in [DWB06] into two folds, such that results reported there can be easily opposed to corresponding ones obtained by our sampling method. Note that this data set contains 151 independent examples of known secondary structures of non-coding RNA from the Rfam database [GJBM+03, GJMM+05], where each independent example has been taken from a different RNA family. It will be denoted by *S-151Rfam database* in the sequel.

For adequate comparisons in case of the lightweight SCFGs and CLLMs, we only considered those principles to derive a prediction from our sample and only corresponding values of γ_{t-o} for which corresponding results are given in [DWB06]. Accordingly, for every structure used for evaluation, we generated a set of 1000 candidate structures[14] with the sampling algorithm and afterwards computed the corresponding MF structure and γ_{t-o}-MEA structures, respectively. These predicted foldings were then opposed to the native secondary structure of the molecule (as given in the database) in order to calculate the corresponding sensitivity and PPV, respectively. The corresponding cross-validation results for the mixed S-151Rfam database are listed in Tables 6.1 and 6.2.

As we can see from Table 6.1a, the MF structure predictions obtained by sampling on the basis of our sophisticated SCFG become more accurate when considering the realistic value of $\min_{HL} = 3$. Nevertheless, comparing all results from Table 6.1 yields the observation that our sophisticated SCFG does not generally outperform any lightweight SCFG and corresponding CLLM, as the most elaborate (generatively or discriminatively trained) grammars G6 to G8 seem to have a greater predictive power when considering the most likely folding of a given input sequence. This might be caused by the fact that the SCFG design underlying the sampling

[13] In order to perform a k-fold cross-validation, $k \geqslant 2$, on the basis of a given probabilistic model and a set of real-world data, we first have to partition the data randomly into k approximately equal-sized subsets ("folds"). Then, for any $i \in \{1, \ldots, k\}$, we must estimate the model parameters from all objects that are *not* contained in fold i (training set) and validate the results obtained for all objects that actually belong to fold i (benchmark set). The corresponding result of the cross-validation process is then the average of the results derived for the different folds i, $1 \leqslant i \leqslant k$.

[14] This sample size has proven to be adequate for most applications, as even for a huge set of possible secondary structures of a given sequence, a sample of only 1000 structures can yield statistical reproducibility of typical sampling statistics, even if samples can be entirely different (see [DL03]).

Sampling Parameters	MEA struct.		
	Sens.	PPV	γ_{t-o}
$min_{HL} = 1, min_{hel} = 1$	0.6029	0.6192	4.0
$min_{HL} = 1, min_{hel} = 2$	0.6325	0.5896	4.0
$min_{HL} = 3, min_{hel} = 1$	0.6090	0.6230	4.0
$min_{HL} = 3, min_{hel} = 2$	0.6311	0.5867	4.0

(a) Sensitivity and PPV derived by applying the SCFG based statistical sampling algorithm and choosing a particular γ_{t-o}-MEA structure as predicted folding. Notably, all results were computed by two-fold cross-validation procedures, using the same folds of the S-151Rfam database as in [DWB06] and a sample size of 1000 structures.

Grammar	Generative MEA		Discriminative MEA	
	Sens.	PPV	Sens.	PPV
G1	0.18	0.11	0.48	0.33
G2	0.53	0.36	0.67	0.64
G3	0.56	0.51	0.54	0.53
G4	0.33	0.23	0.34	0.23
G5	0.06	0.04	0.06	0.04
G6	0.62	0.63	0.62	0.67
G6s	0.62	0.64	0.65	0.65
G7	0.63	0.63	0.63	0.67
G8	0.63	0.62	0.65	0.62

(b) Corresponding results from [DWB06].

Table 6.2.: **Comparison of MEA prediction accuracies.** Results were obtained by determining a single MEA structure for each given sequence, where the MEA parsing methods are based on the indicated models and γ_{t-o} was adjusted to allow a direct comparison.

algorithm is too comprehensive to allow for a reliable parameter estimation with respect to the rather sparse but diverse mixed S-151Rfam data set.

Table 6.2 however indicates that when constructing particular γ_{t-o}-MEA structures of generated samples, the corresponding prediction results are not significantly less accurate than those obtained by the considered MEA parsing algorithms based on (generatively or discriminatively trained) lightweight grammars. Moreover, there seems to be a slight trade-off between the sensitivity and PPV of the predicted foldings when applying the sophisticated SCFG sampling approach with different values of min_{hel}, that is when either allowing or prohibiting isolated base pairs (see Table 6.2a).

For an even more informative comparison of the predictive powers of the distinct lightweight grammar parsing techniques and the sophisticated SCFG based sampling method, the performance has also be measured at several different settings of the γ_{t-o} parameter. In fact, by determining the (adjusted) sensitivity and PPV for various values of γ_{t-o}, we are able to derive corresponding *receiver operating characteristic (ROC)* curves for the γ_{t-o}-MEA prediction selecting principle (according to the different parameter combinations considered for statistical sampling). Here, we decided to consider any value of

$$\gamma_{t-o} \in \{1.25^k \mid -12 \leqslant k \leqslant -1\} \cup \{2^k \mid 0 \leqslant k \leqslant 12\}$$

in order to obtain appropriate ROC curves. For each curve, the estimated *area under the curve (AUC)* is reported in Table 6.3a.

Comparing these results to the corresponding values obtained on the basis of the considered lightweight models (Table 6.3b), we immediately observe that for 5 out of 9 generatively-trained grammars and 4 out of 9 discriminatively-trained grammars, the probabilistic sampling approach (and thus the underlying sophisticated SCFG) yields significantly better results. In all other cases, the sampling variant performs worse, but the corresponding results actually bare no substantial differences with respect to the observed prediction quality.

For a comparison of the predictive accuracy of our sophisticated SCFG sampling approach to several leading probabilistic and physics-based prediction methods, we again considered the S-151Rfam database together with our various strategies to derive a prediction from our samples. The observed sensitivity and PPV measures are collected in Table 6.4.

Sampling Parameters	MEA struct.
$\min_{HL} = 1, \min_{hel} = 1$	0.499491
$\min_{HL} = 1, \min_{hel} = 2$	0.506602
$\min_{HL} = 3, \min_{hel} = 1$	0.507454
$\min_{HL} = 3, \min_{hel} = 2$	0.508762

(a) Estimated AUCs, where the corresponding ROC curves are found by computing one γ_{t-o}-MEA structure for any considered setting of γ_{t-o} from a particular statistical sample generated by the SCFG based algorithm. Notably, all results were computed by two-fold cross-validation procedures, using the same folds of the S-151Rfam database as in [DWB06] and a sample size of 1000 structures.

Grammar	Generative MEA	Discriminative MEA
G1	0.0392	0.2713
G2	0.3640	0.5797
G3	0.4190	0.4159
G4	0.1361	0.1350
G5	0.0026	0.0031
G6	0.5446	0.5600
G6s	0.5501	0.5642
G7	0.5456	0.5582
G8	0.5464	0.5515

(b) Corresponding results from [DWB06].

Table 6.3.: **Comparison of prediction accuracies by means of areas under ROC curves.** Values were found by determining a set of MEA structures (for reliable choices of parameter γ_{t-o}) for each given sequence, where the MEA parsing methods are based on distinct models.

Sampling Parameters	MF struct.		MEA struct.			Centroid struct.		
	Sens.	PPV	Sens.	PPV	γ_{t-o}	Sens.	PPV	γ_{t-o}
$\min_{HL} = 1, \min_{hel} = 1$	0.4433	0.5447	0.6342	0.5842	6.0	0.6522	0.5612	6.0
			0.5083	0.6808	2.0	0.4387	0.7180	1.5
$\min_{HL} = 1, \min_{hel} = 2$	0.4894	0.5551	0.6546	0.5593	6.0	**0.6624**	0.5354	6.0
			0.4980	0.6850	1.5	0.4801	0.6977	1.5
$\min_{HL} = 3, \min_{hel} = 1$	0.4852	0.5948	0.6348	0.5826	6.0	0.6464	0.5616	6.0
			0.4627	0.7044	1.5	0.4487	**0.7241**	1.5
$\min_{HL} = 3, \min_{hel} = 2$	0.5171	0.5661	0.6502	0.5568	6.0	0.6411	0.5700	4.0
			0.4342	0.7228	1.0	0.4917	0.7103	1.5

(a) Sensitivity and PPV derived by applying the SCFG based statistical sampling algorithm and selecting the predicted folding according to any of the described schemes. Notably, all results were computed by two-fold cross-validation procedures, using the same folds of the S-151Rfam database as in [DWB06] and a sample size of 1000 structures.

Method	References	Sens.	PPV	γ_{t-o}
`CONTRAfold`	[DWB06]	0.7377	0.6686	6.0
`Mfold` v3.2	[Zuk89b, Zuk03]	0.6943	0.6063	–
`ViennaRNA` v1.6	[HFS+94, Hof03]	0.6877	0.5922	–
`PKNOTS` v1.05	[RE99]	0.6030	0.5269	–
`ILM`	[RSZ04]	0.5330	0.4098	–
`CONTRAfold`	[DWB06]	0.5540	0.7920	0.75
`Pfold` v3.2	[KH99, KH03]	0.4906	0.7535	–

(b) Accuracies of other methods, as reported in [DWB06].

Table 6.4.: **Comparison with established tools.** Table opposes the sophisticated SCFG sampling approach to leading secondary structure prediction methods (that are not based on sampling).

First, we observe that accuracies similar to those of `Mfold` and `ViennaRNA` can be reached by our SCFG based sampling method when predicting γ_{t-o}-MEA and γ_{t-o}-centroid structures (for adjusted settings of the trade-off parameter γ_{t-o}, respectively), whereas the worst results are obtained when choosing the MF structure as predicted folding (see Table 6.4a). Furthermore, according to the presented results, our SCFG based sampling approach has been outperformed only by half of the existing probabilistic and energy-based structure prediction methods.

In conclusion, we observe that our sophisticated SCFG cannot significantly improve the predictive power of grammar based methods. Contrarily, the usage of γ_{t-o}-MEA structures as well as γ_{t-o}-centroids (both as introduced in Section 6.5) can improve the quality of predictions derived by a sampling approach. The highest values for sensitivity resp. PPV have been observed for γ_{t-o}-centroids ($\gamma_{t-o} = 6.0$ resp. $\gamma_{t-o} = 1.5$) where we were able to achieve a predictive accuracy close to the one of `Mfold` and `ViennaRNA`.

However, these observations have been made in connection with a mixed and lean database which might be too small to reliably estimate the rich set of parameters of our grammar. Furthermore, as outlined in Section 6.1, it might be possible that a sophisticated grammar design is able to capture structural properties (including aspects which are caused by interaction with proteins or by other *non-energetic* details of RNA folding) typical to a single RNA family by the respective parameter values. This possibility – besides other things – will be investigated in the following section. There, we will compare our sampling method to a corresponding physics based approach since that for sure is incapable of adapting to a certain class since its parameters are assumed fixed.

6.6.2. Comparison of Sample Distributions

Since the considered sampling strategy produces statistically representative sample sets of the complete structure ensemble for a given sequence, we can not only judge the quality of predictions derived from a particular sample, but also the quality of the generated sample as it. In this section, we will compare the sample distribution implied by our sophisticated SCFG to the one induced by the PF based sampling method as implemented in the `Sfold` software. For that purpose, we will consider probability profiles as well as (and most interestingly from the perspective of biologists) a number of different comparisons on the basis of *abstract shapes* as introduced in [JRG08] and discussed throughout Section 2.2.3.

Finally, before we start our examinations, it should be mentioned that in order to derive all results for the particular applications that will follow throughout this section, we have implemented our own version of `Sfold`'s sampling procedure as described in [DL03]. For this implementation, we decided to employ the common thermodynamic parameters from Mathews et al. [MSZT99], which were also utilized for version 3.0 of the `Mfold` software [Zuk03].

6.6.2.1. RNA Data

For the previously mentioned reasons, we decided to no only consider the mixed S-151Rfam database for our subsequent comparisons, but also use several other databases that contain more structures having more similar shapes. In particular, we took the tRNA database from [SHB+98], where we filtered out all sequences with unidentified bases, yielding a total of 2163 distinct tRNA structures (having lengths in $[64, 93]$ and an average length of 76). Additionally, we created another set of 1149 distinct sequences (with lengths in $[102, 135]$ and about 119 on average), retrieved from a 5S rRNA database [SBEB02]. These data sets of tRNA and 5S rRNA structures, along with the mixed S-151Rfam set, will be the basis for the following studies.

6.6.2.2. Probability Profiling for Specific Loop Types

A representative sample of all possible secondary structures for a given RNA sequence can be used to derive estimates for the probability (conditioned to the sequence) of any structural motif to show up at the different sequence positions. For example, *probability profiling* of unpaired bases in RNA secondary structure becomes possible, that is paired and unpaired bases are delineated on statistic grounds derived from the sample set. In detail, for probability profiling, the unpaired bases can either be delineated regardless of the type of loop (like hairpin, bulges and so so) in which they occur. Or, by keeping track of the loop type for unpaired bases, an extension that accounts for the different types of loops is possible. For each nucleotide position i, $1 \leqslant i \leqslant n$, of a given sequence of length n, one computes the probabilities that i is an unpaired base within a specific loop type. These probabilities are given by the observed frequency in a sample set of secondary structures for the given sequence.

For a first comparison of the two different sample distributions, we decided to consider the corresponding probability profiles for the *Escherichia coli* tRNA^{Ala} sequence (as presented in Example 2.1.1). Using a sample size of 1000 structures, we obtain the ten profile plots shown in Figure 6.1 which obviously exhibit the cloverleaf structure of tRNAs.

All these profiles show that the (statistically representative and reproducible) samples generated by the SCFG approach are significantly more accurate than those obtained with the PF approach. Moreover, considering the results for this tRNA example under the assumption of $\min_{\text{HL}} = 3$ (which is always implicitly chosen for the PF approach), we see that the quality of sample sets can be further improved by increasing the minimum allowed helix size $\min_{\text{hel}}$. Moreover, under the assumption of the less realistic minimum hairpin loop size $\min_{\text{HL}} = 1$, the generated results are qualitatively not as good as those for $\min_{\text{HL}} = 3$.

6.6.2.3. The Problem of Overfitting and the Lack of Generalization

In this section we will address two possible issues of our sophisticated grammar in connection with this study: the problem of overfitting and the lack of generalization.

With respect to the latter, it might not be surprising to some readers that the profile plots for *Escherichia coli* tRNA^{Ala} presented in Figure 6.1 indicate an accuracy gain of the probabilistic SCFG approach over the physics-based PF variant for the following reasons: First, it seems inevitable that a sophisticated stochastic model that is trained on trusted tRNAs only produces the typical tRNA cloverleaf shape more often than an alternative variant that is not tailored to a specific structure class but only relies on free energy, such that the SCFG based profiles should inherently show the cloverleaf structure more explicitly. Additionally, it is known that SCFG based approaches work well for short RNA types whose molecules imply a low structural variety, whereas the standard thermodynamic model for RNA secondary structures might perform poorly on some tRNAs [RCM99].

For these reasons, it might be assumed that the higher accuracy reached by the probabilistic sampling approach could be an artefact caused by a lack of generalization of the underlying SCFG model. To show that this is not the case, we performed a series of experiments based on (more and less arbitrary) random sequences. In principle, for any chosen value of $\min_{\text{hel}} \in \{0, \ldots, 7\}$, we generated a set of random RNA sequences in the following way: for a considered sequence length n, we randomly created a number of (not necessarily distinct) secondary structures of size n having the cloverleaf shape, where all four helices (the stem and the three adjacent helices of the multiloop) are formed by exactly $\min_{\text{hel}}$ consecutive base pairs. For any of these cloverleaf structures, we then generated a corresponding sequence by randomly drawing canonical base pairs for the helical regions and arbitrary unpaired bases for the single-strands. Obviously, regardless of the applied sampling approach, the signal towards generating the actual cloverleaf structure should get stronger with increasing value of $\min_{\text{hel}}$ and for

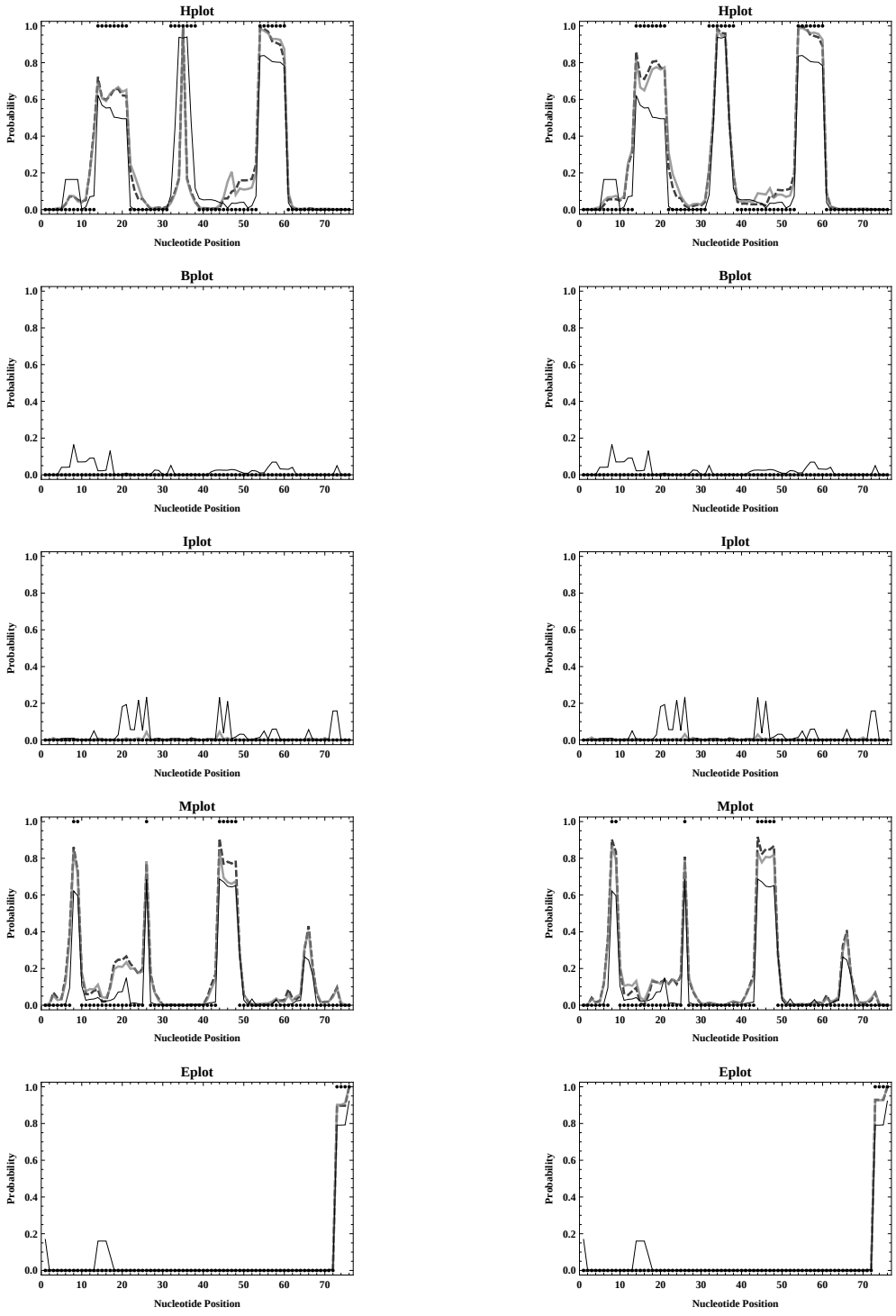


Figure 6.1.: **Comparison of profiling results for PF and SCFG approach.** Figure shows loop profiles for *E.coli* tRNA^{Ala}, obtained with the PF approach and the SCFG variant. Hplot, Bplot, Iplot, Mplot and Eplot display the probability that an unpaired base lies in a hairpin, bulge, interior, multi-branched and exterior loop, respectively. For each considered variant, these five probabilities are computed by a sample of 1000 structures generated by using $\max_{bulge} = 30$. Results for the PF approach are displayed by the thin black lines. For the SCFG approach, we chose $\min_{\text{hel}} = 1$ (thick gray lines) and $\min_{\text{hel}} = 2$ (thick dashed darker gray lines), combined with $\min_{\text{HL}} = 1$ (figures shown on the left) and $\min_{\text{HL}} = 3$ (figures on the right), respectively. The corresponding probabilities for the correct structure of *E.coli* tRNA^{Ala} are also displayed (by black points).

Approach	min_{hel}	num_d	c_d	c_{MF}	c_{CL}	num_{MF}
PF	0	36	8333.33	94085	3331	6
	1	34	8823.53	87785	5338	6
	2	35	8571.43	96083	2745	6
	3	37	8108.11	95332	4492	6
	4	30	10000.	107881	9967	6
	5	29	10344.8	111716	20875	3
	6	33	9090.91	102788	49733	2
	7	27	11111.1	94859	94859	0
SCFG	0	858	349.65	26341	14114	5
	1	916	327.511	22643	15596	4
	2	915	327.869	21258	13912	4
	3	895	335.196	20175	16207	2
	4	914	328.228	19828	17784	2
	5	844	355.45	20560	20560	0
	6	747	401.606	34753	34753	0
	7	658	455.927	59644	59644	0

Table 6.5.: **Results derived from random data sets.** Parameter min_{hel} (defining the minimum allowed length of helical regions) has been used as structural constraint for the generation of random sequences with corresponding (more or less strong) signals towards a cloverleaf structure. num_d denotes the number of distinct shapes (here, *abstract shapes* of level 5 according to [JRG08]) in all samples and c_d the average count of one of these distinct shapes. Furthermore, c_{MF} and c_{CL} represent the count of the most frequent and cloverleaf shape in all samples, whereas num_{MF} denotes the number of distinct shapes that are observed more frequently than the cloverleaf. For any setting of min_{hel}, all tabulated values were computed from a corresponding random data set of cardinality 300 (containing 10 random sequences for any length $n \in \{64, \ldots, 93\}$ according to the length range observed from our tRNA database), respectively. A sample size of 1000 structures and $max_{bulge} = 30$ has been chosen for either approach.

$min_{hel} = 0$, there is absolutely no signal towards the cloverleaf shape, since the corresponding structures have been generated completely at random (by drawing all nucleotides in the sequence independently).

As we can see from Table 6.5 (where the corresponding results have been derived for the most abstract shape level 5), both sampling approaches tend to primarily generating cloverleaf structures if the signals are strong enough, but other shapes are sampled more often if the signal towards cloverleaf is low or does actually not exist. Basically, the SCFG based variant seems to react faster to such signals (by preferring the cloverleaf shape over others more notably already for rather low signals compared to the PF method). However, since for actual random sequences, the typical cloverleaf shape of tRNAs is neither sampled all the time nor significantly more often than any other shape (among a vast number of distinct ones that are observed), there is no reason to believe that the accuracy of the SCFG based sampling strategy (at least for tRNAs) is due to a lack of generalization (or the other way round is due to a model tailored to a certain shape). Since we most likely observe such effects in connection with tRNA and its invariant cloverleaf shape, we skipped similar investigations for the other cases.

$\mathbb{V}[\cdot]$	tRNA	5S rRNA	S-151Rfam
p_1	0	0	0
p_2	5.747×10^{-8}	2.232×10^{-6}	1.613×10^{-5}
p_3	1.223×10^{-7}	6.635×10^{-6}	8.673×10^{-6}
p_4	3.745×10^{-8}	2.718×10^{-6}	1.012×10^{-5}
p_5	9.954×10^{-7}	3.437×10^{-6}	1.983×10^{-5}
p_6	9.579×10^{-7}	1.697×10^{-6}	4.120×10^{-5}
p_7	8.853×10^{-6}	2.849×10^{-5}	7.766×10^{-6}
p_8	8.853×10^{-6}	2.849×10^{-5}	7.766×10^{-6}
p_9	0	0	0
p_{10}	0	0	0
p_{11}	4.541×10^{-9}	1.385×10^{-9}	1.362×10^{-6}
p_{12}	2.645×10^{-8}	8.330×10^{-8}	2.264×10^{-6}
p_{13}	8.500×10^{-9}	6.674×10^{-8}	4.074×10^{-6}
p_{14}	6.762×10^{-10}	3.464×10^{-10}	3.270×10^{-7}
p_{15}	0	0	0
p_{16}	1.234×10^{-8}	7.211×10^{-9}	5.812×10^{-6}
p_{17}	1.234×10^{-8}	7.211×10^{-9}	5.812×10^{-6}
p_{18}	0	1.152×10^{-6}	5.352×10^{-5}
p_{19}	0	3.919×10^{-7}	2.957×10^{-5}
p_{20}	0	4.502×10^{-7}	8.094×10^{-5}
p_{21}	2.695×10^{-3}	2.997×10^{-8}	4.429×10^{-5}
p_{22}	2.695×10^{-3}	2.997×10^{-8}	4.429×10^{-5}
p_{23}	0	0	0
p_{24}	0	0	0
p_{25}	0	0	1.333×10^{-4}
p_{26}	0	0	1.333×10^{-4}
p_{27}	4.052×10^{-7}	1.561×10^{-7}	1.347×10^{-4}
p_{28}	4.052×10^{-7}	1.561×10^{-7}	1.347×10^{-4}
p_{29}	0	0	0

Table 6.6.: **Truncated variances of grammar parameters (transition probabilities).** Values were derived from 100 iterations of training the traditional (length-independent) SCFG $\mathcal{G}_s$ on random subsets containing 90 percent of the original data, respectively, under the assumption of $\min_{HL} = 3$ and $\min_{hel} = 2$.

To see if overfitting is not a problem for our experiments, that is to see if our data sets are rich enough to reliably derive the parameters of our grammar, we performed the following experiments: For each RNA type considered and $\min_{hel} = 2$, $\min_{HL} = 3$ we selected a random 90% portion of the original database[15] and re-estimated the probabilities of all the grammar rules. This process was iterated 100 times, resulting in a sample of 100 parameter sets. Finally, for each parameter we determined its variance along this sample of size 100. The corresponding values are presented in Table 6.6.

Note that the variances 0 in most cases result for intermediate symbols without alternatives; for their productions a probability of 1 is predetermined. However, all the other variances are rather small too and we can conclude that overfitting is no issue in connection with our sophisticated grammar and the training sets used.

[15]Such that the sample size equals that of the training sets used for subsequent k-fold cross-validation experiments.

6.6.2.4. Prediction Accuracy – Sensitivity and PPV

To compare the quality of predictions derived from samples generated by the PF approach to those implied by our SCFG, we again performed two-fold cross-validations based on the mixed S-151Rfam data set. Furthermore, we partitioned the more comprehensive tRNA and 5S rRNA databases into 10 approximately equal-sized folds and derived corresponding 10-fold cross-validations results, respectively.

Note that for any sequence, we predicted one structure according to each of the principles introduced in Section 6.5, where for the sake of completeness we considered the default choice $\gamma_{t-o} = 1$ for MEA and centroid structures, as well as varying values for γ_{t-o} (the same ones as considered above) to obtain AUC values. The determined sensitivity and PPV measures are collected in Tables 6.7a to 6.9a, whereas the corresponding AUC values are reported in Tables 6.7b to 6.9b; plots of the respective ROC curves can be found in Figures B.1 to B.3 of Section B. Obviously, the provided AUC values allow for a reliable comparison of the accuracies that can be reached by either sampling approach when calculating γ_{t-o}-MEA and γ_{t-o}-centroid structures for the produced samples.

Let us first consider the results presented in Table 6.7. Here, we observe that for the low invariant tRNAs, the accuracy of predictions computed by statistical sampling methods can be significantly improved when using the SCFG approach. Moreover, the quality of predictions can be further improved by considering the realistic value of $\min_{HL} = 3$ (also implicitly chosen for the PF approach) instead of the unrealistic choice $\min_{HL} = 1$. However, it seems that increasing the value of parameter $\min_{hel}$ does not have a mentionable impact on the resulting prediction accuracy.

According to Table 6.8, the predictions for 5S rRNAs are less accurate than for tRNAs. In detail, for 5S RNAs the predictive accuracy as measured by sensitivity and PPV is slightly higher for the PF approach when selecting the most frequently sampled structure as prediction. By constructing a MEA structure and especially the unique centroid structure, however, we observe significant differences between both sensitivity and PPV obtained by either sampling approach. The corresponding AUCs confirm the advantages of the PF approach on these data. Furthermore, the case $\gamma_{t-o} = 1$ implies that base pairings of the native foldings generally occur less frequently in samples generated by the SCFG based algorithm (FN is greater), but the sampled pairs are more often correct (FP is smaller). Considering the unique centroid predictions, this means that the SCFG method rarely samples incorrect pairings (otherwise, those would be part of the prediction), while pairs which are sampled with a high frequency typically are native ones. This decreased precision may be implied by the comparably high structural diversity of 5S rRNAs and the corresponding reduced ability of our SCFG model to capture typical structural features of the considered family within its parameters.

Last but not least, similar results can be observed for the S-151Rfam data set in connection with the default choice $\gamma_{t-o} = 1$, as shown in Table 6.9a. In fact, the performance gap between the two different sampling approaches remains quite the same as for our 5S rRNA database, although this mixed data set is less comprehensive and contains structures that not only belong to distinct RNA types but also partially contained pseudoknots that had to be removed, such that this S-151Rfam set might not be considered a high-quality training basis. In contrast to the 5S rRNAs however, considering the AUC values of Table 6.9b reveals slight advantages of our SCFG over PFs.

In conclusion, we have three different scenarios for the three different data sets: for tRNAs our SCFG performs best for fix and varying γ_{t-o}, for 5S rRNA the PF approach is superior in both cases and for the S-151 Rfam data set the SCFG is beaten by the PF approach for $\gamma_{t-o} = 1$ while the SCFG gives rise to better AUCs.

Approach	Parameters	MF struct.		MEA struct.		Centroid	
		Sens.	PPV	Sens.	PPV	Sens.	PPV
PF	$\max_{bulge} = 30$	0.6565	0.5890	0.6434	0.6035	0.6159	0.6344
SCFG	$\min_{HL} = 1, \min_{hel} = 1$	0.7791	0.8445	0.7324	0.8939	0.6754	0.9158
	$\min_{HL} = 1, \min_{hel} = 2$	0.8004	0.8457	0.7685	0.8878	0.7113	0.9123
	$\min_{HL} = 3, \min_{hel} = 1$	0.8545	0.8517	0.7848	0.9021	0.7304	0.9213
	$\min_{HL} = 3, \min_{hel} = 2$	0.8677	0.8593	0.8182	0.8953	0.7713	0.9168

(a) Sensitivity and PPV.

Approach	Parameters	MEA struct.	Centroid
PF	$\max_{bulge} = 30$	0.482435	0.526743
SCFG	$\min_{HL} = 1, \min_{hel} = 1$	0.828522	0.833894
	$\min_{HL} = 1, \min_{hel} = 2$	0.830787	0.839843
	$\min_{HL} = 3, \min_{hel} = 1$	0.855406	0.861640
	$\min_{HL} = 3, \min_{hel} = 2$	0.857251	0.867135

(b) AUC values.

Table 6.7.: **Prediction results for our tRNA database.** Tabulated values were computed by 10-fold cross-validation procedures, using sample size 1000.

Approach	Parameters	MF struct.		MEA struct.		Centroid	
		Sens.	PPV	Sens.	PPV	Sens.	PPV
PF	$\max_{bulge} = 30$	0.5897	0.5806	0.6015	0.6191	0.5789	0.6508
SCFG	$\min_{HL} = 1, \min_{hel} = 1$	0.4251	0.5362	0.3403	0.6967	0.2689	0.8044
	$\min_{HL} = 1, \min_{hel} = 2$	0.4542	0.5435	0.3638	0.6901	0.2727	0.8069
	$\min_{HL} = 3, \min_{hel} = 1$	0.4728	0.5290	0.3544	0.7033	0.2764	0.8091
	$\min_{HL} = 3, \min_{hel} = 2$	0.5167	0.5577	0.3860	0.7010	0.2846	0.8140

(a) Sensitivity and PPV.

Approach	Parameters	MEA struct.	Centroid
PF	$\max_{bulge} = 30$	0.481019	0.520171
SCFG	$\min_{HL} = 1, \min_{hel} = 1$	0.409278	0.408549
	$\min_{HL} = 1, \min_{hel} = 2$	0.417286	0.418584
	$\min_{HL} = 3, \min_{hel} = 1$	0.419116	0.417095
	$\min_{HL} = 3, \min_{hel} = 2$	0.433954	0.431642

(b) AUC values.

Table 6.8.: **Prediction results for our 5S rRNA database.** Tabulated values were computed by 10-fold cross-validation procedures, using sample size 1000.

Approach	Parameters	MF struct.		MEA struct.		Centroid	
		Sens.	PPV	Sens.	PPV	Sens.	PPV
PF	$\max_{\mathtt{bulge}} = 30$	0.6652	0.5188	0.6633	0.5450	0.6437	0.5799
SCFG	$\min_{\mathrm{HL}} = 1, \min_{\mathrm{hel}} = 1$	0.4433	0.5447	0.3815	0.7386	0.3235	0.7749
	$\min_{\mathrm{HL}} = 1, \min_{\mathrm{hel}} = 2$	0.4894	0.5551	0.4263	0.7181	0.3474	0.7743
	$\min_{\mathrm{HL}} = 3, \min_{\mathrm{hel}} = 1$	0.4852	0.5948	0.3935	0.7426	0.3352	0.7825
	$\min_{\mathrm{HL}} = 3, \min_{\mathrm{hel}} = 2$	0.5171	0.5661	0.4342	0.7228	0.3588	0.7683

(a) Sensitivity and PPV.

Approach	Parameters	MEA struct.	Centroid
PF	$\max_{\mathtt{bulge}} = 30$	0.450688	0.497350
SCFG	$\min_{\mathrm{HL}} = 1, \min_{\mathrm{hel}} = 1$	0.499491	0.507125
	$\min_{\mathrm{HL}} = 1, \min_{\mathrm{hel}} = 2$	0.506602	0.509403
	$\min_{\mathrm{HL}} = 3, \min_{\mathrm{hel}} = 1$	0.507454	0.512327
	$\min_{\mathrm{HL}} = 3, \min_{\mathrm{hel}} = 2$	0.508762	0.514958

(b) AUC values.

Table 6.9.: **Prediction results for the mixed S-151Rfam database.** Tabulated values were computed by two-fold cross-validation procedures, using the same folds as in [DWB06] and sample size 1000.

6.6.2.5. Sampling Quality – Specific Values Related to Shapes

Note that the previously considered measures for assessing the accuracy of secondary structure predictions (sensitivity and PPV) depend only on the numbers of correctly and incorrectly predicted base pairs (compared to the native structure). From the perspective of biologists, however, it is usually much more important to get information on the correct structural properties (described by the corresponding abstract shapes) of the native folding than to obtain high sensitivity and PPV when using computational prediction methods.

Therefore, in order to further investigate the sampling quality, we decided to consider the following specific values related to the shapes of sampled structures:

- *Frequency of prediction of correct structure* ($\mathrm{CSP}_{\mathrm{freq}}$): In how many cases is the predicted secondary structure (or its shape) equal to the correct structure (or the correct shape)?
- *Frequency of correct shape occurring in a sample* ($\mathrm{CSO}_{\mathrm{freq}}$): In how many cases can the correct shape (on different levels) be found in the generated sample set?
- *Number of occurrences of correct shape in a sample* ($\mathrm{CS}_{\mathrm{num}}$): How many times can the correct shape be found in the generated sample set?
- *Number of different shapes in a sample* ($\mathrm{DS}_{\mathrm{num}}$): How many different secondary structures (or shapes) can be found in the generated sample set?

To compute the desired values, we considered the predicted structures and the corresponding sample sets that were derived for the calculation of the sensitivity and PPV measures in the last section (Tables 6.7a to 6.9a). The respective results are collected in Tables B.1 to B.6 in Section B. Some of the most interesting ones are reported in Tables 6.10 to 6.12.

Comparing the corresponding values, we immediately observe that for our tRNA and 5S rRNA databases, the predicted shapes are in almost all cases significantly more often equal to the correct ones when using the SCFG based sampling strategy instead of the PF alternative. This means given rich and explicit training data, the frequency of correct structure predictions

Value	Variant	Parameters	Shape Level					
			0	1	2	3	4	5
CSP_{freq}	PF	$max_{bulge} = 30$	0.0633	0.1216	0.2071	0.2117	0.2639	0.3694
(MF)	SCFG	$min_{HL} = 3, min_{hel} = 1$	0.2450	0.4448	0.6417	0.6417	0.6422	0.7356
CSP_{freq}	PF	$max_{bulge} = 30$	0.0416	0.1049	0.1923	0.1960	0.2496	0.3559
(MEA)	SCFG	$min_{HL} = 3, min_{hel} = 2$	0.1008	0.2917	0.5525	0.5525	0.5543	0.6241
CSP_{freq}	PF	$max_{bulge} = 30$	0.0264	0.0800	0.1595	0.1627	0.1932	0.2677
(Centroid)	SCFG	$min_{HL} = 3, min_{hel} = 2$	0.0758	0.2150	0.4563	0.4563	0.4568	0.5003
CSO_{freq}	PF	$max_{bulge} = 30$	0.5196	0.6740	0.8160	0.8239	0.8798	0.9556
	SCFG	$min_{HL} = 3, min_{hel} = 1$	0.7148	0.9459	0.9875	0.9880	0.9885	0.9991
CS_{num}	PF	$max_{bulge} = 30$	21.073	58.200	136.67	140.63	205.54	328.56
	SCFG	$min_{HL} = 3, min_{hel} = 2$	34.898	173.73	513.05	513.06	513.08	595.26
DS_{num}	PF	$max_{bulge} = 30$	355.32	130.22	81.796	33.125	22.585	4.8848
	SCFG	$min_{HL} = 3, min_{hel} = 2$	592.84	103.04	18.921	18.921	18.921	12.053

Table 6.10.: **Comparison of sampling quality for tRNAs.** Table contains results related to the shapes of selected predictions and sampled structures, obtained from our tRNA database. They were computed by 10-fold cross-validation procedures, using sample size 1000.

Value	Variant	Parameters	Shape Level					
			0	1	2	3	4	5
CSP_{freq}	PF	$max_{bulge} = 30$	0.0000	0.0009	0.0078	0.0513	0.0261	0.6353
(MF)	SCFG	$min_{HL} = 3, min_{hel} = 2$	0.0009	0.0096	0.0244	0.0609	0.1027	0.8207
CSP_{freq}	PF	$max_{bulge} = 30$	0.0000	0.0052	0.0139	0.0835	0.0696	0.6640
(MEA)	SCFG	$min_{HL} = 3, min_{hel} = 2$	0.0000	0.0009	0.0009	0.0035	0.0557	0.5387
CSP_{freq}	PF	$max_{bulge} = 30$	0.0000	0.0026	0.0104	0.0775	0.0731	0.7214
(Centroid)	SCFG	$min_{HL} = 3, min_{hel} = 2$	0.0000	0.0000	0.0000	0.0009	0.0139	0.1549
CSO_{freq}	PF	$max_{bulge} = 30$	0.0009	0.1662	0.3063	0.7580	0.6883	0.9817
	SCFG	$min_{HL} = 3, min_{hel} = 2$	0.0026	0.4509	0.6372	0.9904	0.9974	0.9991
CS_{num}	PF	$max_{bulge} = 30$	0.0009	0.7571	3.4207	36.641	30.288	600.35
	SCFG	$min_{HL} = 3, min_{hel} = 2$	0.0026	1.3795	3.1949	36.673	71.080	609.58
DS_{num}	PF	$max_{bulge} = 30$	710.75	333.72	237.71	93.335	63.661	7.0951
	SCFG	$min_{HL} = 3, min_{hel} = 2$	999.68	885.81	762.67	239.28	123.91	13.558

Table 6.11.: **Comparison of sampling quality for 5S rRNAs.** Table contains results related to the shapes of selected predictions and sampled structures, obtained from our 5S rRNA database. They were computed by 10-fold cross-validation procedures, using sample size 1000.

Value	Variant	Parameters	Shape Level					
			0	1	2	3	4	5
CSP_{freq}	PF	$max_{bulge} = 30$	0.0661	0.1255	0.1586	0.2050	0.2183	0.4834
(MF)	SCFG	$min_{HL} = 3, min_{hel} = 2$	0.0530	0.1258	0.1522	0.1788	0.1985	0.4240
CSP_{freq}	PF	$max_{bulge} = 30$	0.0660	0.1123	0.1453	0.1984	0.2051	0.4902
(MEA)	SCFG	$min_{HL} = 1, min_{hel} = 2$	0.0264	0.1193	0.1391	0.1523	0.1789	0.4239
CSP_{freq}	PF	$max_{bulge} = 30$	0.0793	0.1321	0.1653	0.1917	0.2449	0.5100
(Centroid)	SCFG	$min_{HL} = 3, min_{hel} = 2$	0.0197	0.0927	0.1125	0.1390	0.1391	0.3577
CSO_{freq}	PF	$max_{bulge} = 30$	0.3638	0.4433	0.4766	0.5231	0.6488	0.7947
	SCFG	$min_{HL} = 1, min_{hel} = 2$	0.2717	0.5630	0.6158	0.7284	0.8079	0.9605
CS_{num}	PF	$max_{bulge} = 30$	40.390	88.886	121.55	158.32	195.83	453.58
	SCFG	$min_{HL} = 3, min_{hel} = 2$	15.059	63.707	83.965	125.82	142.99	391.39
DS_{num}	PF	$max_{bulge} = 30$	540.74	304.36	255.40	150.89	117.24	18.795
	SCFG	$min_{HL} = 3, min_{hel} = 2$	840.03	522.53	452.04	307.61	273.92	77.536

Table 6.12.: **Comparison of sampling quality for the S-151Rfam set.** Table contains results related to the shapes of selected predictions and sampled structures, obtained from the S-151Rfam database. They were computed by two-fold cross-validation procedures, using sample size 1000.

(CSP_{freq}) is basically higher when relying on the ensemble distribution induced by our sophisticated SCFG. Moreover, the samples generated with the SCFG method generally contain the correct shapes considerably more often than those obtained with the corresponding PF algorithm and are thus more accurate as regards the frequency of correct structure occurrences (CSO_{freq}).

However, having only a lean training set of mixed RNAs like the S-151Rfam database at hand, then the energy-based sampling approach seems to outperform its probabilistic counterpart, at least with respect to shape prediction.

Furthermore, as regards tRNAs and 5S rRNAs, the observed averaged number of correct shapes in a sample set (CS_{num}) is greater when using the SCFG approach, whereas for the S-151Rfam set of mixed structural RNAs, an arbitrary sample obviously contains more instances of the correct shape when using the PF variant. For 5Sr RNAs, this observation especially holds for the two most interesting shape types (the most accurate shape type 1 and the most abstract type 5) with the realistic parameter choice $min_{hel} = 2$ for the SCFG strategy (see Table B.4 or Table 6.11). Finally, the observed averaged number of different shapes in a sample (DS_{num}) is in most cases significantly larger for the SCFG based sampling method[16]. This actually means that samples generated according to the distribution induced by a sophisticated SCFG design imply a greater diversity of candidate structures for a given input sequence than corresponding Boltzmann samples.

Consequently, the SCFG based statistical sampling approach evaluated within this chapter effectively overcomes the main pitfall of MFE based methods addressed in Section 6.1, namely that the predicted set of suboptimal foldings for a given sequence usually contains mostly structures without fundamental differences. However, there is neither clear evidence that the distribution induced by a sophisticated SCFG generally yields more realistic results than a corresponding energy-based Boltzmann distribution, nor the other way round. In fact, this seems to strongly depend on the RNA type of the given sequence, and most importantly on the quality of a corresponding training set and on the performance of the thermodynamic model on

[16]Note that in the few cases where the PF approach yields more different shapes, we generally further restricted the possible structures by prohibiting isolated base pairs ($min_{hel} > 1$), which are in fact allowed in PF calculations.

such RNAs. Altogether, we conclude that fundamental differences might be expected between Boltzmann samples and corresponding statistical sample sets obtained by a sophisticated SCFG approach, which finally disproves hypothesis H_0 proposed in Section 6.1.

6.7. Conclusions

By comprehensive comparisons, we showed that incorporating only additional information obtained from databases of trusted RNA sequences with annotated secondary structures (SCFG variant) instead of the recent thermodynamic parameters for RNA secondary structure (PF variant) into a statistical sampling algorithm results in significant differences with respect to both predictive accuracy and overall quality of generated sample sets. Actually, we can draw the conclusion that the ensemble distribution induced by the considered sophisticated SCFG is less centered than the corresponding Boltzmann distribution of possible structures. This effectively yields more variability during the sampling process and consequently reduces the problem of getting stuck in local optima (which is inevitably inherited from optimization algorithms), resulting in a more diverse sample set that might also contain structures which are fundamentally different to the most probable ones. Thus, the discussed probabilistic sampling approach may be used to address exactly the critical features of deterministic structure prediction methods and hence eventually realizes the intentions related to statistical sampling techniques towards RNA structure prediction. Furthermore, note that despite the potential major quality improvement of the SCFG variant over the PF approach for certain RNA types, the worst-case time complexity and memory requirement for the construction of a statistically representative and reproducible sample for a given sequence are actually the same.

According to these aspects, the SCFG approach that has been evaluated within this chapter may inspire the development of new high quality (sampling) algorithms, for example for RNA structures with pseudoknots or RNA-RNA interactions, due to the following reasons: First, as mentioned in Section 3.4.1, RNA structure prediction including pseudoknots based on thermodynamics is $\mathcal{NP}$-hard, but some MFE based algorithms have been developed to include certain types of pseudoknots [RE99, RG04]. Moreover, the PF algorithm [McC90] has been extended to include a class of pseudoknots [DP03, DP04], such that a sampling extension could also be developed for structures including pseudoknots. However, one of the main problems with these approaches towards pseudoknot prediction is their dependence on the thermodynamic parameters and energy functions which limits the performance accuracies in very significant ways, since there exists little knowledge on the thermodynamic behavior of pseudoknotted structures. Nevertheless, it is known how to model RNA secondary structures with pseudoknots (and also RNA-RNA interactions) by special more powerful grammar models. For example by so-called *multiple context-free grammars (MCFGs)*, which are defined by a weakly context-sensitive grammar formalism that deals with tuples of strings (see, for instance [SMFK91, KSK06, Kat07, KAS09, Wil10]). Notably, MCFGs for all common classes of RNA pseudoknots are presented in [NW]. The key point is that when applying such grammar models, one does not have to face the problem that no appropriate energy parameters are available. Hence, by completely abstracting from thermodynamics and considering only typical structural information obtained by training a convenient grammar on structural databases, one might be able to generalize the sampling strategy discussed in this chapter to a competitive algorithm for predicting pseudoknotted RNA secondary structures (or RNA-RNA interactions).

Chapter 7

Statistical Sampling Based on a Length-Dependent SCFG Model

The aim of this chapter is to present a comprehensive study on how enriching the underlying SCFG by additional information on the lengths of generated substructures (that is, by incorporating *length-dependencies* into the SCFG based sampling algorithm, which is actually possible without significant losses in performance) affects the reliability of the induced RNA model and the accuracy of sampled secondary structures.

As we will see, significant differences with respect to the overall quality of generated sample sets and the resulting predictive accuracy are typically implied. In principle, when considering the more specialized length-dependent SCFG model as basis for statistical sampling, a higher accuracy of predicted foldings can be reached at the price of a lower diversity of generated candidate structures (compared to the more general traditional SCFG variant or sampling based on PFs that rely on free energies).

7.1. Objectives and Outline

Motivated by the idea discussed in [Mai07] of improving the SCFG approach for RNA secondary structure prediction by explicitly considering the *lengths* of particular substructures, the main objectives of this chapter are given as follows: First, we want to investigate to which extend the additional incorporation of length-dependencies into a sophisticated SCFG changes the quality of the induced probabilistic RNA model. Our second aim is to quantify the differences in resulting accuracy that can be observed when applying both probabilistic models (length-dependent and traditional one) to identical inputs. For our examinations, we decided to rely on the elaborate SCFG design devised in Section 6.3 and analogously use it as the basis for a probabilistic statistical sampling algorithm, since this effectively makes it possible to perform comprehensive comparisons of both variants with respect to different meaningful applications that can immediately be considered in connection with sampling approaches. In fact, we will present a fundamental analysis of the resulting sample sets from different relevant perspectives in order to see if the incorporation of additional length information into SCFGs yields a quality improvement.

The plan for the rest of the chapter is given as follows: Section 7.2 formally introduces the considered (L)SCFG model for RNA secondary structures by determining appropriate length intervals. The needed modifications of the original sampling algorithm as presented in Section 6.4 to manage the additional length-dependencies are described in Section 7.3. Section 7.4 discusses the potentials and possible drawbacks of extending the underlying sophisticated SCFG model to a length-dependent one. Particularly, Section 7.4 contains important results concerning the quality of the underlying probabilistic model with respect to both overfitting and lack of generalization. Furthermore, it examines if adding length-dependency actually improves the accuracy of predictions obtained from statistical sampling and the overall quality of generated sample sets (with respect to the produced shapes). For this purpose, corresponding results obtained by the length-dependent and the traditional version of the probabilistic sampling approach are opposed to each other. Additionally, all results are compared to corresponding ones obtained with the competing PF approach implemented in the `Sfold` program for further judgements. Finally, Section 7.5 concludes this chapter.

7.2. Finding Appropriate Length Intervals

As described in Section 3.3.8, when considering a particular LSCFG, it is reasonable to group the lengths together into several intervals, since then the grammar parameters (transition and emission probabilities) can be stored as a vector, respectively, which makes it possible to retrieve them in algorithms and applications without further computational efforts. However, in order to guarantee that a specific grouping results in a consistent LSCFG, we have to find a partitioning Q of $\mathbb{N}$ that is consistent according to Definition 3.3.5 (see [WN11] for details). Furthermore, the interval lengths should principally grow with increasing subword length, where the lengths of the last interval should be chosen in accordance with the training data (see Section 3.3.8).

Therefore, in order to find appropriate length intervals for applications based on grammar $\mathcal{G}_s$, we first partition the productions in $R_{\mathcal{G}_s}$ into subsets, where each subset generates terminal words of different lengths. Let $m_p := (2 \cdot m_s + m_h)$ denote the minimum allowed size of a paired substructure. Furthermore, recall that in this thesis, we will only consider the common choices of $m_h = \min_{HL} \in \{1, 3\}$ and $m_s = \min_{hel} \in \{1, 2\}$. Then, we obtain the following:

Lengths		Rules that can actually produce a terminal word of these lengths		
$= 0$		$p_{28} : U \to \epsilon$,		
$\geqslant 0$		$p_{26} : N \to U$,		
$= 1$		$p_8 : C \to Z$,	$p_{17} : H \to Z$,	$p_{22} : B \to Z$,
		$p_{29} : Z \to \circ$,		
$\geqslant 1$		$p_1 : S \to T$,	$p_2 : T \to C$,	$p_{27} : U \to ZU$,
$\geqslant 2$		$p_7 : C \to ZC$,	$p_{16} : H \to ZH$,	$p_{21} : B \to ZB$,
$\geqslant m_h$	$\in [1;3]$	$p_{11} : L \to F$,	$p_{15} : F \to Z^{m_h-1}H$,	
$\geqslant m_h + 2$	$\in [3;5]$	$p_{10} : P \to (L)$,	$p_{12} : L \to P$,	
$\geqslant m_p$	$\in [3;7]$	$p_3 : T \to A$,	$p_9 : A \to ({}^{m_s}L)^{m_s}$,	
		$p_{24} : O \to UAN$,	$p_{25} : N \to UAN$,	
$\geqslant m_p + 1$	$\in [4;8]$	$p_4 : T \to CA$,	$p_5 : T \to AT$,	
		$p_{13} : L \to G$,	$p_{18} : G \to BA$,	$p_{19} : G \to AB$,
$\geqslant m_p + 2$	$\in [5;9]$	$p_6 : T \to CAT$,	$p_{20} : G \to BAB$,	
$\geqslant 2 \cdot m_p$	$\in [6;14]$	$p_{14} : L \to M$,	$p_{23} : M \to UAO$.	

Consequently, a partitioning Q of $\mathbb{N}$ that is consistent with $\mathcal{G}_s$ (according to Definition 3.3.5) can be given as follows:

$$\begin{aligned} Q \;=\; & \big\{[0;0],[1;1],[2;\max(2,m_h-1)]\big\} \cup \\ & \big\{[m_h;m_h],[m_h+1;m_h+1],[m_h+2;\max(m_h+2,m_p-1)]\big\} \cup \\ & \big\{[m_p;m_p],[m_p+1;m_p+1],[m_p+2;2\cdot m_p-1],[2\cdot m_p;\infty]\big\}. \end{aligned}$$

However, for the sake of simplicity, it would be more convenient to consider only a grouping of lengths into intervals that is appropriate for all our different structural parameter choices. In order to keep the estimated probabilities accurate, we decided to make the intervals longer as the considered subwords get longer, for the following two reasons: First, since typically any training set contains fewer data points per length as the length gets longer and second, since the influence a change in length has on the probabilities of productions most likely depends on the relative change rather than the absolute one.

Therefore, we decided to use the successively increasing intervals[1]

$$\begin{aligned} [i;i] \quad & \text{for} \quad 0 \leqslant i \leqslant 40, \\ [i;i+1] \quad & \text{for} \quad 41 \leqslant i \leqslant 59, \\ [i;i+2] \quad & \text{for} \quad 61 \leqslant i \leqslant 82, \\ [i;i+3] \quad & \text{for} \quad 85 \leqslant i \leqslant 97, \\ [i;i+4] \quad & \text{for} \quad 101 \leqslant i \leqslant 136, \\ [i;i+9] \quad & \text{for} \quad 141 \leqslant i \leqslant 191, \quad \text{and} \\ [i;i+19] \quad & \text{for} \quad 201 \leqslant i \leqslant 281, \end{aligned}$$

together with the longer intervals

$$\begin{aligned} & [301;330], \quad [331;360], \quad [361;390], \\ & [391;430], \quad [431;470], \quad [471;510], \\ & [511;560], \quad [561;610], \\ & [611;670], \quad [671;730], \\ & [731;800], \quad [801;900], \quad [901;1000], \quad \text{and} \\ & [1001;\infty]. \end{aligned}$$

[1]Note that these are basically the same intervals as used in [WN11] and have thus proven convenient.

Obviously, this partitioning of $\mathbb{N}$ (into 105 distinct length intervals) is consistent with $\mathcal{G}_s$ for all considered structural parameter choices and thus in any case yields a consistent LSCFG $\mathcal{G}_s$.

Finally, note that under the common assumption that all emissions come from the same distribution, there are accordingly at most

$$\begin{aligned} & (\mathtt{card}(\mathcal{R}_{\mathcal{G}_s}) - \mathtt{card}(\mathcal{I}_{\mathcal{G}_s})) \cdot 105 + \mathtt{card}(\Sigma_{\mathcal{G}_r}) \cdot 1 + \mathtt{card}(\Sigma_{\mathcal{G}_r})^2 \cdot 105 \\ = \; & (29 - 15) \cdot 105 + 4 + 16 \cdot 105 \\ = \; & 1470 + 4 + 1680 \\ = \; & 3154 \end{aligned}$$

free parameters that need to be estimated for the LSCFG $\mathcal{G}_s$ when considering these 105 intervals. This number is obviously indeed to a large extend greater than the corresponding parameter number (of at most 34) implied in case of the plain SCFG $\mathcal{G}_s$.

However, it should be mentioned that the actual number of relevant (that is, being greater than 0) free parameters will usually be much smaller, since a potentially significant amount of the length-dependent probabilities will inevitably always be equal to zero (independent on the used training data). This is due to the partitioning of data points according to different lengths and the constraints imposed by the structural parameters $\min_{\mathrm{hel}}$ and $\min_{\mathrm{HL}}$. For instance, as regards multiloops, we might only obtain $\Pr(L \to M, l) \neq 0$ for $l \geqslant 2 \cdot m_p$, whereas for $l < 2 \cdot m_p$, $\Pr(L \to M, l) = 0$ must always hold.

7.3. Algorithm

In this section, we will describe how to modify the routines and formal definitions proposed in Section 6.4 in order to obtain a corresponding statistical sampling method for RNA secondary structures according to the length-dependent SCFG model defined in the last section. Therefore, recall that in accordance with the popular PF variant presented in [DL03], the SCFG based sampling method has two basic steps. Its first step (preprocessing) computes the inside and outside probabilities for all substrings of an RNA sequence based on the considered SCFG. In the second step (structure sampling), base pairs (and unpaired bases) are randomly drawn according to the conditional sampling probabilities for the considered fragment (that are calculated by using only the inside and outside values derived in step one and the probabilities of the grammar rules) in order to sample complete secondary structures.

7.3.1. Computation of Inside and Outside Probabilities

Notably, if grammar parameters are separated into transitions and emissions, then probabilistic Earley parsing can easily be applied to work for all SCFGs (length-dependent or not) by a few simple modifications of the corresponding subroutines. Basically, for implementing probabilistic Earley parsing for length-dependent grammars, one needs to postpone the inclusion of rule probabilities to the completion step, since only at this point the length of the generated subword is known. In our case, this means that instead of considering both the transition and emission probabilities already in the initial prediction steps, one has to include the right rule probabilities (multiplied by corresponding factors, see Section 3.3.8) in the completion steps and the corresponding emission probabilities in the scanner steps, respectively. Under the assumption that the lengths are grouped together in several intervals, these modifications do not influence the run-time significantly.

The corresponding modified versions of the inside and outside algorithms from Section 6.4.1 are given by Algorithms 7 and 11 (together with their subroutines).

Algorithm 7 Computation of inside values (length-dependent)

Input: RNA sequence r of length $n \geqslant 1$,
set $\mathcal{R}_{\mathcal{G}_s,\bullet}$ of production rules used by Earley's algorithm, and
transition probabilities $\Pr_{tr}(\mathit{rule}, l)$ for the productions $\mathit{rule} \in \mathcal{R}_{\mathcal{G}_s}$, as well as
emission probabilities $\Pr_{em}(x, l)$ for individual unpaired bases $x \in \{a, c, g, u\}$ and
emission probabilities $\Pr_{em}(x_1x_2, l)$ for individual base pairs $x_1x_2 \in \{a, c, g, u\}^2$,
all trained on the same RNA structure data.

for $j = 1$ **to** $n + 1$ **do**
  for $i = j$ **to** 1 **do**
    for $p = 1$ **to** $\mathtt{card}(\mathcal{R}_{\mathcal{G}_s})$ **do**
      for $q = 0$ **to** $k(p)$ **do**
        Consider $\mathit{rule} = \mathrm{ind}^{-1}(p, q) \in \mathcal{R}_{\mathcal{G}_s,\bullet}$ /*i.e. the rule having index (p, q) in our ordering*/
        if $\mathit{rule} = X \to \gamma w_{j-1} \bullet \delta$ **then**
          Scan(i, j, p, q)
        else if $\mathit{rule} = X \to \bullet\gamma$ **then**
          Predict(j, p)
        else if $\mathit{rule} = X \to \gamma Y \bullet \delta$ **then**
          Complete(i, j, p, q)
        end if
      end for
    end for
  end for
end for

Algorithm 8 Predicting inside items

procedure Predict(j, p)
Consider $\mathit{rule} = X \to \bullet\gamma = \mathrm{ind}^{-1}(p, 0) \in \mathcal{R}_{\mathcal{G}_s,\bullet}$ /*i.e. the rule having index $(p, 0)$ in our ordering*/
if $\gamma = \epsilon$ /*rule is ϵ-rule*/ **then**
  $\mathrm{inside}[j, (p, 0), j] = \Pr_{tr}(X \to \gamma, 0)$
else
  $\mathrm{inside}[j, (p, 0), j] = 1$
end if
return
end procedure

Algorithm 9 Scanning inside items

procedure Scan(i, j, p, q)
Consider $\mathit{rule} = X \to \gamma w_{j-1} \bullet \delta = \mathrm{ind}^{-1}(p, q) \in \mathcal{R}_{\mathcal{G}_s,\bullet}$ /*i.e. the rule having index (p, q) in our ordering*/
if $w_{j-1} =' \circ '$ **then**
  $\mathrm{inside}[i, (p, q), j] = \Pr_{em}(r_{j-1}, 1) \cdot \mathrm{inside}[i, (p, q-1), j-1]$
else if $w_{j-1} =' (\,'$ **then**
  $\mathrm{inside}[i, (p, q), j] = \mathrm{inside}[i, (p, q-1), j-1]$
else if $w_{j-1} =')\,'$ **then**
  $\mathrm{inside}[i, (p, q), j] = \Pr_{em}(r_i r_{j-1}, (j-1) - i + 1) \cdot \mathrm{inside}[i, (p, q-1), j-1]$
end if
if $q = k(p)$ /*$\mathit{rule} = X \to \gamma w_{j-1}\bullet$, i.e. rule is completed in this scanning step*/ **then**
  $\mathrm{inside}[i, (p, q), j] = \mathrm{inside}[i, (p, q), j] \cdot \Pr_{tr}(X \to \gamma w_{j-1}, (j-1) - i + 1)$
end if
return
end procedure

Algorithm 10 Completing inside items

procedure Complete(i, j, p, q)
Consider *rule* $= X \to \gamma Y \bullet \delta = \text{ind}^{-1}(p, q) \in \mathcal{R}_{\mathcal{G}_s,\bullet}$ /*i.e. the rule having index (p, q) in our ordering*/
$\text{inside}[i, (p, q), j] = \sum_{k=i}^{j} \left(\text{inside}[i, (p, q-1), k] \cdot \left(\sum_{rule_B \in \mathcal{R}_B} \text{inside}[k, \text{ind}(rule_B), j]\right)\right)$
if $q = k(p)$ /**rule* $= X \to \gamma Y \bullet$, i.e. rule is completed*/ **then**
$\text{inside}[i, (p, q), j] = \text{inside}[i, (p, q), j] \cdot \Pr_{tr}(X \to \gamma Y, (j-1) - i + 1)$
end if
return
end procedure

Algorithm 11 Computation of outside values (length-dependent)

Input: RNA sequence r of length $n \geqslant 1$,
set $\mathcal{R}_{\mathcal{G}_s,\bullet}$ of production rules used by Earley's algorithm, and
transition probabilities $\Pr_{tr}(rule, l)$ for the productions $rule \in \mathcal{R}_{\mathcal{G}_s}$, as well as
emission probabilities $\Pr_{em}(x, l)$ for individual unpaired bases $x \in \{a, c, g, u\}$ and
emission probabilities $\Pr_{em}(x_1 x_2, l)$ for individual base pairs $x_1 x_2 \in \{a, c, g, u\}^2$,
all trained on the same RNA structure data, and also
the corresponding inside values (computed by Algorithm 7).
$\text{outside}[1, \text{ind}(S \to T\bullet), n+1] = 1$ /*initialization*/
for $j = n+1$ **to** 1 **do**
for $i = 1$ **to** j **do**
for $p = \texttt{card}(\mathcal{R}_{\mathcal{G}_s})$ **to** 1 **do**
for $q = k(p)$ **to** 0 **do**
Consider *rule* $= \text{ind}^{-1}(p, q) \in \mathcal{R}_{\mathcal{G}_s,\bullet}$ /*i.e. the rule having index (p, q) in our ordering*/
if *rule* $= X \to \bullet\gamma$ **then**
/*PredictReverse():*/
Do nothing
else if *rule* $= X \to \gamma w_j \bullet \delta$ **then**
ScanReverse(i, j, p, q)
else if *rule* $= X \to \gamma Y \bullet \delta$ **then**
CompleteReverse(i, j, p, q)
end if
end for
end for
end for
end for

Algorithm 12 Scanning outside items

procedure ScanReverse(i, j, p, q)
Consider *rule* $= X \to \gamma w_j \bullet \delta = \text{ind}^{-1}(p, q) \in \mathcal{R}_{\mathcal{G}_s,\bullet}$ /*i.e. the rule having index (p, q) in our ordering*/
if $w_j = '\,.\,'$ **then**
$\text{outside}[i, (p, q-1), j] = \Pr_{em}(r_j, 1) \cdot \text{outside}[i, (p, q), j+1]$
else if $w_j = '\,(\,'$ **then**
$\text{outside}[i, (p, q-1), j] = \text{outside}[i, (p, q), j+1]$
else if $w_j = '\,)\,'$ **then**
$\text{outside}[i, (p, q-1), j] = \Pr_{em}(r_i r_j, j-i+1) \cdot \text{outside}[i, (p, q), j+1]$
end if
if $q = k(p)$ /**rule* $= X \to \gamma w_j \bullet$, i.e. rule is completed in this scanning step*/ **then**
$\text{outside}[i, (p, q-1), j] = \text{outside}[i, (p, q-1), j] \cdot \Pr_{tr}(X \to \gamma w_j, j-i+1)$
end if
return
end procedure

Algorithm 13 Completing outside items

procedure CompleteReverse(i, j, p, q)
Consider $rule = X \to \gamma Y \bullet \delta = \text{ind}^{-1}(p, q) \in \mathcal{R}_{\mathcal{G}_s,\bullet}$ /*i.e. the rule having index (p, q) in our ordering*/
if $q = k(p)$ /*$rule = X \to \gamma Y\bullet$, i.e. rule is completed*/ **then**
  $\texttt{fact} = \Pr_{tr}(X \to \gamma Y, (j-1) - i + 1)$
else
  $\texttt{fact} = 1$
end if
for $k = i$ **to** j **do**
  $\text{outside}[i, (p, q-1), k] = \text{outside}[i, (p, q-1), k] +$
    $\left(\text{outside}[i, (p, q), j] \cdot \left(\sum_{rule_B \in \mathcal{R}_B} \text{inside}[k, \text{ind}(rule_B), j]\right)\right) \cdot \texttt{fact}$
  for $rule_B \in \mathcal{R}_B$ **do**
    $\text{outside}[k, \text{ind}(rule_B), j] = \text{outside}[k, \text{ind}(rule_B), j] +$
      $(\text{outside}[i, (p, q), j] \cdot \text{inside}[i, (p, q-1), k]) \cdot \texttt{fact}$
  end for
end for
return
end procedure

These modified algorithms show how to perform the complete inside computation and – once the inside values are computed – how to calculate the corresponding outside values of all items.

Note that for the sake of simplicity and in order to demonstrate that both algorithms work in either case (length-dependent or not), we relied on the notation proposed in Section 3.3.8.2, this means we used

$$\Pr_{tr}(X \to \gamma, l) = \begin{cases} \Pr_{tr}(X \to \gamma, l) \cdot \frac{1}{c_{\gamma,l}} & \text{if } \mathcal{G}_r \text{ length-dependent,} \\ \Pr_{tr}(X \to \gamma) & \text{else,} \end{cases}$$

and

$$\Pr_{em}(x_1 x_2, l) = \begin{cases} \Pr_{em}(x_1 x_2, l) & \text{if } \mathcal{G}_r \text{ length-dependent,} \\ \Pr_{em}(x_1 x_2) & \text{else,} \end{cases}$$

as well as

$$\Pr_{em}(x, 1) = \Pr_{em}(x).$$

In this context, it should be mentioned that in our algorithms, we are actually using the probability

$$\begin{aligned} &\Pr_{tr}(A \to ({}^{m_s}L)^{m_s}, j - i + 1) \\ &\times \Pr_{em}(x_i x_j, j - i + 1) \\ &\times \Pr_{em}(x_{i+1} x_{j-1}, j - i + 1 - 2) \\ &\times \Pr_{em}(x_{i+2} x_{j-2}, j - i + 1 - 4) \\ &\quad \dots \\ &\times \Pr_{em}(x_{i+(m_s-1)} x_{j-(m_s-1)}, j - i + 1 - 2 \cdot (m_s - 1)) \end{aligned}$$

for the initialization of a helix (of minimum allowed size $m_s := \min_{hel}$) with first base pair i.j, which is adequate under the commonly used assumption that the emission probabilities come from the same distribution. However, going strictly with the formal definition, we would have

to consider the term

$$\begin{aligned}&\Pr_{tr}(A \to ({}^{m_s}L)^{m_s}, j-i+1)\\ \times&\Pr_{em}(x_i x_{i+1} x_{i+2} \dots x_{i+(m_s-1)} x_{j-(m_s-1)} \dots x_{j-2} x_{j-1} x_j, j-i+1),\end{aligned}$$

which means if $\min_{hel} > 1$ is chosen, we would have to derive and use an additional set of emission probabilities for any possible combination of $\min_{hel}$ consecutive base pairs. Nevertheless, this inaccuracy could easily be corrected by modifying our grammar definition such that production $p_9 : A \to ({}^{m_s}L)^{m_s}$ can be simulated by the composition of new productions

$$\begin{aligned}&p_9 : A \to (A_1),\\ &1 : A_1 \to (A_2),\\ &1 : A_2 \to (A_3),\\ &\dots,\\ &1 : A_{m_s-1} \to (L).\end{aligned}$$

Then, our algorithms work perfectly conform with the formal definition.

Finally, note that by factoring in the rule probability $\Pr_{tr}(X \to \gamma, l)$ of production $X \to \gamma \in \mathcal{R}_{\mathcal{G}_s}$ in the last scanning or completion steps of the corresponding items $[i, \mathrm{ind}(X \to \gamma\bullet), j]$, $1 \leqslant i, j \leqslant n+1$, instead of as usually done initially in the prediction steps of the corresponding items $[i, \mathrm{ind}(X \to \bullet\gamma), j]$, this rule probability $\Pr_{tr}(X \to \gamma, l)$ is not incorporated as a factor into the corresponding inside values $[i, \mathrm{ind}(X \to \gamma \bullet \delta), j]$, $1 \leqslant i, j \leqslant n+1$, if $\delta \neq \epsilon$. This means these values are not correctly computed. However, for $\delta = \epsilon$, the inside values of items $[i, \mathrm{ind}(X \to \gamma \bullet \delta), j] = [i, \mathrm{ind}(X \to \gamma\bullet), j]$, $1 \leqslant i, j \leqslant n+1$, are indeed correctly calculated. Hence, the needed traditional inside probabilities $\alpha_X(i, j)$, $1 \leqslant i, j \leqslant n$, can be accurately derived for these items (see Section 6.4.1.3). The same holds for the corresponding outside values.

Last but not least, we observe that the modifications that led to Algorithms 7 and 11 do not imply a significant additional computation effort. Therefore, for a sequence r of size n, there still results cubic time complexity and quadratic memory requirement in the worst-case for the computation of all inside and outside probabilities $\alpha_A(i, j)$ and $\beta_A(i, j)$, $A \in \mathcal{I}_{\mathcal{G}_s}$ and $1 \leqslant i, j \leqslant n$.

7.3.2. Computation of Sampling Probabilities and Structure Sampling

Obviously, the equations defining the needed sampling probabilities for all considered cases as presented in Section 6.4.2.1 depend not only on inside and outside values for the given RNA sequence, but also on probabilities $\Pr(\mathit{rule})$ of production rules $\mathit{rule} \in \mathcal{R}_{\mathcal{G}_s}$ of the underlying SCFG. Thus, in order to obtain the respective sampling probabilities based on the corresponding LSCFG model, besides computing the inside and outside probabilities in a slightly different way as described just before, we additionally have to consider length-dependent rule probabilities (multiplied by corresponding factors) instead of their traditional length-independent counterparts in the respective definitions. This can easily be done by replacing the factors $\Pr(\mathit{rule})$ by $\Pr(\mathit{rule}, l)$, since then the corresponding definitions can be considered in either case (length-dependent or not). Anyway, the corresponding sampling algorithm and the use of the diverse sampling probabilities (derived length-dependently or conventionally) remain the same as described in Section 6.4.2.

In summary, by using the LSCFG approach instead of the corresponding length-independent variant, we can produce a statistical sample of the complete ensemble of all possible structures for a given sequence without significant losses in performance. However, when comparing the results of both SCFG methods, significant differences can be observed, as we will see in the next section.

7.4. Applications and Discussion

The purpose of this section is to explore the benefits and potential drawbacks of enriching the sophisticated SCFG design introduced in Section 6.3 with additional information on the lengths of generated subwords (corresponding to particular RNA substructures).

In principle, the main question is to what extend are the sampling quality and the predictive power of the corresponding sampling variants affected by relying on the more elaborate LSCFG model (with a resulting comparatively huge number of more specific parameters) instead of on the conventional SCFG model (that implies only a rather moderate parameter number). It would also be interesting to see how much the performances of the traditional and the length-dependent SCFG variant (with the more generalized and specialized transition and emission probabilities, respectively) differ from that of the popular PF approach (that employs many hundreds of mostly experimentally obtained thermodynamic parameters). Therefore, we decided to consider a number of meaningful applications in connection with sampling approaches to the generated samples and for any of those applications oppose the results obtained with the proposed LSCFG based sampling approach to corresponding outputs of the simple SCFG variant studied in Chapter 6 and the PF method as described in [DL03][2].

7.4.1. Considered RNA Data and Probabilistic Parameters

In order to obtain an adequate basis for the investigations that will be performed within this section, we took the same sets of real world RNA data as were used for the corresponding applications in Section 6.6:

- First, a (very rich) tRNA database (of 2163 distinct structures with lengths in $[64, 93]$) obtained from [SHB+98].
- Second, a (not quite so rich) 5S rRNA database (of 1149 distinct sequences with lengths in $[102, 135]$) retrieved from [SBEB02].
- And last but not least, a (rather sparse) mixed structural database (of 151 distinct RNA molecules with lengths in $[23, 568]$) as collected in [DWB06].

Note that the latter will again be denoted by S-151Rfam database and is ought to illustrate quality differences of the corresponding results compared to the rich (and pure) tRNA and 5S rRNA data sets. It needs to be mentioned that all molecules in the considered benchmark sets are shorter than 1000 nucleotides, such that the probabilities of the last interval (of the 105 reasonable length intervals presented in Section 7.2) may not influence our results.

Anyway, Table 7.1 shows that quite different numbers of relevant length-dependent (rule and emission) probabilities result when training our grammar on the three different data sets, respectively. However, although the numbers of relevant grammar parameters are unsurprisingly to a large extend greater when considering the LSCFG model rather than the traditional length-independent variant, they are indeed of a considerably smaller (in case of tRNAs and 5S rRNAs) or at least only of a similar (in case of the S-151Rfam data) order of magnitude than the numbers of energy parameters employed in standard thermodynamic models. Hence, if statistical parameter learning makes sense in connection with free energy approaches (see Section 3.3.10), then it should also yield reasonable results in case of LSCFG based probabilistic methods. Motivated by this assumption, we decided to start our examinations in the next section by considering one of the most intuitive applications in connection with statistical sampling methods that is of great practical interest.

[2] It should be noted that for our examinations, we will again rely on our own version of `Sfold`'s sampling procedure (as mentioned in Section 6.6.2).

Training data	Model	Structural Constraints	num_{tr}	num_{em}^{unp}	num_{em}^{bp}
tRNA	SCFG	$min_{HL} \in \{1,3\}, min_{hel} = 1$	28	4	15
		$min_{HL} \in \{1,3\}, min_{hel} = 2$	27	4	14
	LSCFG	$min_{HL} = 1, min_{hel} = 1$	334	4	162
		$min_{HL} = 1, min_{hel} = 2$	281	4	155
		$min_{HL} = 3, min_{hel} = 1$	332	4	162
		$min_{HL} = 3, min_{hel} = 2$	279	4	155
5S rRNA	SCFG	$min_{HL} \in \{1,3\}, min_{hel} \in \{1,2\}$	28	4	16
	LSCFG	$min_{HL} = 1, min_{hel} = 1$	392	4	572
		$min_{HL} = 1, min_{hel} = 2$	357	4	565
		$min_{HL} = 3, min_{hel} = 1$	390	4	572
		$min_{HL} = 3, min_{hel} = 2$	355	4	565
S-151Rfam	SCFG	$min_{HL} \in \{1,3\}, min_{hel} \in \{1,2\}$	29	4	6
	LSCFG	$min_{HL} = 1, min_{hel} = 1$	1171	4	477
		$min_{HL} = 1, min_{hel} = 2$	1055	4	446
		$min_{HL} = 3, min_{hel} = 1$	1155	4	470
		$min_{HL} = 3, min_{hel} = 2$	1022	4	434

Table 7.1.: **Comparison of relevant grammar parameters.** Tabulated values are the numbers of relevant parameters (transition and emission probabilities being greater than zero). They are obtained from training the respective database (in the traditional or length-dependent way). Here, num_{tr} denotes the number of relevant transition probabilities. Accordingly, num_{em}^{unp} and num_{em}^{bp} denote the numbers of relevant emission probabilities of unpaired bases and base pairs, respectively.

7.4.2. Probability Profiling for Specific Loop Types

For a first comparison of the sampling results obtained with the presented LSCFG approach to those for the PF variant, we decided to consider the corresponding probability profiles for the *Escherichia coli* $tRNA^{Ala}$ sequence (from Example 2.1.1). They are shown in Figure 7.1. It is obvious that the statistical samples generated by the (L)SCFG approach are significantly more accurate than those obtained with the PFs. Furthermore, comparing the plots in Figure 7.1 to those in Figure 6.1 that were computed without considering length-dependency, we see that the sampling results can indeed be improved by incorporating additional length information into the underlying SCFG model; the correct cloverleaf structure of the considered tRNA is almost exactly reached in all sampled structures.

Nevertheless, before we proceed with applications of the considered sampling approaches to RNA structure prediction, we first want to discuss some important results with respect to the quality of probabilistic structure models underlying the proposed LSCFG based sampling method, in particular the ones induced by the three considered RNA classes, respectively.

7.4.3. The Problem of Overfitting and the Lack of Generalization

Analogous to Section 6.6.2.3, we will address two possible issues of our sophisticated LSCFG in connection with this study: the problem of overfitting and the lack of generalization.

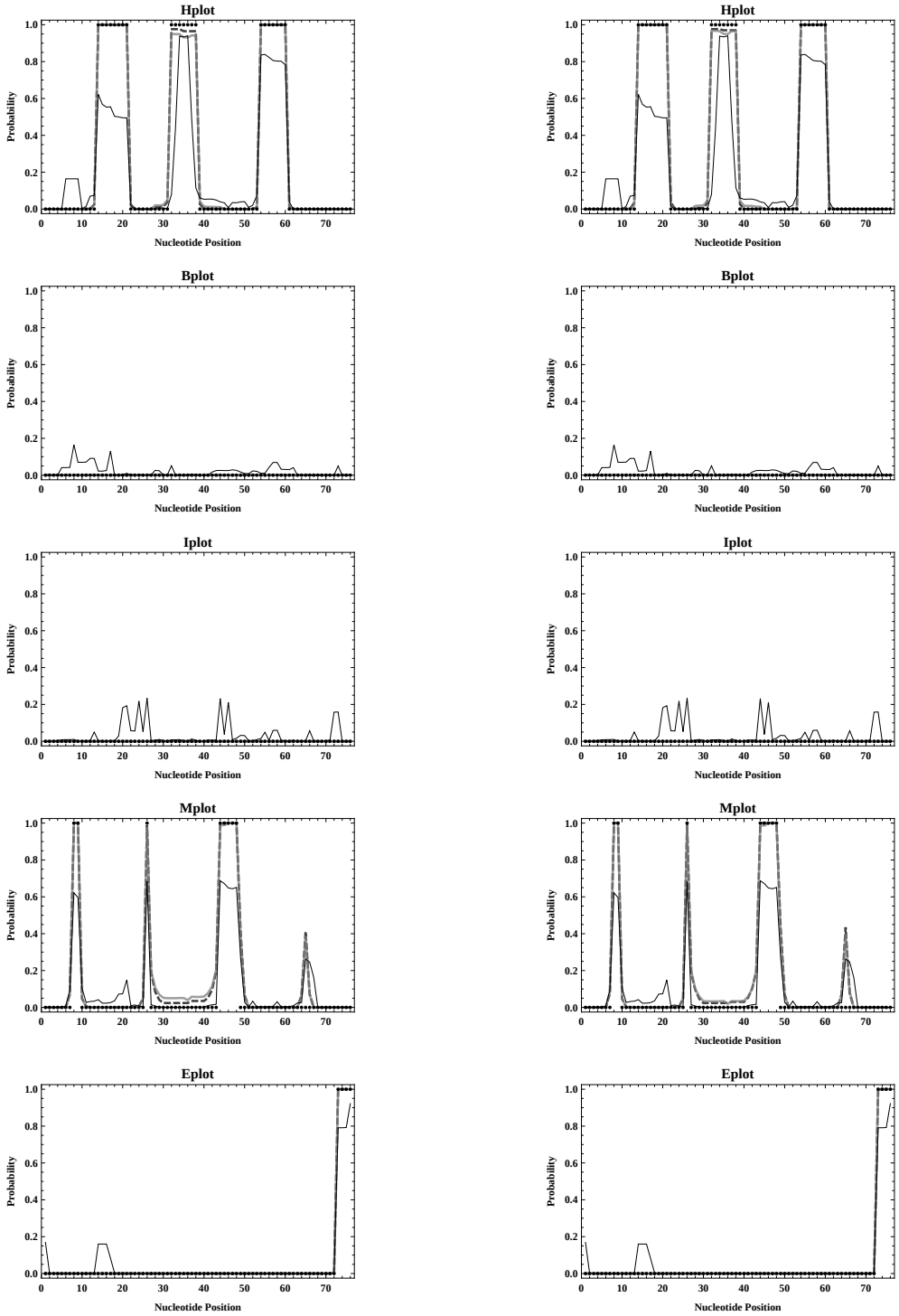


Figure 7.1.: **Comparison of profiling results for PF and LSCFG approach.** Figure shows loop profiles for *E.coli* $tRNA^{Ala}$ corresponding to those presented in Figure 6.1, but obtained with the PF approach and the length-dependent SCFG variant.

Approach	min_{hel}	num_d	c_d	c_{MF}	c_{CL}	num_{MF}
PF	0	36	8333.33	94085	3331	6
	1	34	8823.53	87785	5338	6
	2	35	8571.43	96083	2745	6
	3	37	8108.11	95332	4492	6
	4	30	10000.	107881	9967	6
	5	29	10344.8	111716	20875	3
	6	33	9090.91	102788	49733	2
	7	27	11111.1	94859	94859	0
SCFG	0	858	349.65	26341	14114	5
	1	916	327.511	22643	15596	4
	2	915	327.869	21258	13912	4
	3	895	335.196	20175	16207	2
	4	914	328.228	19828	17784	2
	5	844	355.45	20560	20560	0
	6	747	401.606	34753	34753	0
	7	658	455.927	59644	59644	0
LSCFG	0	28	10714.3	92727	92727	0
	1	28	10714.3	91276	91276	0
	2	25	12000.	88660	88660	0
	3	27	11111.1	94323	94323	0
	4	27	11111.1	94536	94536	0
	5	27	11111.1	100720	100720	0
	6	27	11111.1	107157	107157	0
	7	26	11538.5	115788	115788	0

Table 7.2.: **(Additional) results derived from random data sets.** Tabulated values for the LSCFG approach were computed from the same random data sets and in the same way as those for the PF and traditional SCFG approach (which have already been presented in Table 6.5).

With respect to the latter, it seems important to mention that the profiling results for *Escherichia coli* $tRNA^{Ala}$ validate two obvious assumptions: First, if our LSCFG is trained on trusted tRNAs only, it should inevitably produce the typical tRNA cloverleaf shape more often than the alternative PF variant that is not suited to a specific class of RNA structures. Second, as the additional consideration of length-dependencies yields more specialized probabilities for the distinct structural motifs and tRNA molecules naturally show a low structural variety, the LSCFG based profiles should inherently show the cloverleaf structure more explicitly.

Consequently, it is likely that the higher accuracy reached by the LSCFG model could be an artefact caused by lack of generalization of the underlying stochastic structure model. To get evidence of the correctness of this assumption, we took the random sequence sets from Section 6.6.2.3[3] and applied the different sampling approaches. The results are collected in Table 7.2.

Undoubtably, the presented statistics demonstrate that the LSCFG variant mainly samples cloverleaf structures, even if the signal towards cloverleaf is low or does actually not exist

[3]For any fixed value of min_{hel}, the corresponding set has been generated by randomly creating secondary structures (with corresponding sequences) having the cloverleaf shape, where all four helices (the stem and the three adjacent helices of the multiloop) are formed by exactly min_{hel} consecutive (canonical) base pairs.

($\min_{\mathrm{hel}} = 0$ means completely random sequences, that is no signal). This yields the assumption that incorporating length-dependencies into the underlying sophisticated SCFG model in fact causes lack of generalization, as the cloverleaf shape is always preferred over others, regardless of the signal induced by the actual sequence composition. Hence, there is some reason to believe that the accuracy gain of the LSCFG sampling approach (at least for tRNA profiling) is due to the high degree of specialization of the underlying stochastic structure model (very explicitly tailored to a certain shape), which bares an undesirable lack of generalization (to possible but usually less likely other shapes). Since we most likely observe such effects in connection with tRNA and its invariant cloverleaf shape, we skipped similar investigations for the other cases.

Nevertheless, in order to investigate if overfitting may be a problem for the subsequent examinations, that is to see if our different data sets are rich enough to reliably derive the parameters of our grammar even in the length-dependent case, we performed the following experiments (also similar to Section 6.6.2.3): For each of the three considered structural RNA databases, we selected a random 90% portion of the original database. We then re-estimated the probabilities of all grammar rules (for any of the previously chosen length intervals, respectively). Since the number of feasible structures that can be considered for training is reduced by prohibiting small hairpin loops and isolated base pairs, we decided to rely on the most realistic restrictions of $\min_{\mathrm{hel}} = 2$ and $\min_{\mathrm{HL}} = 3$ for our SCFG $\mathcal{G}_s$ in order to obtain the potentially most meaningful results.

The corresponding re-estimation process was iterated 100 times for any database, resulting in a sample of 100 parameter sets, respectively, each of them consisting of exactly $\mathtt{card}(\mathcal{R}_{\mathcal{G}_s})$ sets of length-dependent probabilities $p_i(I)$ for the distinct length intervals I rather than of one single (conventional, that is length-independent) probability value p_i, $1 \leqslant i \leqslant \mathtt{card}(\mathcal{R}_{\mathcal{G}_s})$. Therefore, for each of the distinguished length-dependent grammar parameters $p_i(I)$, we determined its variance along the constructed sample of size 100 and subsequently computed the maximum variance (observed for a particular length interval I) among all variances $\mathbb{V}[p_i(I)]$ implied by production rule f_i, for $1 \leqslant i \leqslant \mathtt{card}(\mathcal{R}_{\mathcal{G}_s})$. Formally, for each set of length-dependent probabilities corresponding to grammar parameter p_i, $1 \leqslant i \leqslant \mathtt{card}(\mathcal{R}_{\mathcal{G}_s})$, we calculated $\max_I \mathbb{V}[p_i(I)]$. The resulting values are collected in Table 7.3.

Note that the variances 0 in most cases result for production rules finishing the generation of unpaired regions (for example $p_8 : C \to Z$ or $p_{28} : U \to \epsilon$), since those can only produce words of one particular length (1 or 0), whereas longer words (unpaired regions) are generated by the corresponding alternative productions with same left-hand side (for example $p_7 : C \to ZC$ or $p_{27} : U \to ZU$), and the weights on the production rules must indeed sum up to unity for any considered length interval. Thus, since we use unary intervals for lengths 0 and 1, respectively, for any production ending a run of unpaired bases, a probability of 1 is predetermined, yielding variance 0. For basically the same reason, there must result a variance of 0 for production $p_{29} : Z \to \circ$, that is this observation is due to the fact that this rule unexceptionally generates words of length 1 (an arbitrary unpaired base) and there exist no other alternatives for the corresponding premise implying words of that particular length.

However, all the other (maximum) variances presented in Table 7.3 (at least for tRNAs and 5S rRNAs) are rather small, too. Therefore, we may assume that overfitting is not really an issue in connection with our sophisticated SCFG and the training sets used (at least for the rich tRNA and 5S rRNA data), even in the case of length-dependent parameter estimation procedures. It remains to mention that the tabulated (maximum) variances derived for the considered length-dependent grammar parameters are in most cases indeed larger than the corresponding variances for the conventional parameters which do not depend on the lengths of generated subwords, as can be observed by comparing Table 7.3 to Table 6.6.

$\max_I \mathbb{V}[\cdot(I)]$	tRNA	5S rRNA	S-151Rfam
p_1	1.1372×10^{-4}	4.6414×10^{-5}	2.1343×10^{-2}
p_2	7.1533×10^{-5}	7.0953×10^{-5}	4.8821×10^{-3}
p_3	7.0888×10^{-7}	2.4405×10^{-5}	1.8768×10^{-3}
p_4	2.0229×10^{-7}	7.7689×10^{-6}	1.0272×10^{-3}
p_5	3.5269×10^{-5}	2.5606×10^{-5}	2.4133×10^{-3}
p_6	4.9101×10^{-6}	6.4274×10^{-6}	4.9589×10^{-3}
p_7	1.1616×10^{-5}	3.0681×10^{-5}	1.7759×10^{-4}
p_8	0	0	0
p_9	4.9153×10^{-6}	3.5895×10^{-5}	6.0461×10^{-3}
p_{10}	5.5523×10^{-6}	1.5978×10^{-5}	2.9525×10^{-3}
p_{11}	2.6480×10^{-6}	5.7427×10^{-6}	8.8191×10^{-4}
p_{12}	6.1203×10^{-6}	1.5467×10^{-5}	1.7275×10^{-3}
p_{13}	1.6234×10^{-7}	6.3548×10^{-6}	3.1334×10^{-4}
p_{14}	2.9152×10^{-6}	3.2344×10^{-6}	6.5392×10^{-5}
p_{15}	3.1928×10^{-6}	1.0465×10^{-4}	2.0547×10^{-3}
p_{16}	1.9113×10^{-6}	6.4819×10^{-6}	1.1604×10^{-4}
p_{17}	0	0	0
p_{18}	0	1.0346×10^{-4}	1.6601×10^{-3}
p_{19}	0	8.9041×10^{-5}	2.0498×10^{-3}
p_{20}	1.8388×10^{-3}	1.1285×10^{-4}	9.3347×10^{-3}
p_{21}	4.1771×10^{-5}	6.9182×10^{-7}	7.3819×10^{-5}
p_{22}	0	0	0
p_{23}	9.5068×10^{-5}	4.1479×10^{-5}	3.6034×10^{-2}
p_{24}	5.1666×10^{-5}	6.1313×10^{-4}	5.1346×10^{-2}
p_{25}	1.6458×10^{-5}	0	1.7848×10^{-3}
p_{26}	1.2797×10^{-6}	1.7441×10^{-4}	1.6096×10^{-2}
p_{27}	8.0792×10^{-7}	4.0028×10^{-6}	6.0669×10^{-4}
p_{28}	0	0	0
p_{29}	0	0	0

Table 7.3.: **Truncated maximum variances of any set of grammar parameters (transition probabilities) for different length intervals.** Values were derived from 100 iterations of (length-dependently) training our SCFG $\mathcal{G}_s$ on random subsets containing 90 percent of the original data, respectively, under the assumption of $\min_{HL} = 3$ and $\min_{hel} = 2$.

7.4.4. Prediction Accuracy – Sensitivity and PPV

To investigate how the quality of predictions changes when using the (length-dependent) SCFG approach for computing the sampling probabilities, we once more consider the common accuracy measures sensitivity and PPV (see Section 6.5). In fact, we decided to perform a suitable k-fold cross-validation for any of our three different RNA databases in order to assess the differences in the predictive accuracy of sample sets generated according to either approach.

For the sake of simplicity, we used the same partitions of the comprehensive tRNA and 5S rRNA databases into $k = 10$ and of the mixed S-151Rfam database into $k = 2$ approximately equal-sized folds as in Sections 6.6.2.4 and 6.6.2.5 for deriving the corresponding k-fold cross-validations results, respectively. Accordingly, for any sequence, we only sampled a set of

Approach	Parameters	MF struct.		MEA struct.		Centroid	
		Sens.	PPV	Sens.	PPV	Sens.	PPV
PF	$\max_{bulge} = 30$	0.6565	0.5890	0.6434	0.6035	0.6159	0.6344
SCFG	$\min_{HL} = 1, \min_{hel} = 1$	0.7791	0.8445	0.7324	0.8939	0.6754	0.9158
	$\min_{HL} = 1, \min_{hel} = 2$	0.8004	0.8457	0.7685	0.8878	0.7113	0.9123
	$\min_{HL} = 3, \min_{hel} = 1$	0.8545	0.8517	0.7848	0.9021	0.7304	0.9213
	$\min_{HL} = 3, \min_{hel} = 2$	0.8677	0.8593	0.8182	0.8953	0.7713	0.9168
LSCFG	$\min_{HL} = 1, \min_{hel} = 1$	0.8542	0.9535	0.8335	0.9736	0.8250	0.9783
	$\min_{HL} = 1, \min_{hel} = 2$	0.8530	0.9502	0.8518	0.9613	0.8435	0.9657
	$\min_{HL} = 3, \min_{hel} = 1$	0.8602	0.9526	0.8371	0.9733	0.8278	0.9775
	$\min_{HL} = 3, \min_{hel} = 2$	0.8575	0.9494	0.8562	0.9609	0.8477	0.9651

Table 7.4.: **Sensitivity and PPV values for our tRNA database.** Results were computed by 10-fold cross-validation procedures, using sample size 1000.

Approach	Parameters	MF struct.		MEA struct.		Centroid	
		Sens.	PPV	Sens.	PPV	Sens.	PPV
PF	$\max_{bulge} = 30$	0.5897	0.5806	0.6015	0.6191	0.5789	0.6508
SCFG	$\min_{HL} = 1, \min_{hel} = 1$	0.4251	0.5362	0.3403	0.6967	0.2689	0.8044
	$\min_{HL} = 1, \min_{hel} = 2$	0.4542	0.5435	0.3638	0.6901	0.2727	0.8069
	$\min_{HL} = 3, \min_{hel} = 1$	0.4728	0.5290	0.3544	0.7033	0.2764	0.8091
	$\min_{HL} = 3, \min_{hel} = 2$	0.5167	0.5577	0.3860	0.7010	0.2846	0.8140
LSCFG	$\min_{HL} = 1, \min_{hel} = 1$	0.8996	0.9408	0.8959	0.9513	0.8873	0.9574
	$\min_{HL} = 1, \min_{hel} = 2$	0.8726	0.9239	0.8714	0.9280	0.8673	0.9333
	$\min_{HL} = 3, \min_{hel} = 1$	0.8992	0.9405	0.8958	0.9509	0.8863	0.9568
	$\min_{HL} = 3, \min_{hel} = 2$	0.8721	0.9231	0.8712	0.9276	0.8667	0.9330

Table 7.5.: **Sensitivity and PPV values for our 5S rRNA database.** Results were computed by 10-fold cross-validation procedures, using sample size 1000.

Approach	Parameters	MF struct.		MEA struct.		Centroid	
		Sens.	PPV	Sens.	PPV	Sens.	PPV
PF	$\max_{bulge} = 30$	0.6652	0.5188	0.6633	0.5450	0.6437	0.5799
SCFG	$\min_{HL} = 1, \min_{hel} = 1$	0.4433	0.5447	0.3815	0.7386	0.3235	0.7749
	$\min_{HL} = 1, \min_{hel} = 2$	0.4894	0.5551	0.4263	0.7181	0.3474	0.7743
	$\min_{HL} = 3, \min_{hel} = 1$	0.4852	0.5948	0.3935	0.7426	0.3352	0.7825
	$\min_{HL} = 3, \min_{hel} = 2$	0.5171	0.5661	0.4342	0.7228	0.3588	0.7683
LSCFG	$\min_{HL} = 1, \min_{hel} = 1$	0.1815	0.5422	0.1390	0.7523	0.1251	0.8003
	$\min_{HL} = 1, \min_{hel} = 2$	0.1646	0.5322	0.1276	0.6706	0.1099	0.7114
	$\min_{HL} = 3, \min_{hel} = 1$	0.1761	0.5354	0.1396	0.7614	0.1238	0.8039
	$\min_{HL} = 3, \min_{hel} = 2$	0.1528	0.5023	0.1230	0.6634	0.1094	0.7118

Table 7.6.: **Sensitivity and PPV values for the mixed S-151Rfam database.** Results were computed by two-fold cross-validation procedures, using the same folds as in [DWB06] and sample size 1000.

1000 structures with the LSCFG approach and then applied different principles to obtain a corresponding structure prediction, specifically the ones introduced in Section 6.5. All corresponding sensitivity and PPV measures are collected in Tables 7.4 to 7.6.

Obviously, the results for tRNAs and 5S rRNAs lead to the conclusion that by using the LSCFG approach for statistical sampling, a significantly higher predictive accuracy can be reached than by sampling based on PFs. Moreover, we immediately observe that the incorporation of length-dependencies into the SCFG approach can have a positive impact on the resulting prediction accuracy: Although the predictions for the longer and thus more variant 5S rRNAs are less accurate than for the shorter tRNAs when using the conventional SCFG approach, the consideration of length-dependent probabilities for the production rules (obtained by training them on real world data) makes the underlying SCFG model explicit enough to handle the larger variety of structure motifs and guarantees high quality prediction results.

However, as we expected, for the S-151Rfam database, the more specialized LSCFG approach yields the worst prediction results, whereas the highest accuracy for this mixed data set is reached with PF sampling that relies on thermodynamic parameters and is not suited for a particular RNA type. In fact, this observation is strongly related to the fact that the S-151Rfam data set is rather sparse and additionally contains structures that belong to distinct RNA types that obey to different structural properties, such that it can not be considered an optimal training basis. This problem is considerably increased by the partitioning of (the already rather few) data points according to the various interval lengths for our LSCFG variant, which is actually in accordance with the worse results for the S-151Rfam set compared to the rich and pure tRNA and 5S rRNA sets as presented in Table 7.3.

Altogether, we can assume that if a reasonable RNA secondary structure database (containing a sufficiently large number of known structures that are of the same or similar RNA types) can be used for estimating the parameters of the underlying LSCFG model, then even for RNA molecules with a high variability of typical structural features (for which the traditional SCFG method lacks the ability to identify the typical shape of the respective family by considering the estimated length-independent parameters), the predictive results might be of high quality and potentially manage to outperform predictions obtained with the PF variant that is based on the competing free energy approach.

All these observations may be affirmed on the basis of more reliable accuracy results found from corresponding ROC curves for γ_{t-o}-MEA and γ_{t-o}-centroid structures as introduced in Sections 6.5.2 and 6.5.3. In fact, for

$$\gamma_{t-o} \in \{1.25^k \mid -12 \leqslant k \leqslant -1\} \cup \{2^k \mid 0 \leqslant k \leqslant 12\},$$

the respective estimated AUC values observed for any of the considered databases are reported in Tables 7.7 to 7.9. Plots of some of the respective ROC curves can be found in Figures C.1 to C.3 of Section C). As intended, the provided AUC values allow for a more reliable comparison of the accuracies that can be reached by either approach on the basis of MEA and centroid structures for the produced samples, respectively, but eventually yield basically the same conclusions.

Finally, recall that according to the definitions of sensitivity and PPV, these two accuracy measures depend only on the numbers of correctly and incorrectly predicted base pairs (compared to the native structure), whereas biologists are usually much more interested in getting reliable (abstract) shape predictions. For this reason, a corresponding discussion will follow in the next section.

Approach	Parameters	MEA struct.	Centroid
PF	$\max_{bulge} = 30$	0.482435	0.526743
SCFG	$\min_{HL} = 1, \min_{hel} = 1$	0.828522	0.833894
	$\min_{HL} = 1, \min_{hel} = 2$	0.830787	0.839843
	$\min_{HL} = 3, \min_{hel} = 1$	0.855406	0.861640
	$\min_{HL} = 3, \min_{hel} = 2$	0.857251	0.867135
LSCFG	$\min_{HL} = 1, \min_{hel} = 1$	0.936285	0.919736
	$\min_{HL} = 1, \min_{hel} = 2$	0.916900	0.910218
	$\min_{HL} = 3, \min_{hel} = 1$	0.936337	0.920387
	$\min_{HL} = 3, \min_{hel} = 2$	0.916641	0.910321

Table 7.7.: **AUC values for our tRNA database.** Results were computed by 10-fold cross-validation procedures, using sample size 1000.

Approach	Parameters	MEA struct.	Centroid
PF	$\max_{bulge} = 30$	0.481019	0.520171
SCFG	$\min_{HL} = 1, \min_{hel} = 1$	0.409278	0.408549
	$\min_{HL} = 1, \min_{hel} = 2$	0.417286	0.418584
	$\min_{HL} = 3, \min_{hel} = 1$	0.419116	0.417095
	$\min_{HL} = 3, \min_{hel} = 2$	0.433954	0.431642
LSCFG	$\min_{HL} = 1, \min_{hel} = 1$	0.914801	0.918933
	$\min_{HL} = 1, \min_{hel} = 2$	0.854520	0.863009
	$\min_{HL} = 3, \min_{hel} = 1$	0.914114	0.918600
	$\min_{HL} = 3, \min_{hel} = 2$	0.853399	0.862744

Table 7.8.: **AUC values for our 5S rRNA database.** Results were computed by 10-fold cross-validation procedures, using sample size 1000.

Approach	Parameters	MEA struct.	Centroid
PF	$\max_{bulge} = 30$	0.450688	0.497350
SCFG	$\min_{HL} = 1, \min_{hel} = 1$	0.499491	0.507125
	$\min_{HL} = 1, \min_{hel} = 2$	0.506602	0.509403
	$\min_{HL} = 3, \min_{hel} = 1$	0.507454	0.512327
	$\min_{HL} = 3, \min_{hel} = 2$	0.508762	0.514958
LSCFG	$\min_{HL} = 1, \min_{hel} = 1$	0.270606	0.269354
	$\min_{HL} = 1, \min_{hel} = 2$	0.206630	0.208092
	$\min_{HL} = 3, \min_{hel} = 1$	0.271388	0.266478
	$\min_{HL} = 3, \min_{hel} = 2$	0.205790	0.209557

Table 7.9.: **AUC values for the mixed S-151Rfam database.** Results were computed by two-fold cross-validation procedures, using the same folds as in [DWB06] and sample size 1000.

Value	Variant	Parameters	Shape Level					
			0	1	2	3	4	5
CSP_{freq} (MF)	PF	$max_{bulge} = 30$	0.0633	0.1216	0.2071	0.2117	0.2639	0.3694
	SCFG	$min_{HL} = 3, min_{hel} = 1$	0.2450	0.4448	0.6417	0.6417	0.6422	0.7356
	LSCFG	$min_{HL} = 3, min_{hel} = 1$	0.3440	0.5137	0.6805	0.6805	0.6810	0.7628
CSP_{freq} (MEA)	PF	$max_{bulge} = 30$	0.0416	0.1049	0.1923	0.1960	0.2496	0.3559
	SCFG	$min_{HL} = 3, min_{hel} = 2$	0.1008	0.2917	0.5525	0.5525	0.5543	0.6241
	LSCFG	$min_{HL} = 3, min_{hel} = 2$	0.2053	0.4115	0.6958	0.6958	0.6963	0.7869
CSP_{freq} (Centroid)	PF	$max_{bulge} = 30$	0.0264	0.0800	0.1595	0.1627	0.1932	0.2677
	SCFG	$min_{HL} = 3, min_{hel} = 2$	0.0758	0.2150	0.4563	0.4563	0.4568	0.5003
	LSCFG	$min_{HL} = 3, min_{hel} = 2$	0.1956	0.3824	0.6426	0.6426	0.6431	0.7240
CSO_{freq}	PF	$max_{bulge} = 30$	0.5196	0.6740	0.8160	0.8239	0.8798	0.9556
	SCFG	$min_{HL} = 3, min_{hel} = 1$	0.7148	0.9459	0.9875	0.9880	0.9885	0.9991
	LSCFG	$min_{HL} = 3, min_{hel} = 1$	0.8391	0.9441	0.9778	0.9783	0.9783	0.9986
CS_{num}	PF	$max_{bulge} = 30$	21.073	58.200	136.67	140.63	205.54	328.56
	SCFG	$min_{HL} = 3, min_{hel} = 2$	34.898	173.73	513.05	513.06	513.08	595.26
	LSCFG	$min_{HL} = 3, min_{hel} = 2$	104.09	300.04	730.09	730.09	730.54	826.21
DS_{num}	PF	$max_{bulge} = 30$	355.32	130.22	81.796	33.125	22.585	4.8848
	SCFG	$min_{HL} = 3, min_{hel} = 2$	592.84	103.04	18.921	18.921	18.921	12.053
	LSCFG	$min_{HL} = 3, min_{hel} = 2$	126.84	8.2815	2.7296	2.7296	2.7296	2.3869

Table 7.10.: **Comparison of sampling quality for tRNAs.** Table contains results related to the shapes of selected predictions and sampled structures, obtained from our tRNA database. They were computed by 10-fold cross-validation procedures, using sample size 1000.

7.4.5. Sampling Quality – Specific Values Related to Shapes

The proclaimed aim of this section is to compare sampling results generated by the PF, SCFG and LSCFG approaches with respect to the abstraction level of shapes of generated structures. Particularly, we will consider the same reasonable specific values related to the abstract shapes of selected predictions and sampled structures as introduced in Section 6.6.2.5 in order to obtain further proof of the high quality of sample sets generated by the proposed LSCFG approach.

The respective results are all collected in Tables C.1 to C.6 in Section C. Some of the most interesting ones are additionally displayed in Tables 7.10 to 7.12. Note that all these specific values have been calculated from the predicted structures and the corresponding sample sets that were derived for the calculation of the sensitivity and PPV measures in the last section.

As regards the considered tRNAs and 5S rRNAs, the predicted shape is in most cases significantly more often equal to the correct one when using the SCFG approach (length-dependent or not) instead of the PF variant. That is, the frequency of correct structure predictions (CSP_{freq}) is often higher when using the sophisticated SCFG instead of PFs, especially when length-dependence is considered. Moreover, the statistical samples generated with either of the two different SCFG approaches generally contain the correct shapes considerably more often than those obtained with the PF method, that is are more accurate as regards the frequency of correct structure occurrences (CSO_{freq}). Notably, again the best results are obtained with the LSCFG devised in this chapter (see Tables 7.10 and 7.11).

Value	Variant	Parameters	Shape Level					
			0	1	2	3	4	5
CSP_{freq}	PF	$max_{bulge} = 30$	0.0000	0.0009	0.0078	0.0513	0.0261	0.6353
(MF)	SCFG	$min_{HL} = 3, min_{hel} = 2$	0.0009	0.0096	0.0244	0.0609	0.1027	0.8207
	LSCFG	$min_{HL} = 1, min_{hel} = 1$	0.2002	0.4239	0.4700	0.4857	0.9426	0.9861
CSP_{freq}	PF	$max_{bulge} = 30$	0.0000	0.0052	0.0139	0.0835	0.0696	0.6640
(MEA)	SCFG	$min_{HL} = 3, min_{hel} = 2$	0.0000	0.0009	0.0009	0.0035	0.0557	0.5387
	LSCFG	$min_{HL} = 3, min_{hel} = 1$	0.1062	0.4065	0.4456	0.4535	0.8990	0.9835
CSP_{freq}	PF	$max_{bulge} = 30$	0.0000	0.0026	0.0104	0.0775	0.0731	0.7214
(Centroid)	SCFG	$min_{HL} = 3, min_{hel} = 2$	0.0000	0.0000	0.0000	0.0009	0.0139	0.1549
	LSCFG	$min_{HL} = 1, min_{hel} = 1$	0.0966	0.2916	0.3238	0.3316	0.8703	0.9686
CSO_{freq}	PF	$max_{bulge} = 30$	0.0009	0.1662	0.3063	0.7580	0.6883	0.9817
	SCFG	$min_{HL} = 3, min_{hel} = 2$	0.0026	0.4509	0.6372	0.9904	0.9974	0.9991
	LSCFG	$min_{HL} = 1, min_{hel} = 1$	0.6258	0.8912	0.9295	0.9504	0.9948	1.0000
CS_{num}	PF	$max_{bulge} = 30$	0.0009	0.7571	3.4207	36.641	30.288	600.35
	SCFG	$min_{HL} = 3, min_{hel} = 2$	0.0026	1.3795	3.1949	36.673	71.080	609.58
	LSCFG	$min_{HL} = 3, min_{hel} = 1$	42.962	347.97	422.19	457.71	875.13	983.67
DS_{num}	PF	$max_{bulge} = 30$	710.75	333.72	237.71	93.335	63.661	7.0951
	SCFG	$min_{HL} = 3, min_{hel} = 2$	999.68	885.81	762.67	239.28	123.91	13.558
	LSCFG	$min_{HL} = 1, min_{hel} = 2$	148.01	10.076	8.5355	4.4627	3.5160	1.1297

Table 7.11.: **Comparison of sampling quality for 5S rRNAs.** Table contains results related to the shapes of selected predictions and sampled structures, obtained from our 5S rRNA database. They were computed by 10-fold cross-validation procedures, using sample size 1000.

Furthermore, for tRNAs and 5Sr RNAs, the observed averaged number of correct shapes in a sample set (CS_{num}) is in all cases to a large extend greater when using the LSCFG approach than when using the length-independent variant or the PF method. However, due to these observations it is not surprising that the observed averaged number of different shapes in a sample (DS_{num}) is always significantly smaller when using the LSCFG approach rather than the PF and especially the traditional length-independent SCFG variant (for which the by far highest diversity within the sample set can be reached). This means by incorporating additional information on fragment lengths into the underlying sophisticated SCFG model, a higher predictive accuracy with respect to the shapes of generated structures (on all abstraction levels) can be reached, at the cost of a lower variability of the generated samples.

However, the results for the mixed S-151Rfam data set presented in Table 7.12 show a completely different picture. Most importantly, the considered specific values related to shapes are basically in all cases better when length-dependencies are *not* considered, that is when sticking to the simple SCFG model. This actually resembles the observations made in the last section for the sensitivity and PPV measures and hence provides additional evidence that incorporating length-dependency into a SCFG model for RNA secondary structures results in a much stronger dependence on the availability of a rich and pure training set.

Value	Variant	Parameters	Shape Level					
			0	1	2	3	4	5
CSP_{freq} (MF)	PF	$max_{bulge} = 30$	0.0661	0.1255	0.1586	0.2050	0.2183	0.4834
	SCFG	$min_{HL} = 3, min_{hel} = 2$	0.0530	0.1258	0.1522	0.1788	0.1985	0.4240
	LSCFG	$min_{HL} = 3, min_{hel} = 1$	0.0199	0.0532	0.0664	0.0730	0.0995	0.3179
CSP_{freq} (MEA)	PF	$max_{bulge} = 30$	0.0660	0.1123	0.1453	0.1984	0.2051	0.4902
	SCFG	$min_{HL} = 1, min_{hel} = 2$	0.0264	0.1193	0.1391	0.1523	0.1789	0.4239
	LSCFG	$min_{HL} = 3, min_{hel} = 1$	0.0132	0.0397	0.0530	0.0596	0.0794	0.2118
CSP_{freq} (Centroid)	PF	$max_{bulge} = 30$	0.0793	0.1321	0.1653	0.1917	0.2449	0.5100
	SCFG	$min_{HL} = 3, min_{hel} = 2$	0.0197	0.0927	0.1125	0.1390	0.1391	0.3577
	LSCFG	$min_{HL} = 3, min_{hel} = 1$	0.0066	0.0397	0.0530	0.0596	0.0728	0.1722
CSO_{freq}	PF	$max_{bulge} = 30$	0.3638	0.4433	0.4766	0.5231	0.6488	0.7947
	SCFG	$min_{HL} = 1, min_{hel} = 2$	0.2717	0.5630	0.6158	0.7284	0.8079	0.9605
	LSCFG	$min_{HL} = 1, min_{hel} = 1$	0.0463	0.2518	0.4041	0.5496	0.5960	0.8408
CS_{num}	PF	$max_{bulge} = 30$	40.390	88.886	121.55	158.32	195.83	453.58
	SCFG	$min_{HL} = 3, min_{hel} = 2$	15.059	63.707	83.965	125.82	142.99	391.39
	LSCFG	$min_{HL} = 1, min_{hel} = 1$	4.6818	30.691	44.362	62.552	92.031	305.66
DS_{num}	PF	$max_{bulge} = 30$	540.74	304.36	255.40	150.89	117.24	18.795
	SCFG	$min_{HL} = 3, min_{hel} = 2$	840.03	522.53	452.04	307.61	273.92	77.536
	LSCFG	$min_{HL} = 1, min_{hel} = 2$	568.66	172.46	143.60	72.662	57.327	9.5317

Table 7.12.: **Comparison of sampling quality for the S-151Rfam set.** Table contains results related to the shapes of selected predictions and sampled structures, obtained from the S-151Rfam database. They were computed by two-fold cross-validation procedures, using sample size 1000.

7.5. Conclusions

By performing a comprehensive comparative study of results obtained with the LSCFG, the traditional SCFG and the PF variant, respectively, we showed that significant differences with respect to both predictive accuracy and overall quality of generated sample sets are implied. Actually, we can conclude that the ensemble distribution induced by the considered LSCFG approach is much more centered than that induced by the conventional SCFG variant, and even seems to be slightly more centered than the Boltzmann-distribution of possible structures. This effectively yields less variability during the sampling process, resulting in a less diverse sample set that might contain typical structures significantly more often than others. In principle, a higher prediction accuracy can be reached at the price of a lower diversity of structures within generated sample sets. This is due to the higher explicitness of the underlying SCFG model implied by training the probabilities of the production rules in a length-dependent way.

Notably, since the prediction accuracy is extremely high, the low variety within generated samples allows for the usage of rather small sample sizes to obtain meaningful structure predictions for a given RNA sequence. This indeed means that only a few candidate structures need to be sampled in order to derive high quality predictions, in contrast to the traditional SCFG (and also to competing PF) approach where a comparatively large number of structures need to be generated in order to guarantee that the proposed folding is sufficiently accurate (and reproducible). However, a significant drawback in this context is the almost unavoidable prediction of the family-specific shape (like for instance the cloverleaf shape in case of tRNAs). In fact, when considering an input sequence for which it is known that it belongs to a particular

class of RNA molecules whose structures usually fold to a certain shape, then other, much simpler approaches based on family-specific knowledge might be applied in order to obtain a corresponding prediction.

Nevertheless, an important positive aspect concerning the application of LSCFG based sampling is that existing algorithms for calculating all inside and outside values and the formulae for computing the needed sampling probabilities for statistical sampling as proposed in Section 6.4 can easily be modified by a few simple changes to cope with the extended SCFG model without significant losses in performance. Consequently, for particular RNA types, the extended LSCFG approach studied in this chapter might be able to improve the sampling quality (with respect to the investigated applications) over the conventional one, and especially over the PF variant, while the worst-case time and space complexities remain the same.

Finally, it should be noticed that in contrast to this benefit, however, there are also a number of undesirable pitfalls that come with the additional incorporation of length-dependencies. In fact, a potential overfitting and lack of generalization of the probabilistic structure model seems to become more likely, the first mainly for rather sparse training sets that are subject to high structural diversity and the latter at least for low invariant RNA types like tRNAs that obey to a single typical shape like the cloverleaf structure. Furthermore, the higher dependence on the availability of a rich training set caused by extending the underlying sophisticated SCFG model to a more explicit length-dependent one reduces the applicability of the corresponding probabilistic sampling approach in practice, especially for molecules where there exists hardly knowledge on the typical structural behavior of their family (in the form of trusted RNA databases).

Chapter 8

Evaluating the Effect of Disturbed Ensemble Distributions on Statistical Sampling

In this chapter, we will consider our (L)SCFG based sampling approach in order to perform an analysis on how the quality of generated sample sets and the corresponding prediction accuracy changes when different degrees of disturbances are incorporated into the needed sampling probabilities. This is motivated by the fact that if the results prove to be resistant even to large errors on the distinct sampling probabilities (compared to the exact ones), then it will be an indication that these probabilities do not need to be computed exactly, but it may be sufficient to only approximate them. Thus, it might then be possible to decrease the worst-case time requirements of such an SCFG based sampling method without significant accuracy losses. If, on the other hand, the quality of sampled structures can be observed to strongly react to slight disturbances already, then there is little hope for improving the complexity by corresponding heuristic procedures. We hence provide a reliable test for the hypothesis that a heuristic method could be implemented to improve the time scaling of RNA secondary structure prediction in the worst-case – without sacrificing much of the accuracy of the results.

Our experiments indicate that absolute errors generally lead to the generation of useless sample sets, whereas relative errors seem to have only small negative impact on both the predictive accuracy and the overall quality of resulting structure samples. Based on the respective observations, we present some useful ideas that might be considered for developing a corresponding time-reduced sampling method guaranteeing an acceptable predictive accuracy. We also discuss some inherent drawbacks that arise in the context of approximation. The key results of this chapter are crucial for the design of an efficient and competitive heuristic prediction method based on the increasingly accepted and attractive statistical sampling approach, as we will see in Chapter 9.

8.1. Motivation and Objectives

Briefly, the main objective of this chapter is given as follows: We will consider the (L)SCFG based statistical sampling approach studied in Chapters 6 and 7 in order to perform a comprehensive experimental analysis on the influence of disturbances (in the considered conditional sampling distributions) on the quality of generated sample sets. Particularly, we want to explore to what extend the quality of produced secondary structure samples for a given input sequence and the corresponding predictive accuracy decreases when different degrees of disturbances are incorporated into the needed sampling probabilities.

The prime motivation for such a disturbance analysis lies in the following facts: Suppose both the samples and predictive results are observed to be rather resistant even to large errors in the distinct sampling probabilities (compared to the exact values). Then it seems adequate to believe that the sampling procedure does not have to calculate these probabilities in the exact way, but it may actually suffice if they are only (adequately) approximated. Thus, in this case it might obviously be possible to employ an approximation algorithm (or at least a heuristic method) for sampling probability calculations in order to decrease the worst-case time (and maybe also space) requirements for statistical sampling and hence finally for structure prediction. Furthermore, to ensure that the quality of the generated sample sets remains sufficiently high, analysis results on the effects of different disturbance levels and types should be taken into account for the development of an appropriate approximation scheme (or heuristic). From the other perspective, suppose the quality of sampled structures seems to strongly react on rather slight disturbances already. In that case, there is obviously little hope that the the worst-case complexities of the sampling method can be improved by finding a suitable heuristic procedure for the computation of the needed sampling probabilities.

The aim of our study might hence be declared as to prove or disprove the hypothesis that a heuristic method could be implemented to improve the worst-case complexity of single sequence RNA structure prediction, and to discuss some potential ideas and inherent drawbacks that seem relevant in connection with still guaranteeing high quality results. Although existing algorithms are in practice quite fast on any sequence for which reasonable structure prediction accuracy is expected[1], sacrificing little accuracy might still be assumed worthwhile, given the practical speedup of efficient heuristic methods compared the corresponding exact (non-heuristic) algorithms.

Since for any input sequence, the time (and space) complexities are dominated by those of the inside-outside computations, the most straightforward way for reducing the time complexity of the overall sampling algorithm might be based on an efficient approximation algorithm or heuristic method for deriving the inside and outside values. Therefore, we will incorporate disturbances into these values (that need to be derived for any input sequence) rather than into the underlying grammar parameters (transition and emission probabilities trained on a suitable RNA database). This means that in this chapter, the source of an error will not come from a flawed learning set, although the study of random errors in the applied grammar parameters would actually be analogous to tests performed in connection with the thermodynamic PF [LB05]. The justification for a disturbance study as aspired here is that the parameters of the (L)SCFG underlying the statistical sampling algorithm from [NS11b, SN12b] might be assumed to be available (or if not, can be estimated beforehand in a single training step and might then be used for numerous input sequences). For this reason, applying random errors on the inside and outside values seems to be a much better test in the context of investigations on the impact of a performance improving heuristic.

[1]For example, it takes less than an hour to predict the thermodynamic PF for a 23S rRNA of 2500 nucleotides.

8.2. Outline

As we will see in this chapter, the (L)SCFG based statistical sampling algorithm strongly reacts to any kind of rather small absolute errors already, whereas its reaction even to rather large relative disturbances is in most cases indeed fair enough to still obtain samples of acceptable quality and corresponding meaningful structure predictions. Hence, it seems possible that a reduction of the worst-case time requirements of the evaluated probabilistic sampling approach might be reached – without sacrificing too much predictive accuracy – by approximating the needed sampling probabilities in an appropriate way. Throughout this chapter, we will actually present some useful considerations on how a corresponding approximation scheme (or heuristic procedure) should be constructed in order to ensure that the sampling quality remains sufficiently high.

The rest of this chapter is organized as follows: Section 8.3 introduces the formal framework, particularly definitions of various types and levels of disturbances, as well as a corresponding (robust) recursive sampling strategy that will be considered within this chapter. A comprehensive disturbance analysis based on exemplary RNA data and the corresponding results will follow in Section 8.4, where both the quality of generated sample sets and their applicability to the problem of RNA structure prediction are investigated. Notably, we not only compare different ways for extracting predictions from generated samples in order to assess the predictive accuracy, but also present results on the abstraction level of shapes. Section 8.4 also includes considerations on how to develop a corresponding time-reduced sampling strategy without significant losses in sampling quality. Some of the key results are discussed in Section 8.4.2.3. Finally, Section 8.5 concludes the chapter.

8.3. Preliminaries

In this section, we provide all needed information and introduce the formal framework that will be used in the sequel. We start by formally defining how a number of different types and levels of disturbances can be incorporated into our (L)SCFG based statistical sampling methods. Then, we present a modified version of the employed sampling strategy that (contrary to the original one) manages to deal with disturbed ensemble distributions.

8.3.1. Considered Disturbance Types and Levels

As indicated earlier, with respect to developing a suitable heuristic method to be applied in practice, it is necessary to know about the effects of different disturbance levels and types to get an idea on how precisely the respective values need to be approximated in order to guarantee sufficiently good results and to find out which types of errors pose fundamental problems and which ones are negligible.

For these reasons, given an arbitrary input sequence r of length n, we decided to consider (more or less) skewed inside probabilities[2]

$$\widehat{\alpha}_X(i,j) := \max\left(\min\left(\alpha_X(i,j) + \alpha_X^{err}(i,j), 1\right), 0\right),$$

for $X \in \mathcal{I}_{\mathcal{G}_s}$ and $1 \leqslant i,j \leqslant n$, rather than the corresponding correct values $\alpha_X(i,j)$ (obtained in the preprocessing step for r) for defining the needed sampling probabilities. More precisely, we want to incorporate different stages of (more or less grave) randomly chosen errors into particular inside values for the given sequence, that is into preliminary chosen subsets of the

[2]Note that the function $\max(\min(x,1),0) = \min(\max(x,0),1)$ ensures that the resulting value is still a probability, that is a real value from $[0,1]$.

set of all precomputed inside probabilities $\alpha_X(i,j)$, $X \in \mathcal{I}_{\mathcal{G}_s}$ and $1 \leqslant i,j \leqslant n$. Note that it actually suffices to consider

$$X \in \mathcal{I}^{\alpha}_{\mathcal{G}_s} := \{T, C, A, P, F, G, B, M, O, N, U\} \subset \mathcal{I}_{\mathcal{G}_s},$$

since only those intermediate symbols are needed for defining the diverse sampling probabilities that are used by the employed sampling strategy for obtaining the distinct conditional distributions for drawing particular random choices.

However, in order to reach our previously declared goal, for any fixed value $\mathtt{prob} \in (0,1]$[3], we decided to draw $\alpha^{err}_X(i,j)$ (uniformly) at random from either of the following sets:

$$\mathtt{func}^{win,op}_{\mathcal{J}}(\mathtt{prob}) := \begin{cases} Interval(\mathtt{func}), & \text{if } X \in \mathcal{I} \subseteq \mathcal{I}^{\alpha}_{\mathcal{G}_s} \text{ and } [(j-i+1 > win \text{ and } op = +) \text{ or} \\ & \qquad (j-i+1 \leqslant win \text{ and } op = -)], \\ \{0\}, & \text{else,} \end{cases}$$

such that only inside values of particularly chosen intermediate symbols that lie outside ($op = +$) or within ($op = -$) a considered window of preliminary fixed size are actually disturbed, that is only for those values $\widehat{\alpha}_X(i,j) \neq \alpha_X(i,j)$ might result. Notably, $Interval(\mathtt{func})$ is not centered on $\alpha_X(i,j)$, as it actually describes the set of error values $\alpha^{err}_X(i,j)$ that might be drawn (uniformly) at random – which are then added to $\alpha_X(i,j)$. Anyway, in the sequel, we will basically consider either

$$\mathtt{func}^{win,op}(\mathtt{prob}) := \mathtt{func}^{win,op}_{\mathcal{I}^{\alpha}_{\mathcal{G}_s}}(\mathtt{prob})$$

(that is, disturbances only inside or outside fix-sized window, but for all intermediate symbols),

$$\mathtt{func}_{\mathcal{I}}(\mathtt{prob}) := \mathtt{func}^{n,+}_{\mathcal{I}}(\mathtt{prob}) = \mathtt{func}^{-1,-}_{\mathcal{I}}(\mathtt{prob})$$

(that is, errors for all subword lengths, but only for particular intermediate symbols), or simply

$$\mathtt{func}(\mathtt{prob}) := \mathtt{func}^{n,+}_{\mathcal{I}^{\alpha}_{\mathcal{G}_s}}(\mathtt{prob}) = \mathtt{func}^{-1,-}_{\mathcal{I}^{\alpha}_{\mathcal{G}_s}}(\mathtt{prob})$$

(that is, disturbances on all considered inside values).

Moreover, $\mathtt{func} \in \{\mathtt{mep}, \mathtt{fep}, \mathtt{mev}, \mathtt{fev}\}$ denotes the actual disturbance type. Principally, we distinguish between two degrees of errors: relative and absolute ones. To generate relative errors, we might either use $\mathtt{func} = \mathtt{mep}$ (which stands for *maximum allowed error percentage*, with respect to the corresponding correct value) or $\mathtt{func} = \mathtt{fep}$ (for *fixed error percentage*, which is ought to force greater and hence more severe random errors). Formally, this means that either

$$Interval(\mathtt{mep}) := [-\mathtt{prob} \cdot \alpha_X(i,j), +\mathtt{prob} \cdot \alpha_X(i,j)]$$

or

$$Interval(\mathtt{fep}) := \{-\mathtt{prob} \cdot \alpha_X(i,j), +\mathtt{prob} \cdot \alpha_X(i,j)\}$$

might be employed for randomly drawing a relative error $\alpha^{err}_X(i,j)$, where $\mathtt{prob} \in (0,1]$ indeed defines the desired percentage. Note that the consideration of symmetric intervals (as defined by $Interval(\mathtt{mep})$) is of interest as it models the case that all errors $\alpha^{err}_X(i,j)$ are bounded but do not need to admit the maximum value possible (according to $\mathtt{prob}$). When studying relative errors in connection with this variant, this basically corresponds to assuming a particular approximation ratio of the underlying algorithm. The consideration of discrete sets (as defined by $Interval(\mathtt{fep})$) corresponds to the case that any error takes on the maximum value possible (according to $\mathtt{prob}$). This variant hence explicitly describes the worst-case (by means of magnitudes of incorporated errors) of the symmetric interval variant and is actually of interest

[3]Note that $\mathtt{prob} \in (0,1]$ is must be preliminary chosen and is then assumed to be fixed. This effectively facilitates the study of disturbances of different magnitudes.

as it enables a more reliable study of the influence of disturbances, particularly in cases where the extenuated symmetric interval variant defined by mep seems to have no effect on the resulting accuracy.

For similar reasons, in order to randomly choose an absolute error $\alpha_X^{err}(i,j)$ for obtaining a (potentially) disturbed probability $\widehat{\alpha}_X(i,j)$, we might equivalently consider either

$$Interval(mev) := [-prob, +prob]$$

or

$$Interval(fev) := \{-prob, +prob\},$$

with $prob \in (0,1]$ being a preliminary fixed value. This means we may use $func = mev$ (which stands for *maximum allowed error value*, independent on the corresponding correct value) and $func = fev$ (for *fixed error value*, usually resulting in more grave disturbances) for causing absolute disturbances.

Note that random errors on all outside probabilities $\beta_X(i,j)$, $X \in \mathcal{I}_{\mathcal{G}_s}$ and $1 \leqslant i,j \leqslant n$, could be generated in basically the same way, but since those values can be deliberately excluded from the definition of sampling probabilities[4], this is actually not necessary for the subsequent investigations.

Finally, it should be clear that for $func \in \{mep, fep\}$ (resulting in relative errors), only the magnitudes of the corresponding sampling probabilities (with respect to the implied skewed conditional sampling distributions) change, such that the exact same structures are possible as in the undisturbed case. Hence, we might expect that only the consideration of sufficiently large percentages $prob \in (0,1]$ for generating errors according to $func_{\mathcal{J}}^{win,op}(prob)$ can cause an actual shifting in the ensemble distribution, resulting in significant quality losses. The contrary holds for absolute errors created according to $func_{\mathcal{J}}^{win,op}(prob)$ with $func \in \{mev, fev\}$. In fact, since the (cardinalities of the) respective sets of relevant sampling choices implied by the skewed ensemble distribution generally differ (to a more or less severe extent) from the corresponding exact ones, it must be expected that only rather small fixed error values of $prob \in (0,1]$ are reasonable choices for our purpose. However, since for distinct subword lengths $j-i+1$, $1 \leqslant i,j \leqslant n$, the corresponding probabilities $\alpha_X(i,j)$ for any $X \in \mathcal{I}_{\mathcal{G}_s}^{\alpha}$ usually imply different orders of magnitudes[5], it seems practically impossible to tell how to find an appropriate fixed error value for creating absolute disturbances.

8.3.2. Resulting Modified Sampling Strategy

It should be clear that after the desired errors (according to any of the previously specified variants of either mep, fep, mev or fev) have been incorporated into the precomputed exact inside (and outside) values for a given sequence, the needed conditional sampling distributions (as considered by a particular strategy) are induced by the exact grammar parameters (specifically, transition probabilities) and the disturbed inside (and outside) probabilities for that sequence. This, however, might create the need to (slightly) modify the respective particularly employed sampling strategy such that it finally gets capable to deal with these skewed distributions.

8.3.2.1. Notations and Main Aspects

In this chapter, we will only consider a modified version of the well-established strategy illustrated in Figure 3.2, which has been formally described in Section 6.4.2 where we used

[4]The outside values are in fact constantly cancelled out; details will follow.

[5]In general, longer words tend to be generated with smaller probability since we have to apply more grammar rules, each implying a factor (typically) less than 1 to the probability.

similar notations etc. as in [DL03] in order to facilitate comparison of `Sfold`'s sampling method based on PFs and ours based on a sophisticated SCFG of comparable complexity. Here, however, we will use a different notation which to our taste is significantly more intuitive and especially more easy to handle in connection with formal descriptions of SCFG based sampling strategies.

For example, suppose fragment $R_{i,j} = r_i \dots r_j$ of input sequence r, $1 \leqslant i, j \leqslant n = |r|$, is to be folded, where it is known that the resulting substructure on $R_{i,j}$ must correspond to a (valid) derivation of a particular intermediate symbol $X \in \mathcal{I}_{\mathcal{G}_s}$ (according to the partially formed structure). Then, the strategy considers the corresponding set $acX(i,j)$ of all choices for (valid) derivations of X on $R_{i,j}$, which actually correspond to all possible substructures on $R_{i,j}$ (the mutually exclusive and exhaustive cases for X on $R_{i,j}$). Under the assumption that the alternatives for intermediate symbol X are equal to $X \to Y$ and $X \to VW$, this set is defined as follows:

$$acX(i,j) := acX_Y(i,j) \cup acX_{VW}(i,j),$$

where

$$\begin{aligned} acX_Y(i,j) &:= \{prob \mid prob = \beta_X(i,j) \cdot \alpha_Y(i,j) \cdot Pr_{tr}(X \to Y) \neq 0\} \\ &= \{\beta_X(i,j) \cdot prob \mid \beta_X(i,j) \neq 0 \text{ and } prob = \alpha_Y(i,j) \cdot Pr_{tr}(X \to Y) \neq 0\} \end{aligned}$$

and

$$\begin{aligned} acX_{VW}(i,j) := \{&(k, prob) \mid i \leqslant k \leqslant j \text{ and} \\ &prob = \beta_X(i,j) \cdot \alpha_V(i,k) \cdot \alpha_W(k+1,j) \cdot Pr_{tr}(X \to VW) \neq 0\} \\ = \{&(k, \beta_X(i,j) \cdot prob) \mid i \leqslant k \leqslant j \text{ and } \beta_X(i,j) \neq 0 \text{ and} \\ &prob = \alpha_V(i,k) \cdot \alpha_W(k+1,j) \cdot Pr_{tr}(X \to VW) \neq 0\}. \end{aligned}$$

Consequently, we have to sample from the corresponding conditional probability distribution induced by $acX(i,j)$, that is the random choice is drawn according to the following set of sampling probabilities:

$$\left\{ \frac{prob}{norm} \;\middle|\; prob \in acX_Y(i,j) \text{ or } (k, prob) \in acX_{VW}(i,j) \right\},$$

where obviously,

$$\sum_{prob \in acX_Y(i,j)} \frac{prob}{norm} + \sum_{(k,prob) \in acX_{VW}(i,j)} \frac{prob}{norm} = 1$$

must hold, which can in general easily be guaranteed by using

$$norm = \beta_X(i,j) \cdot \alpha_X(i,j).$$

However, if there may occur inconstancies in the distribution induced by the underlying grammar model (for example if a particular implementation faces problems that arise from numerical imprecisions or if the distribution has been deliberately disturbed as we intend to do in the sequel), we should instead use

$$\begin{aligned} norm &= \sum_{prob \in acX_Y(i,j)} prob + \sum_{(k,prob) \in acX_{VW}(i,j)} prob \\ &= \beta_X(i,j) \cdot \left(\sum_{\beta_X(i,j) \cdot prob \in acX_Y(i,j)} prob + \sum_{(k, \beta_X(i,j) \cdot prob) \in acX_{VW}(i,j)} prob \right) \end{aligned}$$

$$= \beta_X(i,j) \cdot \left(\alpha_Y(i,j) \cdot \Pr_{tr}(X \to Y) + \sum_{i \leqslant k \leqslant j} \alpha_V(i,k) \cdot \alpha_W(k+1,j) \cdot \Pr_{tr}(X \to VW) \right)$$
$$= \beta_X(i,j) \cdot norm_\alpha,$$

which then ensures that the corresponding sampling probabilities still sum up to unity, such that they indeed define a conditional probability distribution.

Note that the sampling strategy effectively works conform with the SCFG model. This means it actually samples one of the possible parse trees of the given input sequence. This is achieved in the following way: at any point in the already partially constructed parse tree, it randomly draws one of the respective mutually exclusive and exhaustive cases (that correspond to the distinct grammar rules with same premise) in order to generate one of the possible subtrees for the given input sequence (which obviously corresponds to one the possible substructures on the considered sequence fragment).

Hence, according to the sampling process, we could have never gotten to a point where we have to consider all mutually exclusive and exhaustive cases for a particular premise $X \in \mathcal{I}_{\mathcal{G}_s}$ on an actual sequence fragment $R_{i,j}$, $1 \leqslant i,j \leqslant n$, if the grammar could not derive the sentential form $r_1 \dots r_{i-1} X r_{j+1} \dots r_n$ from the start symbol (axiom) $S \in \mathcal{I}_{\mathcal{G}_s}$, that is if the outside value $\beta_X(i,j)$ would be equal to 0. This in fact means that the respective probability distribution (conditioned on the considered fragment $R_{i,j}$) from which the strategy randomly samples one of the possible substructures (one valid subtree of the already partially constructed parse tree) is not influenced by the corresponding outside probability. This is due to the fact that $\beta_X(i,j) > 0$ indeed only represents a scaling factor common to all sampling probabilities for the relevant mutually exclusive and exhaustive cases. For this reason, we can obviously without loss of information remove the outside values from the definitions of the needed sampling probabilities.

The correctness of this simplification can easily be formally proven by considering the above defined set $acX(i,j)$ of all choices for possible derivations of intermediate symbol X on sequence fragment $R_{i,j}$. In fact, the sampling strategy randomly draws one of the elements from $acX(i,j)$ according to the corresponding distribution induced by normalizing the probabilities of the elements in $acX(i,j)$ such that they sum up to unity. Particularly, we have

$$1 = \sum_{\beta_X(i,j) \cdot prob \in acX_Y(i,j)} \frac{\beta_X(i,j) \cdot prob}{\beta_X(i,j) \cdot norm_\alpha} + \sum_{(k, \beta_X(i,j) \cdot prob) \in acX_{VW}(i,j)} \frac{\beta_X(i,j) \cdot prob}{\beta_X(i,j) \cdot norm_\alpha}$$
$$= \frac{1}{norm_\alpha} \cdot \left(\sum_{\beta_X(i,j) \cdot prob \in acX_Y(i,j)} prob + \sum_{(k, \beta_X(i,j) \cdot prob) \in acX_{VW}(i,j)} prob \right)$$
$$= \frac{1}{norm_\alpha} \cdot \left(\sum_{prob \in acX_Y(i,j)} \frac{prob}{\beta_X(i,j)} + \sum_{(k, prob) \in acX_{VW}(i,j)} \frac{prob}{\beta_X(i,j)} \right),$$

since $\beta_X(i,j) \neq 0$ holds (due to the definitions of $acX_Y(i,j)$ and $acX_{VW}(i,j)$).

Formal definitions of all corresponding sets $acX(i,j)$, $X \in \mathcal{I}_{\mathcal{G}_s}$ and $1 \leqslant i,j \leqslant n$, that are considered by the modified strategy for any input sequence of length n, including formulae for deriving the respective conditional sampling probabilities, will be given in Section 8.3.2.2. Notably, all those formulae still only depend on some of the parameters of the underlying (L)SCFG model and the corresponding inside values, such that after a preprocessing of the given sequence (which includes the complete inside computation and needs $\mathcal{O}(n^3)$ time in the worst-case), a random candidate structure can be generated in $\mathcal{O}(n^2)$ time.

Nevertheless, due to the consideration of disturbed ensemble distributions, we have to deal with the following aspects concerning the original sampling strategy as described in Section 6.4.2:

Without any errors in the conditional probability distributions (that is, by using the exact probabilistic parameters for the given input sequence, particularly the corresponding inside values), it always successfully generates the sampled loop type for a considered sequence fragment. For example, suppose the sampling procedure decides that base pair $r_i.r_j$ should close a multiloop, then the sequence fragment $R_{i+1,j-1} = r_{i+1} \dots r_{j-1}$ is guaranteed to be folded into an admissible multiloop that by definition contains at least two helical regions radiating out from this loop. However, by using disturbed sampling probabilities – given by the exact parameters of the underlying (L)SCFG model and disturbed inside values for input sequence r, derived by incorporating any sort of errors – the sampling algorithm may choose to form a particular substructure on the fragment $R_{i+1,j-1}$, although this would actually not be possible.

Therefore, we needed to (slightly) modify the sampling procedure to cope with disturbed distributions, where we actually chose to most intuitive solution: We implemented the strategy such that in any case where the sampled substructure type can not be successfully generated, it settles for the partially formed substructure. That is, it either leaves the complete fragment unpaired (if the desired base pairs could not be sampled at all), or else it for example only creates a bulge/interior loop although a multiloop should have been constructed (but only one helix has been successfully sampled). The resulting modified versions of the distinct sampling steps (in pseudocode) are given in Section 8.3.2.2. Figure 8.1 gives a schematic overview of the overall sampling process.

Note that alternatively, the algorithm could have been modified to revise any decisions that lead to incompletely generated substructures, resulting in some sort of backtracking procedures that obviously would have to be applied in order to sample more realistic overall structures for a given RNA sequence. However, as this effectively results in much more complex modifications and eventually yields significant losses in performance, we opted for the simpler and more straightforward first variant to get rid of the described problem.

8.3.2.2. Formal Description of the Sampling Process

As indicated above, base pairs are randomly sampled according to conditional probability distributions (for corresponding fragments), which are derived from definitions of probabilities for particular choices (such as paired and unpaired bases or specific loop types). Notably, they only depend on the precomputed (skewed) inside probabilities $\widehat{\alpha}_X(i,j)$ for input sequence r, the thereof additionally precalculated probabilities

$$\widehat{\alpha}_{AT}(h,j) := \sum_{l=(h-1)+\min_{ps}}^{(j-1)} \widehat{\alpha}_A(h,l) \cdot \widehat{\alpha}_T(l+1,j),$$

$$\widehat{\alpha}_{AB}(h,j) := \sum_{l=(h-1)+\min_{ps}}^{(j-2)} \widehat{\alpha}_A(h,l) \cdot \widehat{\alpha}_B(l+1,j-1),$$

$$\widehat{\alpha}_{AO}(h,j) := \sum_{l=(h-1)+\min_{ps}}^{(j-1)-\min_{ps}} \widehat{\alpha}_A(h,l) \cdot \widehat{\alpha}_O(l+1,j-1),$$

$$\widehat{\alpha}_{AN}(h,j) := \sum_{l=(h-1)+\min_{ps}}^{(j-1)} \widehat{\alpha}_A(h,l) \cdot \widehat{\alpha}_N(l+1,j-1),$$

corresponding to inside values for combined intermediate symbols, where $i \leqslant h \leqslant j$, and of course the trained grammar parameters (transition probabilities only).

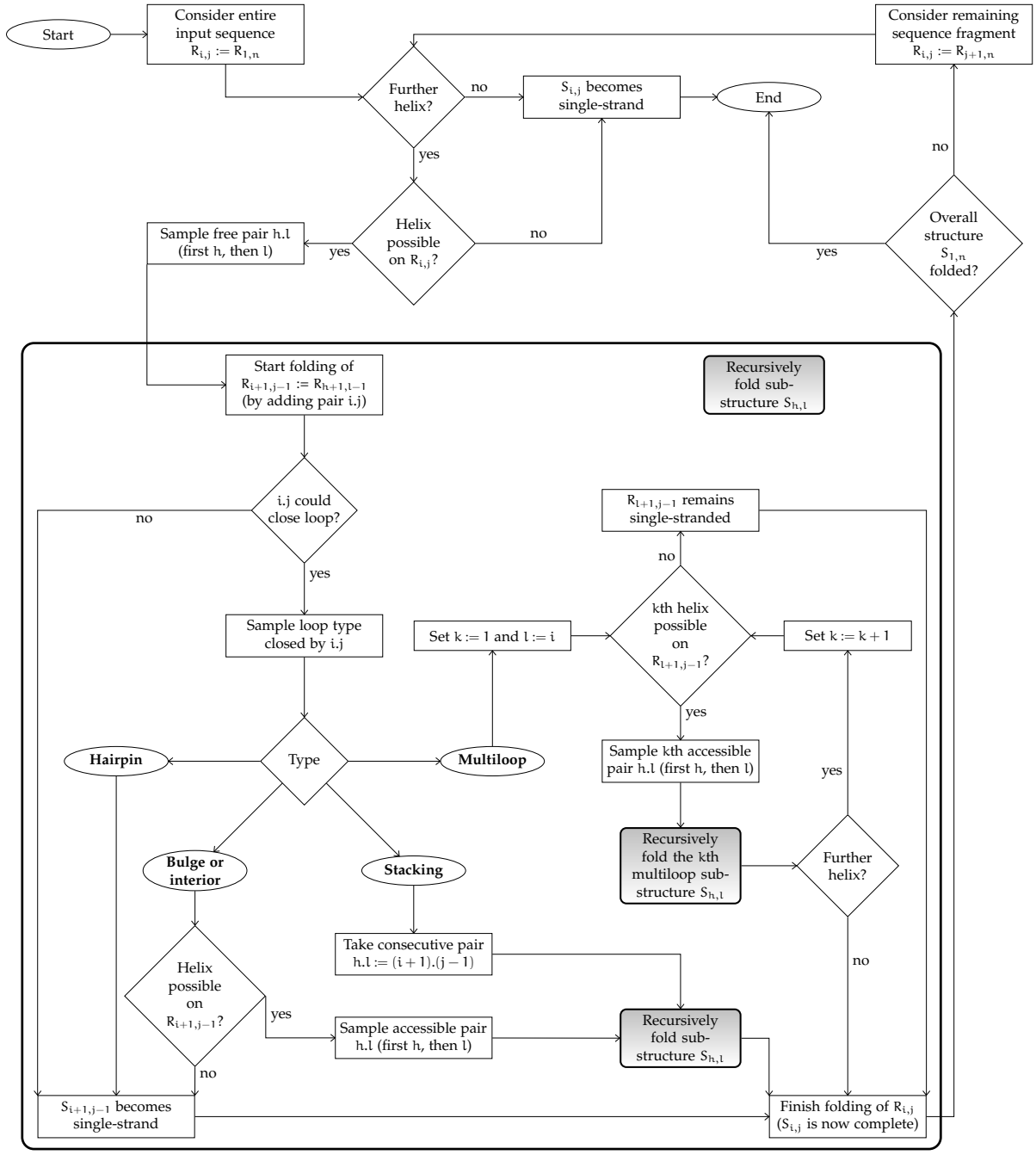


Figure 8.1.: **Flowchart for recursive sampling according to a well-established strategy.** The flowchart describes the sampling of a complete secondary structure $s = S_{1,n}$ for a given input sequence $r = R_{1,n}$ of length n according to an inherently controlled strategy with predetermined order, similar to that of [DL03]).

Algorithm 14 Sampling an entire secondary structure (modified common strategy)

Input: RNA sequence r of length $n \geqslant 1$,
trained transition probabilities $\mathrm{Pr}_{tr}(\mathit{rule})$, for $\mathit{rule} \in \mathcal{R}_{\mathcal{G}_s}$,
precomputed inside probabilities $\widehat{\alpha}_X(i,j)$, for $X \in \mathcal{I}_{\mathcal{G}_s} \cup \{AT, AB, AO, AN\}$ and $1 \leqslant i,j \leqslant n$.
Output: $helices = \big\{(i,j,k) \mid 1 \leqslant i < j \leqslant n$ and $k \geqslant \min_{hel}$ and
$i.j, (i+1).(j-1), \ldots, (i+(k-1)).(j-(k-1))$ are consecutive pairs$\big\}$.

procedure ComputeRandomExteriorLoop()
$helices = \emptyset$
$i = 1$, $j = n$
while $(j - i + 1) \neq 0$ **do**
/*Sample next substructure on $R_{i,j}$ according to $acT(i,j)$, i.e. construct paired substructure starting with free base pair $h.l$, for $i \leqslant h < l \leqslant j$, or leave $R_{i,j}$ unpaired:*/
$extLoopType$ = Sample exterior loop substructure type for $R_{i,j}$ according to $acT(i,j)$
if $extLoopType = C$ **then**
/*$R_{i,j}$ becomes single-stranded:*/
return $helices$
else if $extLoopType = A$ **then**
/*$R_{i,j}$ becomes paired structure:*/
$h = i$, $l = j$
else if $extLoopType = CA$ **then**
/*$R_{i,j}$ becomes paired structure preceded by single-strand:*/
Sample h according to $acT_{CA}(i,j)$
$l = j$
else if $extLoopType = AT$ **then**
/*$R_{i,j}$ becomes paired structure followed by further structure(s):*/
$h = i$
Sample l according to $acT_{AT}(i,j)$
else if $extLoopType = CAT$ **then**
/*$R_{i,j}$ becomes paired structure preceded by single-strand and followed by further structure(s):*/
Sample h according to $acT_{CAT}(i,j)$
Sample l according to $ac^*_{AT}(h,j)$
end if
if $extLoopType \in \{A, CA, AT, CAT\}$ **and** $h.l$ successfully sampled **then**
/*Recursively fold substructures on $R_{h,l}$:*/
$helices = helices \cup \{(h, l, \min_{hel})\}$
$helices = \mathrm{ComputeRandomLoop}(h + (\min_{hel} - 1), l - (\min_{hel} - 1), helices)$
/*Consider the remaining fragment $R_{(l+1),j}$:*/
$i = l + 1$
else
/*Sampling failed (as there exist no valid choices), so stop folding the loop (such that $R_{i,j}$ becomes single-stranded):*/
return $helices$
end if
end while
return $helices$
end procedure

Algorithm 15 Sampling any substructure

procedure ComputeRandomLoop($i, j, helices$)
$loopType$ = Sample loop type closed by i.j according to $acL(i, j)$
if $loopType = F$ **then**
/*Pair i.j closes hairpin loop:*/
return $helices$
else if $loopType = P$ **then**
/*Pair i.j closes stacked pair:*/
$helices[-1, 3] = helices[-1, 3] + 1$ /*increments length of last added helix*/
$helices$ = ComputeRandomLoop($i + 1, j - 1, helices$)
else if $loopType = G$ **then**
/*Pair i.j closes bulge or interior loop:*/
$helices$ = ComputeRandomBulgeInteriorLoop($i, j, helices$)
else if $loopType = M$ **then**
/*Pair i.j closes multiloop:*/
$helices$ = ComputeRandomMultiLoop($i, j, helices$)
else
/*Sampling failed (as there exist no valid choices), so stop folding the loop (such that $R_{i+1,j-1}$ becomes single-stranded hairpin loop):*/
return $helices$
end if
return $helices$
end procedure

Algorithm 16 Sampling a particular bulge or interior loop

procedure ComputeRandomBulgeInteriorLoop($i, j, helices$)
/*Note that the following allows $max_{bulge} = \infty$ (then no restrictions are applied):*/
$loopType$ = Sample bulge or interior loop type on $R_{i+1,j-1}$ according to $acG(i, j)$
if $loopType = BA$ **then**
/*Bulge on the left:*/
Sample h according to $acG_{BA}(i, j)$
$l = j$
else if $loopType = AB$ **then**
/*Bulge on the right:*/
$h = i$
Sample l according to $acG_{AB}(i, j)$
else if $loopType = BAB$ **then**
/*Interior loop:*/
Sample h according to $acG_{BAB}(i, j)$
Sample l according to $ac^*_{AB}(h, j)$
end if
if $loopType \in \{BA, AB, BAB\}$ **and** $h.l$ successfully sampled **then**
/*Recursively fold substructures on $R_{h,l}$:*/
$helices = helices \cup \{(h, l, min_{hel})\}$
$helices$ = ComputeRandomLoop($h + (min_{hel} - 1), l - (min_{hel} - 1), helices$)
else
/*Sampling failed (as there exist no valid choices), so stop folding the loop (such that $R_{i+1,j-1}$ becomes single-stranded hairpin loop):*/
return $helices$
end if
return $helices$
end procedure

Algorithm 17 Sampling a complete multiloop

procedure ComputeRandomMultiLoop($i, j, helices$)
$k = 0$, $l_k = i$
while $(j - l_k - 1) \geqslant \min_{ps}$ **do**
/*Create $(k+1)$th paired substructure on $R_{l_k+1,j-1}$, starting with accessible base pair $h_{k+1}.l_{k+1}$, for $l_k < h_{k+1} < l_{k+1} < j$:*/
if $(k+1) = 1$ **then**
Sample h according to $acM_{UAO}(l_k, j)$
Sample l according to $ac^*_{AO}(h, j)$
else if $(k+1) = 2$ **then**
Sample h according to $acO_{UAN}(l_k, j)$
Sample l according to $ac^*_{AN}(h, j)$
else if $(k+1) \geqslant 3$ **then**
Sample h according to $acN_{UAN}(l_k, j)$
Sample l according to $ac^*_{AN}(h, j)$
end if
if $h.l$ successfully sampled **then**
$h_{k+1} = h$, $l_{k+1} = l$
/*Recursively fold substructures on $R_{h_{k+1},l_{k+1}}$:*/
$helices = helices \cup \{(h_{k+1}, l_{k+1}, \min_{hel})\}$
$helices = \text{ComputeRandomLoop}(h_{k+1} + (\min_{hel} - 1), l_{k+1} - (\min_{hel} - 1), helices)$
/*Decide whether to leave the remaining fragment $R_{l_{k+1}+1,j-1}$ unpaired or not:*/
if $(k+1) \geqslant 2$ **then**
Uniformly draw real value $random \in (0, 1]$
if $random \in (0, dec_U(l_{k+1}, j)]$ **then**
/*No additional base pairs:*/
return $helices$
else if $random \in (dec_U(l_{k+1}, j), 1]$ **then**
/*At least one more paired substructure:*/
$k = k + 1$
end if
end if
else
/*Sampling failed (as there exist no valid choices), so stop folding the loop (such that $R_{l_k+1,j-1}$ becomes single-stranded):*/
return $helices$
end if
end while
return $helices$
end procedure

Algorithms 14 to 17 formally describe how the sampling strategy works. Note that the type (or shape) and actual composition of accessible base pairs and unpaired bases of a particular substructure on a given fragment $R_{i,j}$ are randomly drawn according to the conditional probability distributions induced by the respective sets of all (valid) choices for the unique intermediate symbol of the grammar that generates such substructures (that represents the root of the corresponding derivation subtree). Principally, each of the presented algorithms describing the employed sampling strategy relies on a moderate number of formal set definitions for the respective mutually exclusive and exhaustive cases in order to perform the needed random choices, which basically all obey to the same scheme.

Particularly, for sampling shape and actual composition (free base pairs and unpaired bases) of the exterior loop, Algorithm 14 considers the following sets:

$$acT(i,j) := \{(x, prob) \mid x \in \{C, A, CA, AT, CAT\} \text{ and } prob = \sum_{(y,pr)\in acT_x(i,j)} pr \neq 0\},$$

where

$$\begin{aligned}
acT_C(i,j) &:= \{(0, prob) \mid prob = \widehat{\alpha}_C(i,j) \cdot Pr_{tr}(T \to C) \neq 0\},\\
acT_A(i,j) &:= \{(0, prob) \mid prob = \widehat{\alpha}_A(i,j) \cdot Pr_{tr}(T \to A) \neq 0\},\\
acT_{CA}(i,j) &:= \{(h, prob) \mid (i+1) \leqslant h \leqslant (j+1) - min_{ps} \text{ and}\\
&\qquad prob = \widehat{\alpha}_C(i,h-1) \cdot \widehat{\alpha}_A(h,j) \cdot Pr_{tr}(T \to CA) \neq 0\},\\
acT_{AT}(i,j) &:= \{(l, prob) \mid (i-1) + min_{ps} \leqslant l \leqslant (j-1) \text{ and}\\
&\qquad prob = \widehat{\alpha}_A(i,l) \cdot \widehat{\alpha}_T(l+1,j) \cdot Pr_{tr}(T \to AT) \neq 0\},\\
acT_{CAT}(i,j) &:= \{(h, prob) \mid (i+1) \leqslant h \leqslant j - min_{ps} \text{ and}\\
&\qquad prob = \widehat{\alpha}_C(i,h-1) \cdot \widehat{\alpha}_{AT}(h,j) \cdot Pr_{tr}(T \to CAT) \neq 0\},
\end{aligned}$$

and

$$ac^*_{AT}(h,j) := \{(l, prob) \mid (h-1) + min_{ps} \leqslant l \leqslant (j-1) \text{ and } prob = \widehat{\alpha}_A(h,l) \cdot \widehat{\alpha}_T(l+1,j) \neq 0\}.$$

For sampling the type of the loop closed by a given base pair $i.j$, Algorithm 15 relies on

$$acL(i,j) := \{(x, prob) \mid x \in \{F, P, G, M\} \text{ and } prob = \widehat{\alpha}_x(i+1,j-1) \cdot Pr_{tr}(L \to x) \neq 0\}.$$

Algorithm 16 employs the following sets in order to sample a particular bulge or interior loop (closed by a given base pair $i.j$) on the considered sequence fragment $R_{i+1,j-1}$:

$$acG(i,j) := \{(x, prob) \mid x \in \{BA, AB, BAB\} \text{ and } prob = \sum_{(y,pr)\in acG_x(i,j)} pr \neq 0\},$$

where

$$\begin{aligned}
acG_{BA}(i,j) &:= \{(h, prob) \mid (i+2) \leqslant h \leqslant j - min_{ps} \text{ and}\\
&\qquad prob = \widehat{\alpha}_B(i+1,h-1) \cdot \widehat{\alpha}_A(h,j-1) \cdot Pr_{tr}(G \to BA) \neq 0\},\\
acG_{AB}(i,j) &:= \{(l, prob) \mid i + min_{ps} \leqslant l \leqslant (j-2) \text{ and}\\
&\qquad prob = \widehat{\alpha}_A(i+1,l) \cdot \widehat{\alpha}_B(l+1,j-1) \cdot Pr_{tr}(G \to AB) \neq 0\},\\
acG_{BAB}(i,j) &:= \{(h, prob) \mid (i+2) \leqslant h \leqslant j - min_{ps} - 1 \text{ and}\\
&\qquad prob = \widehat{\alpha}_B(i+1,h-1) \cdot \widehat{\alpha}_{AB}(h,j) \cdot Pr_{tr}(G \to BAB) \neq 0\},
\end{aligned}$$

and

$$ac^*_{AB}(h,j) := \{(h, prob) \mid (h-1) + min_{ps} \leqslant l \leqslant (j-2) \text{ and } prob = \widehat{\alpha}_A(h,l) \cdot \widehat{\alpha}_B(l+1,j-1) \neq 0\}.$$

Finally, for sampling a complete multiloop (closed by a given base pair $i.j$) on the considered sequence fragment $R_{i+1,j-1}$, the following formal definitions are used by Algorithm 17:

$$\begin{aligned}
acM_{UAO}(i,j) &:= \{(h, prob) \mid (i+1) \leqslant h \leqslant j-2\cdot\min_{ps} \text{ and} \\
&\qquad prob = \widehat{\alpha}_U(i+1,h-1)\cdot\widehat{\alpha}_{AO}(h,j)\cdot Pr_{tr}(M\to UAO)\neq 0\},\\
acO_{UAN}(l_k,j) &:= \{(h, prob) \mid (l_k+1) \leqslant h \leqslant j-\min_{ps} \text{ and} \\
&\qquad prob = \widehat{\alpha}_U(l_k+1,h-1)\cdot\widehat{\alpha}_{AN}(h,j)\cdot Pr_{tr}(O\to UAN)\neq 0\},\\
acN_{UAN}(l_k,j) &:= \{(h, prob) \mid (l_k+1) \leqslant h \leqslant j-\min_{ps} \text{ and} \\
&\qquad prob = \widehat{\alpha}_U(l_k+1,h-1)\cdot\widehat{\alpha}_{AN}(h,j)\cdot Pr_{tr}(N\to UAN)\neq 0\},
\end{aligned}$$

as well as

$$\begin{aligned}
ac^*_{AO}(h,j) &:= \{(l, prob) \mid (h-1)+\min_{ps} \leqslant l \leqslant (j-1)-\min_{ps} \text{ and} \\
&\qquad prob = \widehat{\alpha}_A(h,l)\cdot\widehat{\alpha}_O(l+1,j-1)\neq 0\},\\
ac^*_{AN}(h,j) &:= \{(l, prob) \mid (h-1)+\min_{ps} \leqslant l \leqslant (j-1) \text{ and} \\
&\qquad prob = \widehat{\alpha}_A(h,l)\cdot\widehat{\alpha}_N(l+1,j-1)\neq 0\},
\end{aligned}$$

and finally (for deciding whether an additional substructure should be added or not),

$$dec_U(l_{k+1},j) := \frac{\widehat{\alpha}_U(l_{k+1}+1,j-1)\cdot Pr_{tr}(N\to U)}{\widehat{\alpha}_U(l_{k+1}+1,j-1)\cdot Pr_{tr}(N\to U)+\sum\limits_{(h,prob)\in acN(l_{k+1},j)} prob}.$$

It remains to mention that after a preprocessing of the given input sequence – including the complete DP method for deriving all inside probabilities $\widehat{\alpha}_X(i,j)$, for $X\in\mathcal{I}_{\mathcal{G}_s}$ and $1\leqslant i,j\leqslant n$, as well as the subsequent calculation of the additionally needed probabilities $\widehat{\alpha}_x(h,j)$, for $x\in\{AT,AB,AO,AN\}$ and $1\leqslant h,j\leqslant n$, which both take $\mathcal{O}(n^3)$ time and require $\mathcal{O}(n^2)$ storage[6] – each of the probabilities $prob$ defined for a particular choice of a paired base (h or l) in the respective subset ($acX_y(i,j)$ or $ac^*_z(h,j)$) of all possible choices can be derived in constant time. Furthermore, according to their definitions, none of these subsets contains more than n choices for a particular paired base in the worst-case, that is $\mathtt{card}(acX_y(i,j))\in\mathcal{O}(n)$ and $\mathtt{card}(ac^*_z(h,j))\in\mathcal{O}(n)$. Hence, the sampling strategy needs $\mathcal{O}(n)$ time for deriving the respective probability distribution and drawing a corresponding random choice.

Additionally, due to the cardinalities of $\mathcal{O}(n)$ for each of the $\mathcal{O}(1)$ distinct subsets $acX_y(i,j)$ of any main set $acX(i,j)$, each of the probabilities defined for a particular choice of the shape of a random substructure can be computed in $\mathcal{O}(n)$ time (since for each of the $\mathcal{O}(1)$ rules $X\to y$, we have to compute the sum of $\mathtt{card}(acX_y(i,j))\in\mathcal{O}(n)$ terms, where each term is obtained in constant time, see above). Then, the respective probability distribution employed for (shape or loop type) sampling can be derived in constant time (as $\mathtt{card}(acX(i,j))\in\mathcal{O}(1)$). For example, the distribution for sampling the exterior loop substructure type according to $acT(i,j)$ can be derived in $2\cdot\mathcal{O}(1)+3\cdot\mathcal{O}(n)$ time.

Altogether, there obviously results $\mathcal{O}(n)$ time complexity for sampling a random base pair $h.l$ on $R_{i,j}$, $1\leqslant i\leqslant h<l\leqslant j\leqslant n$ (by first sampling the substructure type (if needed), then the leftmost base h and finally the rightmost base l) . Thus, since any structure of size n can have at most $\lfloor\frac{n-\min_{HL}}{2}\rfloor\in\mathcal{O}(n)$ base pairs and any base pair can be sampled in linear time, the time requirements of the sampling strategy for constructing a complete secondary structure $S_{1,n}$ is bounded by $\mathcal{O}(n^2)$.

[6]Note that if we modify the considered SCFG $\mathcal{G}_s$ such that each occurrence of any pattern $x\in\{AT,AB,AO,AN\}$ (in the conclusions of the production rules of $\mathcal{G}_s$) is replaced by a new intermediate symbol $Y\notin\mathcal{I}_{\mathcal{G}_s}$ corresponding to the respective pattern x, then $\widehat{\alpha}_x(i,j)$, $1\leqslant i,j\leqslant n$, is equal to the inside probability $\widehat{\alpha}_Y(i,j)$ of this new intermediate symbol Y and is automatically derived during the inside value computations.

8.4. Analysis of the Influence of Disturbances

The aim of this section is to perform a comprehensive experimental analysis on the influence of disturbances (in the ensemble distribution for a given input sequence) on the quality of sample sets generated by the (L)SCFG based statistical sampling approach devised in Chapters 6 and 7. In fact, we want to explore to what extend the quality of produced secondary structure samples for a given input sequence and the corresponding predictive accuracy decreases when different degrees of errors are incorporated into the needed sampling probabilities.

8.4.1. RNA Structure Data

For our examinations, we decided to consider different sets of trusted RNA secondary structure data for which the (L)SCFG based sampling approach yields good quality results when no disturbances are included in the respective sampling distributions for a given sequence. Therefore, we took

- the same tRNA database (of 2163 distinct tRNA structures with lengths in $[64, 93]$ and about 76 on average, derived from [SHB+98]) and
- the identical 5S rRNA data set (of 1149 distinct sequences with lengths in $[102, 135]$ and about 119 on average, retrieved from [SBEB02])

as utilized in Chapters 6 and 7. In fact, these two rich data sets of trusted RNA secondary structures will be exclusively used as the basis for the following applications, such that the results obtained with disturbed ensemble distributions can easily be opposed to the corresponding exact ones presented in the preceding chapters.

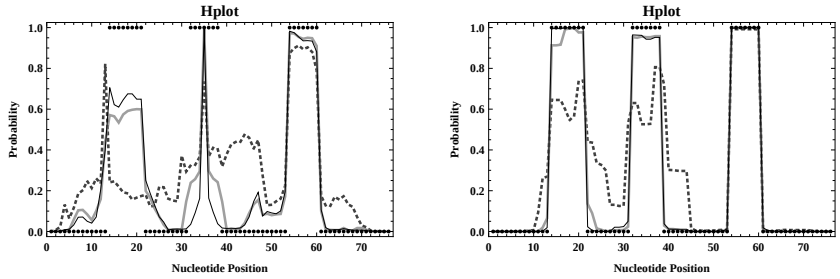
8.4.2. Probability Profiling for Specific Loop Types

In analogy to Chapters 6 and 7, we decided to use probability profiling as a starting point for our disturbance analysis. Particularly, we derived a number of statistical samples for the well-known *Escherichia coli* tRNA^{Ala} sequence by applying the sampling strategy from Section 8.3.2 on the basis of diverse sets of probabilistic parameters for that sequence (here inside probabilities disturbed according to several variants as defined in Section 8.3.1). Then, we calculated corresponding probability profiles. All relevant results are displayed in Figures D.1 to D.14 of Section D. Some of the potentially most interesting ones are presented in Figures 8.2 to 8.4.

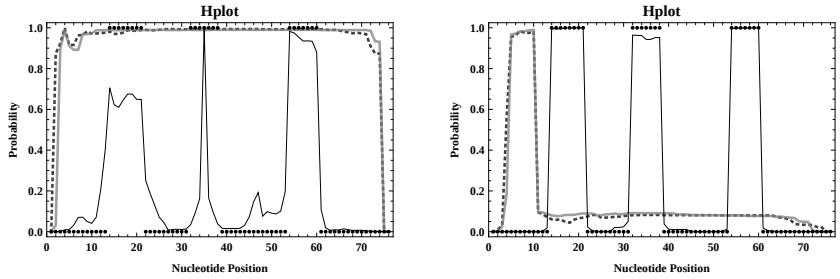
8.4.2.1. Errors on All Values

Let us first consider the profiles displayed in Figure 8.2 (and in Figures D.1 and D.2). Obviously, even if large relative errors on all inside probabilities and hence on the needed conditional sampling probabilities are generated, the sampled structures still exhibit the typical cloverleaf structure of tRNAs, especially for the length-dependent sampling approach where relative disturbances seem to have no significant negative effect on the sampling quality (see Figure 8.2a). However, Figure 8.2b perfectly demonstrates that if the disturbances have been created by adding absolute errors to all inside values, then – even for rather small absolute error values – the resulting samples obtained with both the SCFG and LSCFG approach are useless.

Note that for any given input sequence, it seems to be usually much more important for the employed sampling strategy to be able to identify which ones of the (combinatorially) possible substructures can actually be (validly) formed on the considered sequence fragment rather than to know their exact probabilities (according to the conditional distribution for the respective fragment), for two contrary reasons: First, in order to avoid drawing practically impossible



(a) Relative errors according to mep(prob) (thick gray lines) and fep(prob) (thick dotted darker gray lines), considering percentage prob = 0.99.



(b) Absolute errors according to mev(prob) (thick gray lines) and fev(prob) (thick dotted darker gray lines), using fixed value prob = 10^{-9}.

Figure 8.2.: **Profiling results obtained with relative and absolute disturbances.** Figures show hairpin loop profiles for *E.coli* tRNA^{Ala}, calculated from a random sample of 1000 structures generated with the SCFG (figures on the left) and LSCFG (figures on the right) approach, respectively (under the assumption of the less restrictive grammar parameters $\min_{\text{hel}} = 1$ and $\min_{\text{HL}} = 1$). The exact (undisturbed) results are displayed by the thin black lines, and the correct hairpin loops in *E.coli* tRNA^{Ala} are illustrated by the black points.

choices, which later forces it to leave the considered sequence fragment (at least partially) unpaired[7]. Second, for ensuring that none of the actually valid choices is prohibited during the folding process, such that the sampling procedure might inevitably prefer other (potentially even impossible) substructures.

Consequently, in order to prevent a decline in accuracy of generated structures and a reduction of the overall sampling quality, it seems to be of great importance that the sampling strategy is capable of distinguishing between inside values and especially sampling probabilities that are equal and unequal to zero according to the exact (undisturbed) ensemble distribution for the given input sequence. By adding absolute errors, however, inside or sampling probabilities being equal (unequal) to zero in the exact case might often become unequal (equal) to zero according to the resulting skewed (disturbed) distributions, whereas by incorporating relative errors, all considered inside and sampling probabilities obviously stay equal or unequal to zero (as in the exact case), which intuitively explains the basic observations made from Figure 8.2.

8.4.2.2. Relevant Sampling Probabilities

Nevertheless, in order to draw more detailed conclusions, we counted and compared the relevant (that is, greater than zero) inside and sampling probabilities that were considered for obtaining the profiles presented in Figure 8.2. The results are collected in Tables 8.1 and 8.2.

[7] If those decisions are not revised by employing backtracking procedures, see the description of the modifications incorporated into the sampling algorithm in order to deal with such situations as given in Section 8.3.2.

X	$\mathtt{card}(\mathcal{X}^e)$	$\mathtt{card}(\mathcal{X}^d)$		$\mathtt{card}(\mathcal{X}^e \cap \mathcal{X}^d)$		$\mathtt{card}(\mathcal{X}^e \setminus \mathcal{X}^d)$		$\mathtt{card}(\mathcal{X}^d \setminus \mathcal{X}^e)$	
		mev	fev	mev	fev	mev	fev	mev	fev
A	2649	1733	1698	1547	1529	1102	1120	186	169
B	2926	1704	1741	1660	1709	1266	1217	44	32
C	2926	1847	1847	1806	1808	1120	1118	41	39
F	2926	1873	1891	1840	1849	1086	1077	33	42
G	2696	1777	1781	1616	1617	1080	1079	161	164
M	2548	1597	1573	1357	1338	1191	1210	240	235
N	3002	1957	1945	1957	1945	1045	1057	0	0
O	2770	1838	1818	1727	1692	1043	1078	111	126
P	2649	1721	1745	1553	1563	1096	1086	168	182
T	2926	1865	1905	1822	1869	1104	1057	43	36
U	3002	1938	1913	1938	1913	1064	1089	0	0
AT	2697	2699	2698	2697	2697	0	0	2	1
AB	2552	2554	2553	2552	2552	0	0	2	1
AO	2478	2482	2481	2478	2478	0	0	4	3
AN	2697	2699	2698	2697	2697	0	0	2	1

(a) Traditional SCFG model.

X	$\mathtt{card}(\mathcal{X}^e)$	$\mathtt{card}(\mathcal{X}^d)$		$\mathtt{card}(\mathcal{X}^e \cap \mathcal{X}^d)$		$\mathtt{card}(\mathcal{X}^e \setminus \mathcal{X}^d)$		$\mathtt{card}(\mathcal{X}^d \setminus \mathcal{X}^e)$	
		mev	fev	mev	fev	mev	fev	mev	fev
A	469	1587	1630	325	326	144	143	1262	1304
B	1651	1996	1980	1337	1310	314	341	659	670
C	1096	2001	1987	1053	1025	43	71	948	962
F	871	1888	1850	819	801	52	70	1069	1049
G	729	1603	1583	457	457	272	272	1146	1126
M	359	1517	1525	170	184	189	175	1347	1341
N	1331	1601	1626	786	758	545	573	815	868
O	690	1524	1527	357	355	333	335	1167	1172
P	435	1612	1565	312	306	123	129	1300	1259
T	708	1772	1752	614	594	94	114	1158	1158
U	1571	2038	2059	1323	1322	248	249	715	737
AT	1394	2630	2613	1394	1394	0	0	1236	1219
AB	1829	2485	2469	1829	1829	0	0	656	640
AO	499	2308	2291	499	499	0	0	1809	1792
AN	1832	2620	2602	1831	1832	1	0	789	770

(b) LSCFG model.

Table 8.1.: **Comparison of relevant inside probabilities.** Tabulated values are the numbers of relevant inside probabilities (being greater than zero) that were considered for obtaining the profiles presented in Figure 8.2b (and Figure D.2), where $\mathcal{X}^e := \{\{i,j\} \mid 1 \leqslant i,j \leqslant n \text{ and } \alpha_X(i,j) \neq 0\}$ and $\mathcal{X}^d := \{\{i,j\} \mid 1 \leqslant i,j \leqslant n \text{ and } \widehat{\alpha}_X(i,j) \neq 0\}$.

$\mathcal{X}$	$\mathtt{card}(\mathcal{X}^e)$	$\mathtt{card}(\mathcal{X}^d)$		$\mathtt{card}(\mathcal{X}^e \cap \mathcal{X}^d)$		$\mathtt{card}(\mathcal{X}^e \setminus \mathcal{X}^d)$		$\mathtt{card}(\mathcal{X}^d \setminus \mathcal{X}^e)$	
		mev	fev	mev	fev	mev	fev	mev	fev
$\mathcal{T}_C$	76	36	39	36	39	40	37	0	0
$\mathcal{T}_A$	54	33	41	22	32	32	22	11	9
$\mathcal{T}_{CA}$	1829	766	856	480	673	1349	1156	286	183
$\mathcal{T}_{AT}$	2595	999	961	966	924	1629	1671	33	37
$\mathcal{T}_{CAT}$	2628	1644	1646	1644	1646	984	982	0	0
$\mathcal{AT}$	62102	25675	25693	24940	24671	37162	37431	735	1022
$\mathcal{L}_F$	2775	1750	1777	1750	1777	1025	998	0	0
$\mathcal{L}_P$	2522	1533	1542	1487	1486	1035	1036	46	56
$\mathcal{L}_G$	2552	1543	1539	1540	1536	1012	1016	3	3
$\mathcal{L}_M$	2408	1290	1265	1288	1261	1120	1147	2	4
$\mathcal{G}_{BA}$	59580	23057	23288	22390	22547	37190	37033	667	741
$\mathcal{G}_{AB}$	0	0	0	0	0	0	0	0	0
$\mathcal{G}_{BAB}$	59476	36453	37479	36428	37461	23048	22015	25	18
$\mathcal{AB}$	1041908	454132	457662	441016	442856	600892	599052	13116	14806
$\mathcal{M}_{UAO}$	56901	41296	40054	41208	40010	15693	16891	88	44
$\mathcal{AO}$	980735	488660	456896	473742	442002	506993	538733	14918	14894
$\mathcal{O}_{UAN}$	56999	41352	40087	41329	40066	15670	16933	23	21
$\mathcal{N}_{UAN}$	49970	36636	35377	36615	35356	13355	14614	21	21
$\mathcal{AN}$	985172	511715	490277	497364	474716	487808	510456	14351	15561

(a) Traditional SCFG model.

$\mathcal{X}$	$\mathtt{card}(\mathcal{X}^e)$	$\mathtt{card}(\mathcal{X}^d)$		$\mathtt{card}(\mathcal{X}^e \cap \mathcal{X}^d)$		$\mathtt{card}(\mathcal{X}^e \setminus \mathcal{X}^d)$		$\mathtt{card}(\mathcal{X}^d \setminus \mathcal{X}^e)$	
		mev	fev	mev	fev	mev	fev	mev	fev
$\mathcal{T}_C$	7	7	7	7	7	0	0	0	0
$\mathcal{T}_A$	1	3	1	0	0	1	1	3	1
$\mathcal{T}_{CA}$	33	280	198	13	7	20	26	267	191
$\mathcal{T}_{AT}$	55	256	298	33	25	22	30	223	273
$\mathcal{T}_{CAT}$	161	477	461	157	153	4	8	320	308
$\mathcal{AT}$	2936	17032	26734	1916	1870	1020	1066	15116	24864
$\mathcal{L}_F$	845	795	780	795	780	50	65	0	0
$\mathcal{L}_P$	409	603	581	295	292	114	117	308	289
$\mathcal{L}_G$	669	431	429	428	423	241	246	3	6
$\mathcal{L}_M$	308	152	162	152	162	156	146	0	0
$\mathcal{G}_{BA}$	401	844	881	351	355	50	46	493	526
$\mathcal{G}_{AB}$	0	0	0	0	0	0	0	0	0
$\mathcal{G}_{BAB}$	5074	11964	11884	4002	3916	1072	1158	7962	7968
$\mathcal{AB}$	173376	457279	487255	109429	110855	63947	62521	347850	376400
$\mathcal{M}_{UAO}$	4229	10068	10201	3926	3939	303	290	6142	6262
$\mathcal{AO}$	19149	279648	298214	8090	8203	11059	10946	271558	290011
$\mathcal{O}_{UAN}$	11284	16633	16787	9773	9863	1511	1421	6860	6924
$\mathcal{N}_{UAN}$	11880	18444	18496	10324	10491	1556	1389	8120	8005
$\mathcal{AN}$	89494	306125	329250	45081	44257	44413	45237	261044	284993

(b) LSCFG model.

Table 8.2.: **Comparison of relevant sampling probabilities.** Tabulated values are the numbers of relevant sampling probabilities (being greater than zero) that were considered for obtaining the profiles presented in Figure 8.2b (and Figure D.2), where $\mathcal{X}_y^z := \bigcup_{1 \leqslant i,j \leqslant n} \mathrm{ac}X_y(i,j)$ and $\mathcal{Y}^z := \bigcup_{1 \leqslant i \leqslant h \leqslant j \leqslant n} \mathrm{ac}^*_Y(h,j)$, with $z = e$ and $z = d$ denoting the exact and disturbed values, respectively.

First, it seems obvious that due to the more explicit length-dependent version of the considered grammar parameters (length-dependently trained transition and emission probabilities), there should generally result a much smaller number of relevant inside values and sampling probabilities when applying the LSCFG model rather than the conventional one. Tables 8.1 and 8.2 exemplarily prove this intuitive assumption. Note that this effect might indeed be responsible for the observation that the LSCFG based sampling approach reacts considerably less to large relative errors than the conventional length-independent variant, as indicated by Figure 8.2a: less inside probabilities are effectively disturbed, such that the extend of the relative errors imposed on the corresponding sampling probabilities is inevitably smaller for the LSCFG variant than for the length-independent one.

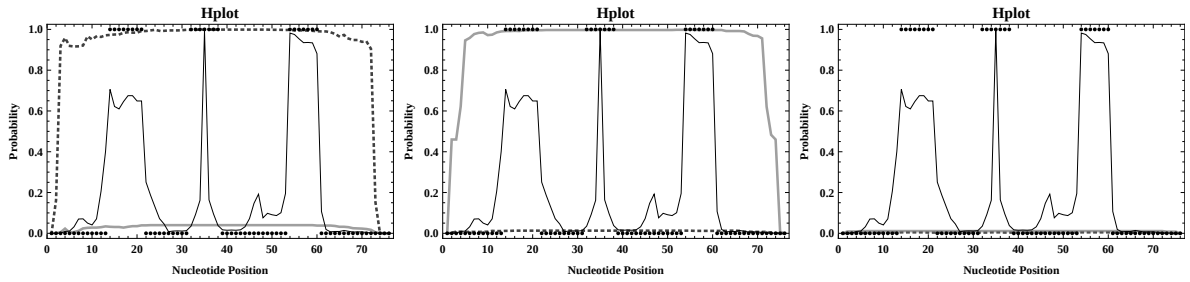
Moreover, there are much more relevant exact inside and sampling probabilities than corresponding relevant disturbed values for basically any (intermediate) symbol when considering the traditional SCFG model, whereas for the LSCFG variant the contrary holds, that is generally way more inside and sampling probabilities are relevant in the disturbed cases than in the exact case. Actually, in both cases (length-dependent and not), the numbers of relevant disturbed inside values $\widehat{\alpha}_X(i,j)$, $1 \leqslant i,j \leqslant n$, are rather similar (for basically all $X \in \mathcal{I}^{\alpha}_{\mathcal{G}_s}$). This is in contrast to the numbers of relevant sampling probabilities (corresponding to valid choices for substructures) for the distinct sampling steps. In fact, those are in general to a large extend greater when using the traditional SCFG approach than under the assumption of the corresponding LSCFG model. This behavior might be the reason for the fundamental differences in the resulting (albeit useless) loop profiles presented in Figure 8.2b.

Finally, it remains to mention that under the assumption of the conventional SCFG model, it happens that for any $X \in \mathcal{I}^{\alpha}_{\mathcal{G}_s}$, most inside values are relevant in both the exact and the disturbed case, whereas significantly less are relevant only in the exact case and very few are only relevant in the disturbed case (see Table 8.1a). Considering the LSCFG variant, however, for any $X \in \mathcal{I}^{\alpha}_{\mathcal{G}_s}$ the least inside values are relevant only in the exact case, as indicated by Table 8.1b. Obviously, this seems to be the natural consequence of the previously formulated observations.

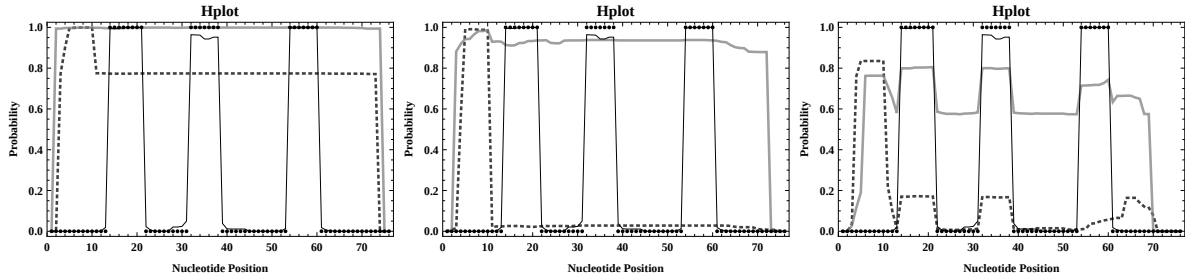
8.4.2.3. Errors Only on Particular Values

Now, in an attempt to find out in which cases particular absolute errors have a very significant (negative) impact on the resulting sampling quality and to identify potentially existing situations where they barely influence the output of the applied statistical sampling algorithm, we want to consider some of the more specialized variants for generating absolute disturbances (as defined in Section 8.3.1). The corresponding profiles are basically shown in Figures 8.3 and 8.4 (as well as in Figures D.3 to D.14).

Notably, even if absolute disturbances may only occur for inside values $\alpha_X(i,j)$, $X \in \mathcal{I}^{\alpha}_{\mathcal{G}_s}$, with $j - i + 1 > win$ (that is, for substructure lengths greater than a particular fixed value win), the corresponding sampling results are of no practical use at all (see Figure 8.3). In fact, there seem to be no noticeable improvements when considering increasing values of win, which means that even if more inside values $\alpha_X(i,j)$, $X \in \mathcal{I}^{\alpha}_{\mathcal{G}_s}$, namely those satisfying $j - i + 1 \leqslant win$, are guaranteed to be exact (contain neither relative nor absolute errors), the resulting samples might not be expected to gain in quality. This observation is actually unfortunate as regards the derivation of a corresponding heuristic version of the inside algorithm, since the inside computation starts by calculating the respective values for small sequence fragments and subsequently considers larger ones. This means the straightforward approach of deriving all values $\alpha_X(i,j)$, $X \in \mathcal{I}^{\alpha}_{\mathcal{G}_s}$, with $j - i + 1 \leqslant win$ in the exact way and approximating only the remaining ones, that is using a constant window size win for exact calculations, might not yield results of acceptable quality if absolute errors can not be ruled out (completely).

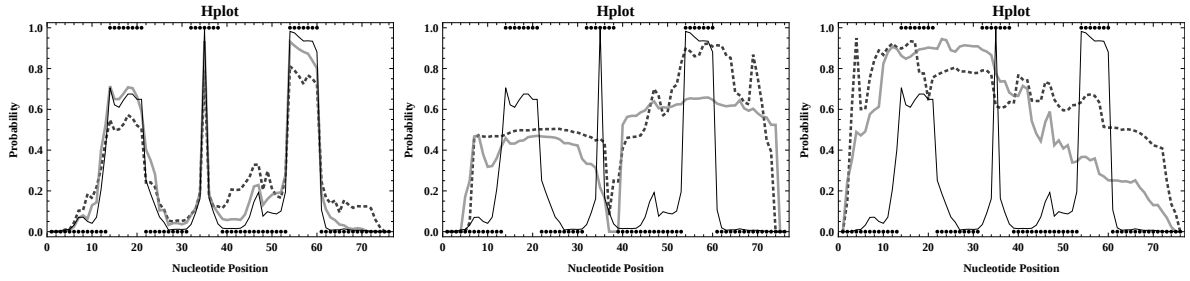


(a) Results for traditional SCFG model.

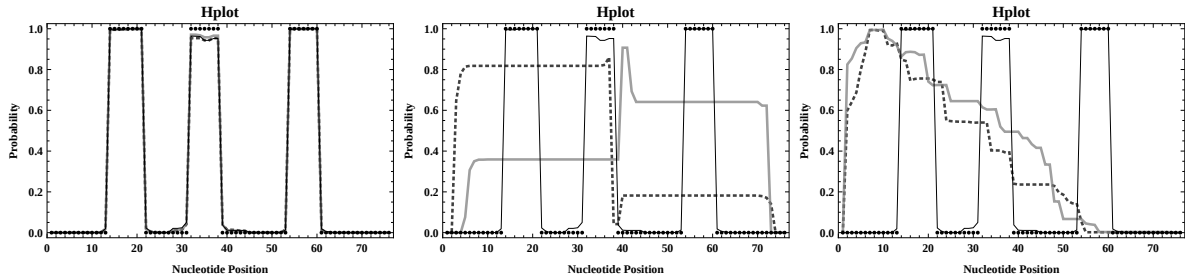


(b) Results for LSCFG model.

Figure 8.3.: **Profiling results obtained with absolute errors only on values for long fragments.** Figures show hairpin loop profiles corresponding to those presented in Figure 8.2b, where absolute errors were derived according to $\mathrm{mev}^{win,+}(\mathtt{prob})$ (thick gray lines) and $\mathrm{fev}^{win,+}(\mathtt{prob})$ (thick dotted darker gray lines), respectively, with $\mathtt{prob} = 10^{-9}$ and $win \in \{15, 38, 60\}$ (figures from left to right).



(a) Results for traditional SCFG model.



(b) Results for LSCFG model.

Figure 8.4.: **Profiling results obtained with absolute errors only on values for short fragments.** Figures show hairpin loop profiles corresponding to those presented in Figure 8.2b, where absolute errors were derived according to $\mathrm{mev}^{win,-}(\mathtt{prob})$ (thick gray lines) and $\mathrm{fev}^{win,-}(\mathtt{prob})$ (thick dotted darker gray lines), respectively, with $\mathtt{prob} = 10^{-9}$ and $win \in \{15, 38, 60\}$ (figures from left to right).

Nevertheless, as we can see from Figure 8.4, if absolute disturbances may only occur for inside values $\alpha_X(i,j)$, $X \in \mathcal{I}^{\alpha}_{\mathcal{G}_s}$, with $j - i + 1 \leqslant win$ (that is, for substructure lengths less than or equal to a particular fixed value win), the corresponding sampling results might actually be of acceptable quality, but seemingly only for rather small values of win. This means in order to obtain a practically applicable heuristic, it seems a good idea to consider a constant (small enough) window of size win and compute all values $\alpha_X(i,j)$, $X \in \mathcal{I}^{\alpha}_{\mathcal{G}_s}$, with $j - i + 1 > win$ in the exact way, thus approximating only those satisfying $j - i + 1 \leqslant win$. However, due to the contrary course of action of traditional inside calculations, this approach can obviously not be realized. Consequently, this observation does not contribute to developing an appropriate heuristic variant of the preprocessing step, but it actually motivates the construction of an innovative sampling strategy that takes on a reverse sampling direction. More specifically, that constructs substructures in an inside-to-outside fashion, contrary to the generation of corresponding derivation trees according to the underlying grammar.

Finally, for the sake of completeness, it should be noted that we performed additional experiments by incorporating absolute errors (for all subword lengths) only for any of the distinct intermediate symbols $X \in \mathcal{I}^{\alpha}_{\mathcal{G}_s}$ at once (that is, by disturbing only the inside values $\alpha_X(i,j)$, $1 \leqslant i,j \leqslant n$, for a particular $X \in \mathcal{I}^{\alpha}_{\mathcal{G}_s}$). The obtained results show that some intermediates are more sensitive with respect to disturbances in the underlying ensemble distribution than others (see Figures D.7 to D.14 of Section D). In principle, the strongest (negative) reactions to the influence of the generated absolute errors were observed for symbols T, C, A, F (for the traditional SCFG model), G and U. Less severe quality losses basically resulted for intermediates M, O, N and P. Moreover, for two symbols, namely F (for the LSCFG model) and B, we recognize no noticeable impact of the caused disturbances to the accuracy of the generated sample sets.

8.4.3. Prediction Accuracy – Sensitivity and PPV

In order to investigate to what extend the accuracy of predicted foldings changes when different dimensions of relative disturbances are incorporated into the needed sampling probabilities, we decided to perform a series of cross-validation experiments based on the same partitions of the tRNA and 5S rRNA databases into 10 approximately equal-sized folds, respectively, as considered in Chapters 6 and 7. In particular, for each sequence, we generated several sample sets on the basis of different relative error types and values. From each of the produced samples, we derived corresponding predictions according to a number of competing reasonable selection principles and construction schemes.

Briefly, we employed two different well-defined selection procedures in order to identify one particular structure from the produced sample as prediction: First, we picked the most likely secondary structure (that is, the one with the highest probability among all sampled candidate structures for the input sequence according to the induced (L)SCFG model), in strong analogy to traditional SCFG based probabilistic structure prediction methods. This choice will be denoted by *most probable (MP)* structure in the sequel. Additionally, as one of the most straightforward and reasonable choices for statistically representative samples of the overall structure ensemble, we took the most frequently sampled folding (that is, the one with the highest number of occurrences among all candidate structures within the generated sample set), which in accordance with Chapters 6 and 7 will be named MF structure in the sequel.

Note that if the samples are indeed representative with respect to the underlying ensemble distribution (that is, if a sufficiently large number of candidate foldings is randomly generated on the basis of the corresponding conditional probability distributions considered by the employed strategy), then these two predictions should be rather identical in most cases, at least if no disturbances are considered. In fact, any representative set of candidate structures for a given input sequence obtained by (L)SCFG based statistical sampling obviously reflects the probability distribution on all feasible foldings of that sequence which strongly depends on the

corresponding inside probabilities. Thus, if the preprocessed inside values contain any errors, then the MF structure of a particular statistically representative sample set corresponds to the most likely folding of the given sequence with respect to the skewed ensemble distribution induced by the disturbed inside values, whereas the MP structure of that sample is indeed equal to the most likely folding (among all generated candidate structures) with respect to the exact ensemble distribution[8]. Hence, the results for MP and MF structure predictions might differ in the disturbed cases, especially as the gravity of generated disturbances grows.

However, we decided to additionally apply the two different construction schemes devised in Section 6.5 for computing a new structure as predicted folding, where the predicted structure itself must not necessarily be contained in the given sample. Particularly, we first determined a single MEA structure of the generated sample set as defined in Section 6.5.2. Furthermore, we calculated the unique consensus structure of the produced sample, that is the corresponding ensemble centroid (for details, see for example [DCL05]). Note that for similar reasons as discussed above for MF structure predictions, MEA and centroid structures obtained from statistically representative sample sets can only reflect the skewed ensemble distribution rather than the exact one in the disturbed case.

Last but not least, we derived two different sets of γ_{t-o}-MEA and γ_{t-o}-centroid structures for the produced samples, respectively, as defined in Sections 6.5.2 and 6.5.3, in order to derive corresponding ROC curves and to calculate the respective estimated AUC values for both the MEA and the centroid prediction principle. This obviously allows for a much more informative and reliable comparison of the predictive powers of the different sampling variants than considering only the corresponding results for the default choice $\gamma_{t-o} = 1$.

The (unadjusted) sensitivity and PPV measures obtained by considering the four different (unparameterized) prediction principles sketched above are listed in Tables D.1a and D.2a, where a few selected ones are presented in Table 8.3. The corresponding AUC values obtained by varying instances of γ_{t-o} are all collected in Tables D.1b and D.2b, some of them are additionally reported in Table 8.4. Note that in accordance with the preceding chapters, we considered any value of

$$\gamma_{t-o} \in \{1.25^k \mid -12 \leqslant k \leqslant -1\} \cup \{2^k \mid 0 \leqslant k \leqslant 12\}$$

in order to obtain appropriate ROC curves and corresponding AUC values. Plots of some of the resulting curves can be found in Figures D.15 to D.18 of Section D.

Let us first consider the results reported in Table 8.3. As we can see, the PPV is principally not affected by the different dimensions of disturbances caused according to mep(prob), as only in the case of MF structure prediction one can observe a slight change for the worse. However, with increasing value of mep, there results a moderate decline in sensitivity (with respect to all four prediction schemes) of up to about 10% for the traditional and 5% for the length-dependent sampling approach in the case of tRNAs, whereas for 5S rRNAs, the sensitivity values only decrease up to about 3% to 4% for both sampling variants. Unsurprisingly, for both RNA data, the change for the worse by means of measured sensitivity is less significant when considering MP structure predictions than when employing any of the other three principles, especially in the case of the LSCFG model. This is due to the fact that MP structures are always extracted by relying on the exact distribution (see discussion above). Altogether, these observations indicate that relative disturbances caused by mep do not have a significant negative effect on the predictive accuracy.

Moreover, Table 8.3 indicates that generating errors according to the fep(prob) variant (unsurprisingly) yields greater losses in the accuracies of selected predictions. In fact, as prob

[8]This is due to the fact that the probability of a particular folding of a given RNA sequence (that is, the probability of the corresponding derivation tree) depends only on the considered set of grammar parameters (transition and emission probabilities).

Approach	Errors	MP struct.		MF struct.		MEA struct.		Centroid	
		Sens.	PPV	Sens.	PPV	Sens.	PPV	Sens.	PPV
SCFG	—	0.7818	0.8437	0.7792	0.8445	0.7324	0.8939	0.6754	0.9158
	mep(0.5)	0.7822	0.8447	0.7599	0.8370	0.7169	0.8927	0.6607	0.9140
	mep(0.99)	0.7590	0.8388	0.6768	0.8004	0.6414	0.8877	0.5817	0.9127
	fep(0.5)	0.7798	0.8440	0.7234	0.8184	0.6864	0.8896	0.6292	0.9134
	fep(0.99)	0.4101	0.7295	0.2864	0.5590	0.2532	0.7776	0.2157	0.8291
LSCFG	—	0.8545	0.9534	0.8542	0.9535	0.8335	0.9736	0.8250	0.9783
	mep(0.5)	0.8545	0.9534	0.8429	0.9524	0.8236	0.9731	0.8150	0.9773
	mep(0.99)	0.8519	0.9533	0.7988	0.9413	0.7833	0.9676	0.7735	0.9726
	fep(0.5)	0.8548	0.9536	0.8224	0.9486	0.8029	0.9707	0.7940	0.9758
	fep(0.99)	0.7530	0.9325	0.5769	0.8623	0.5668	0.9075	0.5567	0.9195

(a) For our tRNA database.

Approach	Errors	MP struct.		MF struct.		MEA struct.		Centroid	
		Sens.	PPV	Sens.	PPV	Sens.	PPV	Sens.	PPV
SCFG	—	0.4251	0.5372	0.4251	0.5363	0.3403	0.6967	0.2689	0.8044
	mep(0.5)	0.4143	0.5280	0.4160	0.5290	0.3334	0.6987	0.2643	0.8051
	mep(0.99)	0.3897	0.5227	0.3894	0.5216	0.2957	0.7069	0.2362	0.8072
	fep(0.5)	0.4055	0.5203	0.4049	0.5198	0.3209	0.7068	0.2532	0.8087
	fep(0.99)	0.2043	0.4410	0.1756	0.3788	0.1066	0.6867	0.0814	0.7666
LSCFG	—	0.8993	0.9412	0.8997	0.9409	0.8959	0.9513	0.8873	0.9574
	mep(0.5)	0.8993	0.9412	0.8909	0.9380	0.8903	0.9478	0.8819	0.9541
	mep(0.99)	0.8989	0.9414	0.8639	0.9269	0.8659	0.9408	0.8574	0.9482
	fep(0.5)	0.8993	0.9412	0.8796	0.9328	0.8798	0.9445	0.8716	0.9515
	fep(0.99)	0.8251	0.9052	0.7162	0.8375	0.7148	0.8661	0.6986	0.8879

(b) For our 5S rRNA database.

Table 8.3.: **Prediction results by means of sensitivity and PPV.** All values have been computed by 10-fold cross-validation procedures, using a sample size 1000 and $\min_{\text{hel}} = \min_{\text{HL}} = 1$.

gets greater, there generally result considerably smaller PPV values for all four prediction schemes (mostly for MF structures) than in the corresponding undisturbed case. Furthermore, the respective sensitivity values degrade enormously, albeit again comparatively less in connection with MP structure predictions. However, these changes for the worse are obviously less significant when using the length-dependent sampling approach instead of the more general conventional variant, which matches the observations made above for disturbances caused by mep(prob). Nevertheless, errors produced according to fep(prob) for moderate percentages prob seem to generally have only a rather small influence on the resulting prediction accuracy. In most cases, only marginal losses in performance can be expected when disturbances are generated by fep(prob) with values prob of up to about 0.5, whereas for percentages of up to about 0.75, there should usually still result an acceptable accuracy of selected predictions (according to any of the four considered extraction principles).

Finally, it should be mentioned that all these observations and conclusions are actually affirmed by comparing the more reliable AUC results given in Table 8.4, which draw a rather similar picture of the behavior of both sampling approaches under the influence of the considered types and dimensions of relative disturbances in the underlying ensemble distribution.

Approach	Errors	MEA struct.	Centroid
SCFG	—	0.828522	0.833894
	mep(0.5)	0.819658	0.823811
	mep(0.99)	0.786645	0.788478
	fep(0.5)	0.805999	0.807240
	fep(0.99)	0.440021	0.422778
LSCFG	—	0.936285	0.919736
	mep(0.5)	0.932121	0.916321
	mep(0.99)	0.916540	0.896024
	fep(0.5)	0.924191	0.908943
	fep(0.99)	0.752030	0.722737

(a) For our tRNA database.

Approach	Errors	MEA struct.	Centroid
SCFG	—	0.409278	0.408549
	mep(0.5)	0.401914	0.400515
	mep(0.99)	0.376683	0.375488
	fep(0.5)	0.400827	0.397566
	fep(0.99)	0.189628	0.182902
LSCFG	—	0.914801	0.918933
	mep(0.5)	0.911963	0.915503
	mep(0.99)	0.902330	0.905126
	fep(0.5)	0.906507	0.911063
	fep(0.99)	0.776239	0.777355

(b) For our 5S rRNA database.

Table 8.4.: **Prediction results by means of AUC values.** All values have been computed by 10-fold cross-validation procedures, using sample size 1000 and $\min_{\text{hel}} = \min_{\text{HL}} = 1$.

8.4.4. Sampling Quality – Specific Values Related to Shapes

We want to complete our analysis of the influence of disturbances to the quality of probabilistic statistical sampling by considering several specific values related to the shapes of predictions and sampled structures, precisely those defined in Section 6.6.2.5. In analogy to Chapters 6 and 7, we can easily compute the respective values from the predicted structures and the corresponding sample sets that were derived for the calculation of the sensitivity and PPV measures in the last section. The obtained results are collected in Tables D.3a to D.4g of Section D. Some of the most interesting ones are recorded in Tables 8.5 and 8.6.

First, as regards tRNAs, we observe that for MP predictions, disturbances caused by mep(prob) do generally not have a noticeable negative impact on the frequency of correct structure predictions (see Table D.3a), and for the three other extraction principles, such disturbances do at least not yield a significant decline of the corresponding CSP_{freq} value for shape levels 2 to 5 and under the assumption of the LSCFG approach, where for MF structures, there indeed results a slightly higher CSP_{freq} value with increasing relative error percentage prob (see Tables D.3b to D.3d). When the more intensive variant as defined by fep(prob) is used for incorporating random errors into the considered sampling probabilities, the LSCFG based sampling algorithm still yields acceptable results with respect to CSP_{freq} on abstraction levels 2 to 5, where for MP and MF structure predictions it obviously behaves quite resistant to the imposed distributions even for large values of prob.

Approach	Errors	Shape Level					
		0	1	2	3	4	5
SCFG	—	0.2413	0.4082	0.5548	0.5548	0.5552	0.6278
	mep(0.5)	0.2409	0.4068	0.5548	0.5548	0.5552	0.6265
	mep(0.99)	0.1877	0.3551	0.5382	0.5382	0.5386	0.6075
	fep(0.5)	0.2339	0.4017	0.5511	0.5511	0.5516	0.6269
	fep(0.99)	0.0014	0.0384	0.1979	0.1979	0.1984	0.2326
LSCFG	—	0.3324	0.4956	0.6574	0.6574	0.6579	0.7351
	mep(0.5)	0.3324	0.4956	0.6574	0.6574	0.6579	0.7351
	mep(0.99)	0.3236	0.4892	0.6560	0.6560	0.6565	0.7332
	fep(0.5)	0.3324	0.4966	0.6588	0.6588	0.6593	0.7369
	fep(0.99)	0.0624	0.2626	0.6246	0.6250	0.6250	0.6967

(a) CSP_{freq} values (for selection principle MP struct.).

Approach	Errors	Shape Level					
		0	1	2	3	4	5
SCFG	—	0.2099	0.3699	0.5594	0.5594	0.5599	0.6302
	mep(0.5)	0.1683	0.3301	0.5372	0.5372	0.5377	0.6047
	mep(0.99)	0.0522	0.1822	0.4517	0.4517	0.4517	0.5215
	fep(0.5)	0.1049	0.2547	0.5155	0.5155	0.5160	0.5793
	fep(0.99)	0.0000	0.0125	0.1110	0.1110	0.1119	0.2062
LSCFG	—	0.3269	0.4892	0.6560	0.6565	0.6565	0.7337
	mep(0.5)	0.2534	0.4235	0.6708	0.6708	0.6713	0.7485
	mep(0.99)	0.1137	0.2954	0.6801	0.6801	0.6801	0.7568
	fep(0.5)	0.1794	0.3653	0.6704	0.6704	0.6709	0.7531
	fep(0.99)	0.0023	0.1262	0.6334	0.6334	0.6357	0.7240

(b) CSP_{freq} values (for selection principle MF struct.).

Approach	Errors	Shape Level					
		0	1	2	3	4	5
SCFG	—	0.0555	0.2094	0.4193	0.4193	0.4207	0.4679
	mep(0.5)	0.0416	0.1817	0.4045	0.4045	0.4055	0.4489
	mep(0.99)	0.0125	0.0989	0.3112	0.3112	0.3126	0.3570
	fep(0.5)	0.0245	0.1364	0.3662	0.3662	0.3666	0.4059
	fep(0.99)	0.0000	0.0014	0.0245	0.0245	0.0250	0.0546
LSCFG	—	0.1854	0.3574	0.4919	0.4919	0.4919	0.5465
	mep(0.5)	0.1405	0.3056	0.4998	0.4998	0.4998	0.5567
	mep(0.99)	0.0730	0.2191	0.4753	0.4753	0.4753	0.5284
	fep(0.5)	0.1003	0.2556	0.4836	0.4836	0.4836	0.5409
	fep(0.99)	0.0009	0.0781	0.3902	0.3902	0.3921	0.4508

(c) CSP_{freq} values (for selection principle MEA struct.).

Approach	Errors	Shape Level					
		0	1	2	3	4	5
SCFG	—	0.0374	0.1276	0.2973	0.2973	0.2977	0.3130
	mep(0.5)	0.0273	0.1045	0.2779	0.2779	0.2783	0.2908
	mep(0.99)	0.0083	0.0541	0.2007	0.2007	0.2007	0.2173
	fep(0.5)	0.0134	0.0795	0.2473	0.2473	0.2473	0.2603
	fep(0.99)	0.0000	0.0009	0.0120	0.0120	0.0120	0.0227
LSCFG	—	0.1729	0.3158	0.4300	0.4300	0.4300	0.4762
	mep(0.5)	0.1322	0.2728	0.4374	0.4374	0.4374	0.4859
	mep(0.99)	0.0693	0.1914	0.4101	0.4101	0.4101	0.4558
	fep(0.5)	0.0957	0.2261	0.4207	0.4207	0.4207	0.4642
	fep(0.99)	0.0009	0.0633	0.3264	0.3264	0.3269	0.3648

(d) CSP_{freq} values (for selection principle Centroid).

Table 8.5.: **Comparison of sampling quality for tRNAs.** Tables record specific values related to shapes of predictions and sampled structures, obtained from our tRNA database. All results were computed by 10-fold cross-validation procedures, using sample size 1000 and $min_{hel} = min_{HL} = 1$.

Similar results are observed for 5S rRNAs (see Tables D.4a to D.4d), where for all four prediction selecting principles, the CSP_{freq} values (for all shape levels in case of MP predictions and at least for shape levels 1 to 5 for all other prediction types) generally do not get significantly worse when applying the LSCFG sampling approach with inside values disturbed according to mep(prob) for any percentage $prob \in (0, 1)$ or according to the more intense relative disturbance variant fep(prob) for moderate values $prob \in (0, 1)$ (of up to about $prob = 0.75$).

Moreover, comparing the discussed CSP_{freq} results for the LSCFG variant to the corresponding ones for the conventional SCFG approach, we get additional evidence that the length-independent sampling method reacts stronger to relative disturbances in the underlying ensemble distribution for a given sequence than its length-dependent counterpart. As already mentioned, this is due to the fact that the ensemble distribution considered in the length-dependent case is much more centered due to the more explicit (length-dependently trained) grammar parameters, such that randomly generated errors on particular probabilities carry less weight.

Now, let us consider the three remaining specific values CSO_{freq}, CS_{num} and DS_{num} that can be used to assess the overall quality of generated sample sets rather than the accuracy of corresponding selected predictions. Basically, the obtained CSO_{freq} and CS_{num} results for tRNAs and 5S rRNAs (as reported in Tables D.3e to D.3f and Tables D.4e to D.4f), respectively, show a similar picture and thus yield similar conclusions as the corresponding CSP_{freq} values discussed above. As a consequence to the fact that for larger relative error percentages prob, for the less intensive disturbance variant defined by mep(prob) and especially for the more grave version implied by fep(prob), the resulting values for CSO_{freq} and CS_{num} usually get smaller, the corresponding DS_{num} values inevitably increase with growing disturbance influences imposed by mep(prob) and especially fep(prob) (see Tables D.3g and D.4g). This actually means that the diversity within the generated sample sets generally gets greater as the overall sampling quality (with respect to occurrences of the correct structure in the sample) decreases, which could be fully expected.

Approach	Errors	Shape Level					
		0	1	2	3	4	5
SCFG	—	0.0000	0.0026	0.0052	0.0131	0.0366	0.7110
	mep(0.5)	0.0000	0.0009	0.0026	0.0113	0.0287	0.7128
	mep(0.99)	0.0000	0.0026	0.0044	0.0095	0.0227	0.6919
	fep(0.5)	0.0000	0.0017	0.0043	0.0113	0.0374	0.6954
	fep(0.99)	0.0000	0.0000	0.0000	0.0017	0.0096	0.5474
LSCFG	—	0.2141	0.4256	0.4744	0.4900	0.9408	0.9843
	mep(0.5)	0.2141	0.4256	0.4744	0.4900	0.9408	0.9843
	mep(0.99)	0.1941	0.4221	0.4761	0.4892	0.9452	0.9852
	fep(0.5)	0.2124	0.4248	0.4726	0.4883	0.9417	0.9852
	fep(0.99)	0.0209	0.3029	0.3725	0.4186	0.8529	0.9809

(a) CSP_{freq} values (for selection principle MP struct.).

Approach	Errors	Shape Level					
		0	1	2	3	4	5
SCFG	—	0.0000	0.0026	0.0052	0.0131	0.0357	0.7128
	mep(0.5)	0.0000	0.0009	0.0026	0.0122	0.0305	0.7180
	mep(0.99)	0.0000	0.0026	0.0044	0.0105	0.0235	0.6902
	fep(0.5)	0.0000	0.0017	0.0043	0.0113	0.0383	0.6971
	fep(0.99)	0.0000	0.0000	0.0000	0.0035	0.0200	0.5439
LSCFG	—	0.2002	0.4256	0.4700	0.4866	0.9417	0.9861
	mep(0.5)	0.1332	0.3960	0.4439	0.4587	0.9434	0.9869
	mep(0.99)	0.0365	0.3630	0.4308	0.4491	0.9304	0.9861
	fep(0.5)	0.0801	0.3847	0.4404	0.4561	0.9400	0.9861
	fep(0.99)	0.0035	0.1497	0.2106	0.3325	0.5440	0.9730

(b) CSP_{freq} values (for selection principle MF struct.).

Approach	Errors	Shape Level					
		0	1	2	3	4	5
SCFG	—	0.0000	0.0000	0.0000	0.0000	0.0261	0.3821
	mep(0.5)	0.0000	0.0000	0.0000	0.0000	0.0209	0.3698
	mep(0.99)	0.0000	0.0000	0.0000	0.0000	0.0122	0.3003
	fep(0.5)	0.0000	0.0000	0.0000	0.0000	0.0252	0.3438
	fep(0.99)	0.0000	0.0000	0.0000	0.0000	0.0026	0.0444
LSCFG	—	0.1062	0.3891	0.4291	0.4378	0.9051	0.9835
	mep(0.5)	0.1010	0.3751	0.4134	0.4239	0.8921	0.9782
	mep(0.99)	0.0392	0.3429	0.3986	0.4213	0.8712	0.9791
	fep(0.5)	0.0740	0.3839	0.4239	0.4387	0.8877	0.9791
	fep(0.99)	0.0017	0.1358	0.1863	0.2942	0.4970	0.9634

(c) CSP_{freq} values (for selection principle MEA struct.).

Approach	Errors	Shape Level					
		0	1	2	3	4	5
SCFG	—	0.0000	0.0000	0.0000	0.0000	0.0104	0.1097
	mep(0.5)	0.0000	0.0000	0.0000	0.0000	0.0104	0.1062
	mep(0.99)	0.0000	0.0000	0.0000	0.0000	0.0078	0.0827
	fep(0.5)	0.0000	0.0000	0.0000	0.0000	0.0061	0.0932
	fep(0.99)	0.0000	0.0000	0.0000	0.0000	0.0009	0.0078
LSCFG	—	0.0966	0.2916	0.3238	0.3316	0.8703	0.9686
	mep(0.5)	0.0879	0.3142	0.3516	0.3621	0.8625	0.9686
	mep(0.99)	0.0322	0.2924	0.3377	0.3595	0.8294	0.9651
	fep(0.5)	0.0662	0.3194	0.3551	0.3638	0.8512	0.9695
	fep(0.99)	0.0017	0.1053	0.1471	0.2219	0.4831	0.9339

(d) CSP_{freq} values (for selection principle Centroid).

Table 8.6.: **Comparison of sampling quality for 5S rRNAs.** Tables record specific values related to shapes of predictions and sampled structures, obtained from our 5S rRNA database. All results were computed by 10-fold cross-validation procedures, using sample size 1000 and $\min_{hel} = \min_{HL} = 1$.

8.5. Conclusions

During our analysis (on the basis of trusted sets of tRNA and 5S rRNA data), we immediately observed that even incorporating only rather small absolute errors into (all or particular instances of the) inside values causes problematic disturbances of the resulting sampling probabilities that generally lead to the generation of useless sample sets. This can be assumed to be due to the fact that the installation of absolute errors usually makes it impossible for the employed sampling strategy to identify which ones of the considered inside probabilities for a given input sequence must originally (that is, in the exact case) have been equal or unequal to zero, which inevitably results in a misguided behavior of the strategy, as it is no longer ensured that it creates only reasonable substructures for a considered sequence fragment.

However, both SCFG approaches (length-dependent and traditional one) behave rather resistant to disturbances of the needed conditional sampling probabilities that are caused by generating (moderate) relative errors on all (and also only on particular) inside values for a given input sequence. In general, even large relative errors seem to have only small negative impact on both the predictive accuracy and the overall quality of generated sample sets. That is, the reaction of the (L)SCFG based statistical sampling algorithm to the relative disturbances is fair enough to still obtain meaningful structure predictions, especially if the most likely structure of the sample is selected as predicted folding – in strong analogy to conventional SCFG based DP algorithms. Furthermore, the overall quality of the resulting sample sets is still acceptable such that they might often also be used for further applications (like for example probability profiling for specific loop types).

Consequently, it seems reasonable to believe that the needed sampling probabilities do not necessarily have to be computed in the exact way, but it may probably suffice to only (adequately) approximate them. In fact, the worst-case time complexity of any particular (L)SCFG based sampling method could potentially be reduced by developing a suitable approximation procedure (or at least an adequate heuristic method) for the computation of the needed sampling probabilities. Nevertheless, an appropriate approximation ratio (or at least an acceptable ratio of correctly and incorrectly computed zero values) should be attempted to ensure that the sampling quality remains sufficiently high, as indicated by the experimental disturbance analysis results discussed within this chapter.

Chapter 9

Heuristic Statistical Sampling Methods for Efficient Secondary Structure Prediction

This chapter introduces several variants of a novel heuristic method for predicting RNA secondary structures that effectively reduce the worst-case time complexity by a linear factor compared to all other modern approaches. It actually requires only $\mathcal{O}(n^2)$ time and storage, respectively, for molecule length n. Principally, all proposed variants rely on a probabilistic statistical sampling approach based on our sophisticated SCFG. This means they generate a random set of candidate structures according to a "noisy" distribution for the given input sequence. This distribution is efficiently obtained by heuristically approximating the corresponding inside-outside values for that sequence. However, a corresponding prediction – most reasonably the most likely candidate folding, in direct analogy to conventional structure prediction via SCFGs – can still be efficiently derived from a generated sample.

Basically, any of the suggested heuristic preprocessing variants for time-reduced inside-outside calculations may be combined with different sampling strategies for the stochastic traceback step, where any strategy may only need $\mathcal{O}(n^2)$ time for generating a random candidate structure for a sequence of length n in order to not diminish to efficiency gain of heuristic preprocessing. Therefore, we will not only consider the well-established strategy implemented in `Sfold`, but also introduce a novel one that is supposed to cope better with noisy ensemble distributions.

Note that a valuable benefit of our heuristic prediction approach is that sampling can easily be parallelized and actually be done in-place if only the most likely folding is to be computed; then, only the most likely candidate structure generated so far needs to be stored and finally communicated. This allows us to efficiently handle increased sample sizes that might be necessary to achieve competitive prediction accuracy in connection with our roughest heuristic variants.

As we will see, our heuristic statistical sampling methods might still yield highly accurate predictions, but may indeed require the consideration of larger sample sizes. In fact, evaluations on the basis of different data sets indicate that at least on the highest shape abstraction level (where only nesting and adjacency of helical regions are accounted for), the correct structure may still be predicted in an acceptable number of times with our heuristic methods. Compared to several leading (physics-based and probabilistic) RNA secondary structure prediction tools, the resulting accuracies may be superior for particular RNA types, but inferior for others, validating the application of this accelerated sampling approach in practice. However, there currently exits only a purposive proof-of-concept implementation in Wolfram Mathematica, such that no highly efficient tool implementing this heuristic approach can be provided to date.

9.1. Motivation and Objectives

In this section, we finally want to address the main objective of this theses, namely the development of a novel method for predicting the secondary structure of RNA molecules, which in the worst-case is more efficient than any precise approach, while still ensuring an acceptable predictive accuracy.

This is motivated for both mathematical and practical reasons. In fact, the empirical studies on the influence of different kinds and levels of disturbances in the sampling probabilities to the quality of generated sample sets performed in the last section yielded the promising conclusion that the worst-case time complexity of any particular (L)SCFG based sampling strategy could potentially be reduced by developing a suitable approximation procedure (or at least an adequate heuristic method) for the computation of the needed sampling probabilities. Most importantly, we already brought up some useful ideas in connection with relevant aspects that should be considered for the construction of a corresponding time-reduced sampling method in order to ensure that the sampling quality (and resulting predictive accuracy) remains sufficiently high. On the other hand, due to the often quite large sizes of native RNA molecules considered in practice, such an accelerated method meets exactly the demands imposed by biologists on computational prediction procedures: rather getting moderately less accurate (but still good quality) results in less time than needing significantly more time for obtaining results that are expectedly not considerably more accurate.

For these reasons, the objective of this chapter can be described as follows: Building on the observations made in Chapter 8 in connection with "skewed" ensemble distributions and the corresponding suggestions for developing an appropriate heuristic method, we formally describe an innovative way to reduce the worst-case time complexity of (L)SCFG based statistical sampling by a linear factor, making it possible to predict for instance the most probable structure among all candidate foldings[1] for a given input sequence of length n (in direct analogy to conventional structure prediction via SCFGs) with only $\mathcal{O}(n^2)$ time and space requirements.

This complexity improvement is basically realized by employing an appropriate heuristic instead of the corresponding exact algorithm for preprocessing the input sequence, that is for deriving a "noisy" distribution on the entire structure ensemble for the input sequence – induced by heuristic approximations of the corresponding inside and outside probabilities. From this distribution, candidate structures can be efficiently sampled[2]. Moreover, we will consider two different sampling strategies: First, (a slight modification of) the widely known sampling procedure from [DL03] (as devised in Section 8.3.2) which basically generates a random structure from outside to inside. And second, a novel alternative strategy that obeys to contrary principles and employs a reverse course of action (from inside to outside) but may be expected to take more advantage of the approximative preprocessing.

Notably, the introduced sampling heuristics will be evaluated based on quite substantial and interesting results, in order to be able to draw reliable conclusions on the effect of approximative preprocessing steps. This chapter also includes a comparison of some of the obtained results to corresponding ones derived with a number of popular RNA tools (that require cubic time in the worst-case) for further judgement of the competitiveness of our heuristic method (needing only quadratic time in the worst-case) with respect to prediction accuracy.

[1] Recall that in case of an unambiguous SCFG modeling all RNA sequences, the most probable structure for a particular sequence is equal to the most likely parse tree for that sequence.

[2] For the practically oriented reader, it may be of interest that with purposive proof-of-concept implementations (in Wolfram Mathematica 7.0), for instance the overall preprocessing time for *E.coli* tRNA^{Ala} (of length $n = 76$) could be reduced from 49.0 (traditional cubic algorithm) to only 3.7 (new quadratic strategy) seconds.

9.2. Outline

As we will see, even building on our new heuristic preprocessing step, both sampling strategies can be applied to obtain MP structure predictions of respectable accuracy. In principle, for sufficiently large sample sizes we may obtain a similar high predictive accuracy as in the case of exact calculations, at least for particular RNA types. The seemingly sole pitfall is that due to the noisy ensemble distribution resulting from approximative computations, the resulting samples are no longer guaranteed to primarily contain rather likely structures with respect to the *exact* distribution of feasible foldings for a given input sequence. Thus, we usually have to generate more candidate structures (that is, consider larger sample sizes) in oder to ensure reproducible structure predictions. However, this is quite unproblematic in practice for the following reasons: First, we can generate the candidate structures *in-place*. This means only the so far most probable structure needs to be stored, such that large sample sizes give no rise to memory consumption. Second, generating samples can easily be parallelized on modern multi-core architectures or grids, such that increasing the sample size must not imply a longer computation time.

The rest of this chapter is organized as follows: Section 9.3 describes all facts concerning the approximative preprocessing step that needs to be applied for decreasing the worst-case time requirements. The alternative novel sampling strategy intended to match well with our heuristic preprocessing methods is described in Section 9.4, where it is also opposed to the well-established strategy devised in Section 8.3.2. Throughout Section 9.5, the overall quality of generated sample sets and especially their applicability to RNA structure prediction are investigated. Notably, we present interesting experiments which show how the prediction accuracy grows with the considered sample size, together with considerations on how an efficient implementation can deal with large sample sets. Furthermore, we quantify the decline in accuracy of selected (MP) predictions that results when employing the different heuristic preprocessing variants rather than a traditional (cubic) inside-outside algorithm. Notably, these examinations also include results on the abstraction level of shapes of predicted structures. A corresponding evaluation of the competitiveness of the proposed heuristic sampling approach by comparison to several leading RNA secondary structure prediction tools will be performed in Section 9.6. Finally, Section 9.7 concludes this chapter and hints at some interesting matters for further research.

9.3. Heuristic Preprocessing

The easiest (and perhaps only) way to reduce the overall time complexity for generating a statistical sample of secondary structures for a given input sequence r is to improve the worst-case time requirements of the preprocessing step, where we have to compute all inside (and outside) probabilities for r under the assumption of the probabilistic model implied by grammar $\mathcal{G}_s$. Therefore, our aim is to lower the $\mathcal{O}(n^3)$ time complexity for preliminary inside (and outside) calculations to $\mathcal{O}(n^2)$, such that the preprocessing eventually has the same worst-case time requirements as the subsequent sampling process for constructing a constant number of random secondary structures for input sequence r.

9.3.1. Basic Idea

The main idea for reaching this time complexity reduction by a factor n in the worst-case is actually quite simple: Instead of deriving the traditional inside values $\alpha_X(i,j)$ and (for some particular sampling strategies maybe also) the corresponding outside probabilities $\beta_X(i,j)$, with $X \in \mathcal{I}_{\mathcal{G}_s}$, for any combination of start position i and end position j, $1 \leqslant i,j \leqslant n$, we consider

only their distance $\mathrm{dist} = j - i + 1$. This means we abstract from the actual position of subword $R_{i,j} = r_i \dots r_j$ in the input sequence and consider only its length

$$\mathrm{dist} = |r_i \dots r_j| = j - i + 1,$$

such that for any $X \in \mathcal{I}_{\mathcal{G}_s}$,

- we do not need to calculate $\mathcal{O}(n^2)$ values $\alpha_X(i,j)$ (and $\beta_X(i,j)$) for $1 \leqslant i,j \leqslant n$,
- but only $\mathcal{O}(n)$ values $\alpha_X(\mathrm{dist})$ (and $\beta_X(\mathrm{dist})$) for $0 \leqslant \mathrm{dist} \leqslant n$.

However, the problem with this approach is that if we do not know the start and end positions i and j of a particular subword of r but only its length dist, then we are inevitably forced to additionally abstract from the actual input sequence, since we are not able to identify the corresponding subword from the set

$$\{r_i \dots r_j \mid j - i + 1 = \mathrm{dist}\}$$

of all subwords having length dist. This means for deriving the probabilities $\alpha_X(\mathrm{dist})$ (and $\beta_X(\mathrm{dist})$) for each $X \in \mathcal{I}_{\mathcal{G}_s}$ and $0 \leqslant \mathrm{dist} \leqslant n$, we need to use reasonable alternative terms (for example, preliminary defined or sequence-dependent averaged terms) rather than the trained emission probabilities during the corresponding inside (and outside) calculations. Hence, for any $\mathrm{dist} \in \{0, \dots n\}$, the resulting inside values $\alpha_X(\mathrm{dist})$ (and outside values $\beta_X(\mathrm{dist})$) are only approximated versions of the corresponding (diverse) exact probabilities $\alpha_X(i,j)$ (and $\beta_X(i,j)$) with $j - i + 1 = \mathrm{dist}$. This drawback can be seen as the inevitable price to be paid for reducing the time complexity of the corresponding inside (and outside) computations from $\mathcal{O}(n^3)$ to $\mathcal{O}(n^2)$. Altogether, this approach manages to improve the worst-case time complexity of the preprocessing step to that of the sampling procedure, but at the cost of an decline in accuracy of the stochastic model employed for sampling particular structures for the given input sequence.

Note that it is also possible to combine both alternatives, that is we can use the traditional algorithms to calculate exact values $\alpha_X(i,j)$ (and $\beta_X(i,j)$) within a window of size W_e, that is for $j - i + 1 \leqslant W_e$ (and $j - i + 1 \geqslant n - W_e$) and then derive the remaining values for $W_e < \mathrm{dist} \leqslant n$ (and $0 \leqslant \mathrm{dist} < n - W_e$) in an approximate fashion, by employing the time-reduced variant for obtaining $\alpha_X(\mathrm{dist})$ (and $\beta_X(\mathrm{dist})$) for each $X \in \mathcal{I}_{\mathcal{G}_s}$. For a constant value of W_e, this effectively yields an improvement in the time complexity of the corresponding complete inside computation, which is then given by $\mathcal{O}(n^2 \cdot W_e)$), whereas the time requirements for such a mixed outside calculation are bounded by $\mathcal{O}(n^3)$, no matter if W_e is constant or not.

According to the considered parameter W_e, there consequently are three possible cases when considering an input sequence r of length n:

- $-\infty \leqslant W_e < 0$ means that all values are only approximated (that is, there exists no window at all). In this case, there results a worst-case time complexity of $\mathcal{O}(n^2)$ for both the inside and outside calculations.
- $0 \leqslant W_e < n$ means that within a window of size W_e, the values are derived exactly, whereas outside this window, they are only determined in an approximate fashion. The worst-case time complexity of the resulting complete inside calculations is then given by $\mathcal{O}(n^2 \cdot W_e + n^2) \subseteq \mathcal{O}(n^3)$ and that of the corresponding outside computations by $\mathcal{O}(n^3 + n^2) = \mathcal{O}(n^3)$.
- $n \leqslant W_e \leqslant \infty$ means that all values are derived exactly (that is, the window covers the whole input sequence). Obviously, the resulting time requirements for computing all needed values are equal to that of the traditional inside and outside algorithms and are thus equal to $\mathcal{O}(n^3)$.

A detailed description (by means of pseudocode) on how the inside and outside variables for a given input sequence based on grammar $\mathcal{G}_r$ can be computed according to any chosen value of W_e will follow in Section 9.3.3. However, we first want to discuss and formally describe the needed approximation details.

9.3.2. Approximation of Emission Probabilities

It should be clear that in contrast to the exact inside and outside algorithms, the approximation variants do not know both start position i and end position j of a given sequence fragment $R_{i,j}$ but only its length $j-i+1$, so they obviously can not get information on the actual nucleotides r_i and r_j at the ends of the considered sequence fragment. Consequently, they are forced to abstract from the actual input sequence, as it is not known which emission probabilities have to be used in the different scanning steps of a corresponding probabilistic Earley parser (see Section 7.3.1). Fortunately, the change from exact calculations for actual sequence fragments to approximate ones that only rely on the length of the considered fragments does not impose any problems with the used transition probabilities as those only (or not even, under the assumption of the plain SCFG model) require exactly that length information.

However, the resulting inside and outside values strongly depend on the incorporation of the correct emission probabilities, especially on those for valid base pairs; a fact that gains more importance in cases where less base pairs are considered valid ones. For this reason, it does not seem sufficient to simply use some preliminary defined fix but completely sequence-independent emission probabilities (like for example the average or the *stickiness*[3] resulting from the trained emission parameters) instead of the correct terms $Pr_{em}(r_i)$ and $Pr_{em}(r_i r_j)$ depending on the actual nucleotides at positions i and j of the input sequence (as done in Algorithms 9 and 12 given in Section 7.3.1). In principle, we need to determine some approximated terms for the emissions of unpaired bases and base pairs, respectively, that

- do not depend on the positions of subwords within the overall input word, but
- should at least depend on the lengths of the corresponding subwords,

where it is strongly recommended to make sure that as much information on the composition of the given input word as possible is incorporated into these approximated terms.

Accordingly, we decided to perform an additional preprocessing of the given RNA sequence r in order to determine the relative frequencies of all four (unpaired) bases and all individual valid base pairs on distinct sequence fragments $R_{i,j}$ of length $\mathtt{dist} = j-i+1$, for each possible fragment length $\mathtt{dist} \in \{0,\ldots,n\}$. In particular, we need to compute all different values of

$$rf_{em}^{ub}(u) := \frac{c_{em}(u)}{\sum_{x \in \Sigma_{\mathcal{G}_r}} c_{em}(x)}$$

and

$$rf_{em}^{bp}(p_1p_2, \mathtt{dist}) := \frac{c_{em}(p_1p_2, \mathtt{dist})}{\sum_{\text{valid pair } x_1x_2 \in \Sigma_{\mathcal{G}_r}^2} c_{em}(x_1x_2, \mathtt{dist})},$$

where

$$c_{em}(u) := \begin{cases} \mathtt{card}(\{i \mid 1 \leqslant i \leqslant n \text{ and } r_i = u\}), & \text{if } u \in \Sigma_{\mathcal{G}_r}, \\ 0, & \text{else,} \end{cases}$$

[3] Briefly, the stickiness is defined by a parameter p to specify the probability that two random bases can form a hydrogen bond. For instance, under the assumption that only canonical (that is, Watson-Crick and wobble GU base pairs) are allowed, then this probability is obviously given by $p := 2 \cdot (p(a) \cdot p(u) + p(c) \cdot p(g) + p(g) \cdot p(u))$, where $p(X)$, $X \in \{a, c, g, u\}$, denotes the probability of base X (for example, derived from a particular RNA database). Note that the stickiness p accounts only for the possible formation of base pairs in a secondary structure, but the thermodynamic effects of different strengths of the different types of base pairs are completely ignored by this parameter. For more information, we refer to [Les74, Neb04b].

and

$$c_{em}(p_1p_2, dist) := \begin{cases} \mathtt{card}(\{i,j \mid 1 \leqslant i \leqslant n \text{ and } \\ \qquad i+1+\min_{HL} \leqslant j \leqslant n \text{ and } \\ \qquad (j-i+1) = dist \text{ and } \\ \qquad r_ir_j = p_1p_2\}), & \text{if } p_1p_2 \in \Sigma^2_{\mathcal{G}_r} \text{ valid and } 0 \leqslant dist \leqslant n, \\ 0, & \text{else.} \end{cases}$$

This can efficiently be done in time $\mathcal{O}(n^2)$ for an input sequence r of length n with a straightforward counting procedure. Note that these relative frequencies indeed provide as much sequence information as possible if only the lengths rather than the actual start and end positions of corresponding fragments are known. Therefore, on their basis, we can eventually define the following sequence-dependent averaged emission probability terms:

$$Pr^{ub,a}_{em}(dist = 1) := \sum_{u \in \Sigma_{\mathcal{G}_r}} Pr_{em}(u) \cdot rf^{ub}_{em}(u),$$

$$Pr^{bp,a}_{em}(dist) := \sum_{\text{valid pair } p_1p_2 \in \Sigma^2_{\mathcal{G}_r}} Pr_{em}(p_1p_2) \cdot rf^{bp}_{em}(p_1p_2, dist).$$

In fact, these expected emission probabilities can reliably be used in the approximate versions of our inside and outside algorithms (see Algorithms 21 and 25 given in Section 9.3.3).

9.3.3. Computation of (Approximated) Inside and Outside Probabilities

The aim of this section is to formally describe how to determine all inside and outside variables for a given sequence r in the previously sketched approximate fashion. In analogy to Sections 6.4.1 and 7.3.1, we will again use the secondary structure grammar $\mathcal{G}_s$ given in Definition 6.3.1 as basis for constructing a grammar specific semiring parser implementing Earley's algorithm.

9.3.3.1. Calculating (Approximated) Inside and Outside Values of Items

First, recall that the semiring parser described in Section 7.3.1 computes inside and outside values for items

$$[i, \text{ind}(\textit{rule}), j],$$

where for a given input word r of length n, i and j, $1 \leqslant i, j \leqslant n+1$, define positions in r and $\text{ind}(\textit{rule})$ denotes the index of production $\textit{rule} \in \mathcal{R}_{\mathcal{G}_s,\bullet}$ in an appropriate ordering of $\mathcal{R}_{\mathcal{G}_s,\bullet}$. While we can still use the same ordering of $\mathcal{R}_{\mathcal{G}_s,\bullet}$ when abstracting from the start and end positions of considered sequence fragments, we however have to consider a different class of items. In fact, the accelerated version of our semiring parser operates on items of the form

$$[dist, \text{ind}(\textit{rule})],$$

where for a given input word r of length n, $dist \in \{0, \ldots, n\}$ defines a subword length.

Therefore, for computing the desired values of all items according to a given input word r, we basically need to employ a number of (more or less) modified versions of the inside and outside algorithms presented in Section 7.3.1. Actually, we can use Algorithms 18 and 23 for deriving exact values within a window of size W_e and Algorithms 19 and 24 for calculating the remaining values in an approximate fashion in order to reduce the overall time complexity.

Algorithm 18 Computation of exact inside values

```
procedure INSIDEalg()
if −∞ ⩽ W_e < 0 then
  return
end if
for j = 1 to n + 1 do
  if (j − j + 1) ⩽ W_e + 1 then
    for p = 1 to card(R_{G_s}) do
      Consider rule = ind^{-1}(p, 0) ∈ R_{G_s,•} /*i.e. the rule having index (p, 0) in our ordering*/
      Predict(j, p)
    end for
  end if
  for i = j to 1 do
    if (j − i + 1) ⩽ W_e + 1 then
      for p = 1 to card(R_{G_s}) do
        for q = 0 to k(p) do
          Consider rule = ind^{-1}(p, q) ∈ R_{G_s,•} /*i.e. the rule having index (p, q) in our ordering*/
          if rule = X → γ w_{j−1} • δ then
            Scan(i, j, p, q)
          else if rule = X → γ Y • δ then
            Complete(i, j, p, q)
          end if
        end for
      end for
    end if
  end for
end for
return
end procedure
```

Algorithm 19 Computation of approximated inside values

procedure INSIDEalgAPPROX()
if $n \leqslant W_e \leqslant +\infty$ **then**
  return
end if
if not $-\infty \leqslant W_e < 0$ **then**
  DeriveAllCorrespondingApproxIOvaluesForItems(inside)
end if
if $\mathtt{dist} = 1 > W_e + 1$ **then**
  for $p = 1$ **to** $\mathtt{card}(\mathcal{R}_{\mathcal{G}_s})$ **do**
    Consider *rule* $= \mathrm{ind}^{-1}(p, 0) \in \mathcal{R}_{\mathcal{G}_s,\bullet}$ /*i.e. the rule having index $(p, 0)$ in our ordering*/
    Predict(p)
  end for
end if
for $\mathtt{dist} = 1$ **to** $n + 1$ **do**
  if $\mathtt{dist} > W_e + 1$ **then**
    for $p = 1$ **to** $\mathtt{card}(\mathcal{R}_{\mathcal{G}_s})$ **do**
      for $q = 0$ **to** $k(p)$ **do**
        Consider *rule* $= \mathrm{ind}^{-1}(p, q) \in \mathcal{R}_{\mathcal{G}_s,\bullet}$ /*i.e. the rule having index (p, q) in our ordering*/
        if *rule* $= X \to \gamma w_{j-1} \bullet \delta$ **then**
          Scan(dist, p, q)
        else if *rule* $= X \to \gamma Y \bullet \delta$ **then**
          Complete(dist, p, q)
        end if
      end for
    end for
  end if
end for
return
end procedure

Algorithm 20 Predicting inside items (approximated)

procedure Predict(p)
Consider *rule* $= X \to \bullet\gamma = \mathrm{ind}^{-1}(p, 0) \in \mathcal{R}_{\mathcal{G}_s,\bullet}$ /*i.e. the rule having index $(p, 0)$ in our ordering*/
if $\gamma = \epsilon$ /**rule* is ϵ-rule*/ **then**
  $\mathrm{inside}[\mathtt{dist} = 1, (p, 0)] = \mathrm{Pr}_{\mathtt{tr}}(X \to \gamma, 0)$
else
  $\mathrm{inside}[\mathtt{dist} = 1, (p, 0)] = 1$
end if
return
end procedure

Algorithm 21 Scanning inside items (approximated)

procedure Scan($\mathtt{dist}, \mathrm{p}, \mathrm{q}$)
Consider *rule* $= \mathrm{X} \to \gamma w_{\mathrm{j}-1} \bullet \delta = \mathrm{ind}^{-1}(\mathrm{p}, \mathrm{q}) \in \mathcal{R}_{\mathcal{G}_s,\bullet}$ /*i.e. the rule having index (p, q) in our ordering*/
if $w_{\mathrm{j}-1} =' \circ '$ **then**
  $\mathrm{inside}[\mathtt{dist}, (\mathrm{p}, \mathrm{q})] = \mathrm{Pr}_{em}^{ub,a}() \cdot \mathrm{inside}[\mathtt{dist} - 1, (\mathrm{p}, \mathrm{q} - 1)]$
else if $w_{\mathrm{j}-1} =' ($' **then**
  $\mathrm{inside}[\mathtt{dist}, (\mathrm{p}, \mathrm{q})] = \mathrm{inside}[\mathtt{dist} - 1, (\mathrm{p}, \mathrm{q} - 1)]$
else if $w_{\mathrm{j}-1} =')'$ **then**
  $\mathrm{inside}[\mathtt{dist}, (\mathrm{p}, \mathrm{q})] = \mathrm{Pr}_{em}^{bp,a}(\mathtt{dist} - 1) \cdot \mathrm{inside}[\mathtt{dist} - 1, (\mathrm{p}, \mathrm{q} - 1)]$
end if
if $\mathrm{q} = \mathrm{k}(\mathrm{p})$ /*$rule = \mathrm{X} \to \gamma w_{\mathrm{j}-1}\bullet$, i.e. rule is completed in this scanning step*/ **then**
  $\mathrm{inside}[\mathtt{dist}, (\mathrm{p}, \mathrm{q})] = \mathrm{inside}[\mathtt{dist}, (\mathrm{p}, \mathrm{q})] \cdot \mathrm{Pr}_{\mathtt{tr}}(\mathrm{X} \to \gamma w_{\mathrm{j}-1}, \mathtt{dist} - 1)$
end if
return
end procedure

Algorithm 22 Completing inside items (approximated)

procedure Complete($\mathtt{dist}, \mathrm{p}, \mathrm{q}$)
Consider *rule* $= \mathrm{X} \to \gamma \mathrm{Y} \bullet \delta = \mathrm{ind}^{-1}(\mathrm{p}, \mathrm{q}) \in \mathcal{R}_{\mathcal{G}_s,\bullet}$ /*i.e. the rule having index (p, q) in our ordering*/
$\mathrm{inside}[\mathtt{dist}, (\mathrm{p}, \mathrm{q})] = \sum_{\mathrm{k}=1}^{\mathtt{dist}} \left(\mathrm{inside}[\mathrm{k}, (\mathrm{p}, \mathrm{q} - 1)] \cdot \left(\sum_{rule_{\mathrm{B}} \in \mathcal{R}_{\mathrm{B}}} \mathrm{inside}[\mathtt{dist} - \mathrm{k} + 1, \mathrm{ind}(rule_{\mathrm{B}})]\right)\right)$
if $\mathrm{q} = \mathrm{k}(\mathrm{p})$ /*$rule = \mathrm{X} \to \gamma \mathrm{Y}\bullet$, i.e. rule is completed*/ **then**
  $\mathrm{inside}[\mathtt{dist}, (\mathrm{p}, \mathrm{q})] = \mathrm{inside}[\mathtt{dist}, (\mathrm{p}, \mathrm{q})] \cdot \mathrm{Pr}_{\mathtt{tr}}(\mathrm{X} \to \gamma \mathrm{Y}, \mathtt{dist} - 1)$
end if
return
end procedure

Algorithm 23 Computation of exact outside values

procedure OUTSIDEalg()
if $-\infty \leqslant W_e < 0$ **then**
  return
end if
$\text{outside}[1, \text{ind}(S \to T\bullet), n+1] = 1$ /*initialization*/
for $j = n+1$ **to** 1 **do**
  for $i = 1$ **to** j **do**
    if $(j - i + 1) \geqslant n + 1 - W_e$ **then**
      for $p = \texttt{card}(\mathcal{R}_{\mathcal{G}_s})$ **to** 1 **do**
        for $q = k(p)$ **to** 0 **do**
          Consider *rule* $= \text{ind}^{-1}(p, q) \in \mathcal{R}_{\mathcal{G}_s, \bullet}$ /*i.e. the rule having index (p, q) in our ordering*/
          if *rule* $= X \to \bullet\gamma$ **then**
            /*PredictReverse():*/
            Do nothing
          else if *rule* $= X \to \gamma w_j \bullet \delta$ **then**
            ScanReverse(i, j, p, q)
          else if *rule* $= X \to \gamma Y \bullet \delta$ **then**
            CompleteReverse(i, j, p, q)
          end if
        end for
      end for
    end if
  end for
end for
return
end procedure

Algorithm 24 Computation of approximated outside values

procedure OUTSIDEalgAPPROX()
if $n \leqslant W_e \leqslant +\infty$ **then**
  return
end if
if not $-\infty \leqslant W_e < 0$ **then**
  DeriveAllCorrespondingApproxIOvaluesForItems(outside)
end if
$\text{outside}[n+1, \text{ind}(S \to T\bullet)] = 1$ /*initialization*/
for $\text{dist} = n+1$ **to** 1 **do**
  for $p = \texttt{card}(\mathcal{R}_{\mathcal{G}_s})$ **to** 1 **do**
    for $q = k(p)$ **to** 0 **do**
      Consider *rule* $= \text{ind}^{-1}(p,q) \in \mathcal{R}_{\mathcal{G}_s,\bullet}$ /*i.e. the rule having index (p,q) in our ordering*/
      if *rule* $= X \to \bullet\gamma$ **then**
        /*PredictReverse():*/
        Do nothing
      else if *rule* $= X \to \gamma w_j \bullet \delta$ **then**
        ScanReverse(dist, p, q)
      else if *rule* $= X \to \gamma Y \bullet \delta$ **then**
        CompleteReverse(dist, p, q)
      end if
    end for
  end for
end for
return
end procedure

Algorithm 25 Scanning outside items (approximated)

procedure ScanReverse(dist, p, q)
Consider *rule* $= X \to \gamma w_j \bullet \delta = \text{ind}^{-1}(p,q) \in \mathcal{R}_{\mathcal{G}_s,\bullet}$ /*i.e. the rule having index (p,q) in our ordering*/
if $w_j =' \circ '$ **then**
  $\text{outside}[\text{dist}, (p, q-1)] = \text{Pr}_{em}^{ub,a}() \cdot \text{outside}[\text{dist}+1, (p,q)]$
else if $w_j =' (\,'$ **then**
  $\text{outside}[\text{dist}, (p, q-1)] = \text{outside}[\text{dist}+1, (p,q)]$
else if $w_j =')\,'$ **then**
  $\text{outside}[\text{dist}, (p, q-1)] = \text{Pr}_{em}^{bp,a}(\text{dist}) \cdot \text{outside}[\text{dist}+1, (p,q)]$
end if
if $q = k(p)$ /*rule $= X \to \gamma w_j \bullet$, i.e. rule is completed in this scanning step*/ **then**
  $\text{outside}[\text{dist}, (p, q-1)] = \text{outside}[\text{dist}, (p, q-1)] \cdot \text{Pr}_{tr}(X \to \gamma w_j, \text{dist})$
end if
return
end procedure

Algorithm 26 Completing outside items (approximated)

procedure CompleteReverse($\mathtt{dist}, p, q$)
Consider $rule = X \to \gamma Y \bullet \delta = \mathrm{ind}^{-1}(p, q) \in \mathcal{R}_{\mathcal{G}_s,\bullet}$ /*i.e. the rule having index (p, q) in our ordering*/
if $q = k(p)$ /*$rule = X \to \gamma Y\bullet$, i.e. rule is completed*/ **then**
$\mathtt{fact} = \mathrm{Pr}_{\mathtt{tr}}(X \to \gamma Y, \mathtt{dist} - 1)$
else
$\mathtt{fact} = 1$
end if
for $k = 1$ **to** $\mathtt{dist}$ **do**
if $k < n + 1 - W_e$ **then**
$\mathrm{outside}[k, (p, q-1)] = \mathrm{outside}[k, (p, q-1)] +$
$(\mathrm{outside}[\mathtt{dist}, (p, q)] \cdot (\sum_{rule_B \in \mathcal{R}_B} \mathrm{inside}[\mathtt{dist} - k + 1, \mathrm{ind}(rule_B)])) \cdot$
$\mathtt{fact}$
end if
if $\mathtt{dist} - k + 1 < n + 1 - W_e$ **then**
for $rule_B \in \mathcal{R}_B$ **do**
$\mathrm{outside}[\mathtt{dist} - k + 1, \mathrm{ind}(rule_B)] = \mathrm{outside}[\mathtt{dist} - k + 1, \mathrm{ind}(rule_B)] +$
$(\mathrm{outside}[\mathtt{dist}, (p, q)] \cdot \mathrm{inside}[k, (p, q-1)]) \cdot \mathtt{fact}$
end for
end if
end for
return
end procedure

Algorithm 27 Transforming exact inside outside values of items into averaged ones

procedure DeriveAllCorrespondingApproxIOvaluesForItems()
for $p = 1$ **to** $\mathtt{card}(\mathcal{R}_{\mathcal{G}_s})$ **do**
for $q = 0$ **to** $k(p)$ **do**
$\mathtt{index} = (p, q) = \mathrm{ind}(rule \in \mathcal{R}_{\mathcal{G}_s,\bullet})$
for $\mathtt{dist} = 1$ **to** $W_e + 1$ **do**
$\mathrm{inside}[\mathtt{dist}, \mathtt{index}] = \mathrm{corrApproxIOval}(\mathrm{inside}, \mathtt{index}, \mathtt{dist})$
end for
for $\mathtt{dist} = n + 1$ **to** $n + 1 - W_e$ **do**
$\mathrm{outside}[\mathtt{dist}, \mathtt{index}] = \mathrm{corrApproxIOval}(\mathrm{outside}, \mathtt{index}, \mathtt{dist})$
end for
end for
end for
return
end procedure

Algorithm 28 Transforming exact inside outside probabilities into averaged ones

procedure DeriveAllCorrespondingApproxIOvalues()
for $nt \in \mathcal{I}_{\mathcal{G}_s}$ **do**
  for $dist = 0$ **to** W_e **do**
    $\alpha_{nt}(dist) = \text{corrApproxIOval}(\alpha_{nt}, dist)$
  end for
  for $dist = n$ **to** $n - W_e$ **do**
    $\beta_{nt}(dist) = \text{corrApproxIOval}(\beta_{nt}, dist)$
  end for
end for
return
end procedure

Note that in order to switch from exact inside and outside calculation to the corresponding approximate variant, respectively, at a certain point – specifically once the considered window size for exact computations given by W_e has been reached – we have to transform the already determined exact values for items $[i, \text{ind}(\textit{rule}), j]$ depending on both the start position i and the end position j of the respective subword into new ones that only depend on the length $j - i + 1$ of the respective subword, that is into some sort of corresponding ones for items $[(j - i + 1), \text{ind}(\textit{rule})]$. This is due to the fact that the approximate inside and outside algorithms, respectively, do only consider the different subword lengths $dist = j - i + 1$, where i and j are actually not known. They can thus only use values of items $[dist, \text{ind}(\textit{rule})]$ for deriving the remaining ones.

In fact, Algorithm 27 illustrates how these transformations of already determined exact values of items into some sort of approximated (averaged) ones can be realized. Algorithm 28 can be used in the same way for obtaining some sort of approximated (averaged) conventional inside probabilities $\alpha_X(dist)$ or outside probabilities $\beta_X(dist)$, $X \in \mathcal{I}_{\mathcal{G}_s}$ and $0 \leqslant dist \leqslant n$, from the corresponding sets of exact ones.

Note that these methods effectively rely on the following definitions:

$$\begin{aligned}
\text{exactVals}(func, index, dist) := \{val \mid\ & val = func[i, index, j] \neq 0 \text{ and } 1 \leqslant i \leqslant n+1 \text{ and} \\
& i - 1 \leqslant j \leqslant n+1 \text{ and } (j - i + 1) = dist + 1\}, \\
\text{exactVals}(func_{nt}, dist) := \{val \mid\ & val = func_{nt}(i, j) \neq 0 \text{ and } 1 \leqslant i \leqslant n \text{ and} \\
& i - 1 \leqslant j \leqslant n \text{ and } (j - i + 1) = dist\},
\end{aligned}$$

as well as

$$\text{corrApproxIOval}(func, index, dist) := \begin{cases} \frac{\sum_{val \in \text{exactVals}(func, index, dist)} val}{\text{card}(\text{exactVals}(func, index, dist))}, & \text{if } func = \text{inside}, \\ \sum_{val \in \text{exactVals}(func, index, dist)} val, & \text{if } func = \text{outside}, \end{cases}$$

$$\text{corrApproxIOval}(func_{nt}, dist) := \begin{cases} \frac{\sum_{val \in \text{exactVals}(func_{nt}, dist)} val}{\text{card}(\text{exactVals}(func_{nt}, dist))}, & \text{if } func = \alpha, \\ \sum_{val \in \text{exactVals}(func_{nt}, dist)} val, & \text{if } func = \beta. \end{cases}$$

It should be mentioned that in either case, which means for each considered $func$ with any possible value of $index$ or nt and window size W_e, the transformation of all exact values into corresponding averaged one can be realized in $\mathcal{O}(n^2)$ time – with appropriate implementations of Algorithms 27 and 28.

9.3.3.2. Deriving the Needed (Approximated) Inside and Outside Probabilities

Since for a given sequence r of length n, an item of the form

$$[i, \mathrm{ind}(X \to \gamma\bullet), j+1],$$

$X \in \mathcal{I}_{\mathcal{G}_s}$ and $1 \leqslant i, j \leqslant n$, asserts that

$$X \Rightarrow \gamma \Rightarrow^* r_i \dots r_j,$$

we have (see Section 6.4.1.3):

$$\alpha_X(i,j) := \begin{cases} \sum_{rule \in \mathcal{R}_X} \mathrm{inside}[i, \mathrm{ind}(rule), j+1], & \text{if } (j-i+1) \leqslant W_e, \\ \text{indeterminate}, & \text{else}, \end{cases}$$

and

$$\beta_X(i,j) := \begin{cases} \max_{rule \in \mathcal{R}_X} \mathrm{outside}[i, \mathrm{ind}(rule), j+1], & \text{if } (j-i+1) \geqslant n - W_e, \\ \text{indeterminate}, & \text{else}. \end{cases}$$

Analogously, since an item of the form

$$[\mathtt{dist}, \mathrm{ind}(X \to \gamma\bullet)],$$

$X \in \mathcal{I}_{\mathcal{G}_s}$ and $0 \leqslant \mathtt{dist} \leqslant n$, asserts that

$$X \Rightarrow \gamma \Rightarrow^* w \text{ with } |w| = \mathtt{dist},$$

we have

$$\alpha_X(\mathtt{dist}) := \sum_{rule \in \mathcal{R}_X} \mathrm{inside}[\mathtt{dist}+1, \mathrm{ind}(rule)]$$

and

$$\beta_X(\mathtt{dist}) := \max_{rule \in \mathcal{R}_X} \mathrm{outside}[\mathtt{dist}+1, \mathrm{ind}(rule)].$$

It remains to mention that in order to define all distinct sampling probabilities for the mutually exclusive and exhaustive cases considered by the common sampling strategy described in Section 8.3.2, we additionally need some of the inside probabilities $\alpha_X(i,j)$, $1 \leqslant i, j \leqslant n$ and/or $\alpha_X(\mathtt{dist})$, $1 \leqslant \mathtt{dist} \leqslant n$, for

$$X \in \widehat{\mathcal{I}}_{\mathcal{G}_s} := \{AT, AB, AO, AN\},$$

depending on the considered choice of W_e. These values can easily be calculated from corresponding inside probabilities of actual grammar symbols by using Algorithm 29 and/or Algorithm 30.

Algorithm 29 Computation of additional exact inside probabilities

```
procedure AdditionToINSIDEalg()
if −∞ ⩽ W_e < 0 then
  return
else
  for j = 0 to n do
    for i = j + 1 to 1 do
      if (j − i + 1) ⩽ W_e then
        for nt ∈ Ĵ_{G_s} do
          Compute α_nt(i, j)
        end for
      end if
    end for
  end for
  return
end if
return
end procedure
```

Algorithm 30 Computation of additional approximated inside probabilities

```
procedure AdditionToINSIDEalgAPPROX()
if n ⩽ W_e ⩽ Infinity then
  return
else
  if not −∞ ⩽ W_e ⩽ 0 then
    for nt ∈ Ĵ_{G_s} do
      DeriveAllCorrespondingApproxIOvalues(α_nt)
    end for
  end if
  for dist = 0 to n do
    if dist > W_e then
      for nt ∈ Ĵ_{G_s} do
        Compute α_nt(dist)
      end for
    end if
  end for
  return
end if
return
end procedure
```

It can easily be verified that Algorithm 29 and Algorithm 30 need $\mathcal{O}(n^2 \cdot W_e)$ and $\mathcal{O}(n^2)$ time, respectively, as they rely on the following definitions:

$$\alpha_X(i,j) := \begin{cases} \sum_{l=(i-1)+\min_{ps}}^{(j-1)} \alpha_A(i,l) \cdot \alpha_T(l+1,j), & \text{if } (j-i+1) \leqslant W_e \text{ and } X = AT, \\ \sum_{\substack{l=\max(x,y),\\ x=j-\max_{bulge}-1,\\ y=i+\min_{ps}}}^{j-2} \alpha_A(i+1,l) \cdot \alpha_B(l+1,j-1), & \text{if } (j-i+1) \leqslant W_e \text{ and } X = AB, \\ \sum_{l=i+\min_{ps}}^{(j-1)-\min_{ps}} \alpha_A(i+1,l) \cdot \alpha_O(l+1,j-1), & \text{if } (j-i+1) \leqslant W_e \text{ and } X = AO, \\ \sum_{l=i+\min_{ps}}^{(j-1)} \alpha_A(i+1,l) \cdot \alpha_N(l+1,j-1), & \text{if } (j-i+1) \leqslant W_e \text{ and } X = AN, \\ \text{indeterminate}, & \text{if } (j-i+1) > W_e \text{ and } X \in \widehat{\mathcal{J}}_{\mathcal{G}_s}, \end{cases}$$

$$\alpha_X(\mathtt{dist}) := \begin{cases} \sum_{d=1}^{\mathtt{dist}-\min_{ps}} \alpha_A(\mathtt{dist}-d) \cdot \alpha_T(d), & \text{if } X = AT, \\ \sum_{\substack{x=\max_{bulge},\\ y=\mathtt{dist}-2-\min_{ps}}}^{\min(x,y),} \alpha_A(\mathtt{dist}-2-d) \cdot \alpha_B(d), & \text{if } X = AB, \\ \sum_{d=\min_{ps}}^{\mathtt{dist}-1-\min_{ps}} \alpha_A(d) \cdot \alpha_O(\mathtt{dist}-1-d), & \text{if } X = AO, \\ \sum_{d=\min_{ps}}^{\mathtt{dist}-1} \alpha_A(d) \cdot \alpha_N(\mathtt{dist}-1-d), & \text{if } X = AN. \end{cases}$$

Note that these additional inside probabilities have to be precomputed to ensure that each of the needed sampling probabilities can be derived in constant time during the sampling process.

9.3.4. (Improved) Approximated Sampling Probabilities

Finally, in an attempt to improve the (conditional) probability distributions from which the respective sampling strategy draws the random choices (for unpaired and paired base or substructures), we can make use of the following fact: During the complete sampling process (more precisely, at any point where a random choice has to be drawn from the corresponding distribution), the sampling strategy actually knows both the start position i and the end position j of the currently considered sequence fragment $R_{i,j}$, $1 \leqslant i, j \leqslant n$. However, these were not necessarily known during the derivation of the corresponding approximated inside and outside values for $\mathtt{dist} = j - i + 1$. Consequently, we can in certain cases easily remove some approximate factors in the corresponding approximated inside and outside probabilities and replace them with the respective correct terms (depending on i, j and of course r) in order to obtain more reliable values. This way, it for instance becomes possible to ensure that only such helices are sampled by the respective strategy that consist of at least $\min_{hel}$ valid base pairs.

Therefore, for any employed sampling strategy, the sampling probabilities from which the respective (conditional) distributions for possible choices are inferred should be defined by using the following improved inside and outside probabilities (instead of the corresponding uncorrected precomputed ones):

$$\widehat{\alpha}_X(i,j) := \begin{cases} \alpha_X(i,j), & \text{if } (j-i+1) \leqslant W_e \text{ and } X \in \mathcal{I}_{\mathcal{G}_s} \setminus \{S,L,H,Z\} \\ & \text{ or } (j-i+1) \leqslant W_e \text{ and } X \in \widehat{\mathcal{I}}_{\mathcal{G}_s}, \\ \alpha_X(j-i+1)\cdot 1, & \text{if } (j-i+1) > W_e \text{ and } X \in \{T,G,M,O,N\} \\ & \text{ or } (j-i+1) > W_e \text{ and } X \in \widehat{\mathcal{I}}_{\mathcal{G}_s}, \\ \alpha_X(j-i+1)\cdot \mathrm{corr}^{ub}_{em}(i,j), & \text{if } (j-i+1) > W_e \text{ and } X \in \{C,F,B,U\}, \\ \alpha_X(j-i+1)\cdot \mathrm{corr}^{bp}_{em}(i,j,\mathrm{min}_{hel}), & \text{if } (j-i+1) > W_e \text{ and } X \in \{A\}, \\ \alpha_X(j-i+1)\cdot \mathrm{corr}^{bp}_{em}(i,j,1), & \text{if } (j-i+1) > W_e \text{ and } X \in \{P\}, \\ \text{not needed}, & \text{if } X \in \{S,L,H,Z\}, \end{cases}$$

$$\widehat{\beta}_X(i,j) := \begin{cases} \beta_X(i,j), & \text{if } (j-i+1) \geqslant n - W_e \text{ and} \\ & \qquad X \in \mathcal{I}_{\mathcal{G}_s} \setminus \{S,C,P,F,H,B,U,Z\}, \\ \beta_X(j-i+1)\cdot 1, & \text{if } (j-i+1) < n - W_e \text{ and } X \in \{T,A,O,N\}, \\ \beta_X(j-i+1)\cdot & \\ \quad \mathrm{corr}^{bp}_{em}(i-\mathrm{min}_{hel}, j+\mathrm{min}_{hel}, \mathrm{min}_{hel}), & \text{if } (j-i+1) < n - W_e \text{ and } X \in \{L,G,M\}, \\ \text{not needed}, & \text{if } X \in \{S,C,P,F,H,B,U,Z\} \text{ or } X \in \widehat{\mathcal{I}}_{\mathcal{G}_s}, \end{cases}$$

where

$$\mathrm{corr}^{ub}_{em}(start,end) := \frac{\mathrm{prod}^{ur}_{em}(start,end)}{\left(\mathrm{Pr}^{ub,a}_{em}(1)\right)^{end-start+1}}$$

and

$$\mathrm{corr}^{bp}_{em}(i,j,numBPs) := \frac{\prod\limits_{0 \leqslant k \leqslant numBPs-1} \mathrm{Pr}_{em}(r_{i+k}r_{j-k})}{\prod\limits_{0 \leqslant k \leqslant numBPs-1} \mathrm{Pr}^{bp,a}_{em}((j-k)-(i+k)+1)},$$

with

$$\mathrm{prod}^{ur}_{em}(start,end) := \prod_{start \leqslant k \leqslant end} \mathrm{Pr}_{em}(r_k).$$

Note that the exact emission products $\mathrm{prod}^{ur}_{em}(start,end)$ for unpaired sequence fragments $R_{start,end}$, $1 \leqslant start \leqslant end \leqslant n$, must be precomputed to guarantee that all needed correction terms and hence all relevant inside and outside probabilities $\widehat{\alpha}_X$ and $\widehat{\beta}_X$, $X \in \mathcal{I}_{\mathcal{G}_s} \cup \widehat{\mathcal{I}}_{\mathcal{G}_s}$, can be calculated in constant time. This precomputation step can easily be realized by employing a simple DP routine which derives all of those emission products in $\mathcal{O}(n^2)$ time.

To complete this section, it should be noticed that Algorithm 31 gives a short overview on how to derive all needed inside and outside probabilities that are used to define the distinct sampling probabilities (for mutually exclusive and exhaustive cases, respectively) considered by a particular sampling strategy.

Algorithm 31 Complete inside outside calculations

Input: RNA sequence r of length $n \geqslant 1$,
grammar parameters (trained on a suitable RNA structure database), here given by:
transition probabilities $\Pr_{tr}(\mathit{rule}, len = \mathrm{len}(\mathit{rule}))$ for $\mathit{rule} \in \mathcal{I}_{\mathcal{G}_s}$,
emission probabilities $\Pr_{em}(r_x, len = 1)$ for $r_x \in \Sigma_{\mathcal{G}_r}$ and
emission probabilities $\Pr_{em}(r_{x_1} r_{x_2}, len = x_2 - x_1 + 1)$ for $r_{x_1} r_{x_2} \in \Sigma^2_{\mathcal{G}_r}$.

procedure PreprocessingOfInputSequence()
if $W_e < n$ **then**
/*Counting frequencies in $\mathcal{O}(n^2)$ time (ensures that $\Pr^{ub,a}_{em}(1)$ and $\Pr^{bp,a}_{em}(dist)$ are computed in $\mathcal{O}(1)$ time):*/
Compute all values $rf^{ub}_{em}(u)$, for $u \in \Sigma_{\mathcal{G}_r}$
Compute all values $rf^{bp}_{em}(p_1p_2, dist)$, for valid $p_1p_2 \in \Sigma^2_{\mathcal{G}_r}$ and $0 \leqslant dist \leqslant n$
/*Simple DP in $\mathcal{O}(n^2)$ time (ensures that $corr^{ub}_{em}(start, end)$ is computed in $\mathcal{O}(1)$ time):*/
Compute all values $prod^{ur}_{em}(start, end)$, for $1 \leqslant start \leqslant end \leqslant n$
end if
/*Compute inside outside values of items (needs $\mathcal{O}(n^2)$ time for $W_e < 0$ and $\mathcal{O}(n^3)$ for $W_e \geqslant 0$):*/
INSIDEalg() /*requires $\mathcal{O}(n^2 \cdot W_e) \subseteq \mathcal{O}(n^3)$ time; executed only for $W_e \geqslant 0$*/
INSIDEalgAPPROX() /*requires $\mathcal{O}(n^2)$ time; executed only for $W_e < n$*/
OUTSIDEalg() /*requires $\mathcal{O}(n^3)$ time; executed only for $W_e \geqslant 0$*/
OUTSIDEalgAPPROX() /*requires $\mathcal{O}(n^2)$ time; executed only for $W_e < n$*/
/*Derive corresponding probabilities employed by a particular sampling strategy (needs $\mathcal{O}(n^2)$ time):*/
Compute all conventional inside and outside probabilities α_X and β_X, for $X \in \mathcal{I}_{\mathcal{G}_s}$
Compute all needed $\widehat{\alpha}_X$ and $\widehat{\beta}_X$ values, for $X \in \mathcal{I}_{\mathcal{G}_s}$
/*Compute additional probabilities if needed (requires $\mathcal{O}(n^2)$ time for $W_e < 0$ and $\mathcal{O}(n^3)$ for $W_e \geqslant 0$):*/
AdditionToINSIDEalg() /*requires $\mathcal{O}(n^2 \cdot W_e) \subseteq \mathcal{O}(n^3)$ time; executed only for $W_e \geqslant 0$*/
AdditionToINSIDEalgAPPROX() /*requires $\mathcal{O}(n^2)$ time; executed only for $W_e < n$*/
return
end procedure

9.3.5. Consequences

It should be clear that after an heuristic preprocessing step as described in preceding subsections, the conditional sampling distributions (as considered by a particular strategy) are derived on the basis of the exact grammar parameters (trained beforehand on suitable RNA data) and the resulting functions $\widehat{\alpha}_X$ (and $\widehat{\beta}_X$), $X \in \mathcal{I}_{\mathcal{G}_s} \cup \widehat{\mathcal{I}}_{\mathcal{G}_s}$, for the input sequence. Due to the uncertainties in the corresponding inside and outside probabilities caused by abstraction from the actual input sequence, the employed strategy inevitably has to deal with (more or less) noisy sampling distributions, which can be expected to generally result in a number of unfortunate consequences.

In fact, with increasing level of approximation (that is, with smaller values W_e) and resulting larger uncertainties in the respective sampling distributions, it seems unavoidable that a greater number of – with respect to the exact ensemble distribution – less likely candidate structures are generated before a more reliable one is sampled than without approximations. Hence, it can be expected that an arbitrary random sample set obtained by employing a particular instance of our heuristic preprocessing method typically contains more candidate structures of lower quality and consequently less high quality ones than a corresponding sample produced after exact preprocessing of the input sequence. This is due to the fact that under the assumption that a sufficiently large number of candidate foldings is randomly generated, samples obtained by statistical methods are generally representative with respect to the underlying ensemble distribution. More precisely with respect to the corresponding conditional probability distributions considered by the employed strategy, which in our heuristic cases are noisy.

For this reason, it can not be recommended to employ one of the commonly used (otherwise reasonable) construction schemes for calculating predictions according to the entire sample set. For instance, we should rather neither predict γ_{t-o}-MEA nor γ_{t-o}-centroid structures of the generated sample set as defined in Section 6.5, since those effectively reflect the overall behavior of the sample. However, those predictions must anyway be considered inappropriate choices in connection with our heuristic methods, since their computation requires $\mathcal{O}(n^3)$ time for an input sequence of length n, which would inevitably undo the time reduction reached by approximating the needed inside and outside values (expect for in the case of γ_{t-o}-centroid structures for the default choice $\gamma_{t-o} = 1$, that is unique centroids, as those can be derived in $\mathcal{O}(n^2)$ time). Moreover, note that for similar reasons, one of the usually most straightforward and intuitive choices for extracting structure predictions from statistically representative samples of the overall structure ensemble, the MF structure, might also be inappropriate in connection with approximative variants: in case of inexact preprocessing, the MF structure of a particular statistically representative sample set corresponds to the most likely folding of the given sequence with respect to the noisy distribution induced by approximating, *not* with respect to the exact one.

Nevertheless, we can without significant losses in performance (without increasing the worst-case time complexity of the overall algorithm) identify the MP structure(s) of the generated sample, in strong analogy to traditional SCFG approaches. In fact, for determining the MP structure(s), the probability of each structure can either be computed on the fly while sampling (by multiplying the probabilities of the production rules which correspond to the respective sampling decisions). Or otherwise – since the underlying (L)SCFG $\mathcal{G}_s$ is unambiguous – is computable in $\mathcal{O}(n^2)$ time for a sequence of length n (making use, for instance of an Earley-style parser). Since for this selection principle, we can actually rely on the exact distribution of feasible structures[4], this seems to be the right choice indeed.

[4]Note that the probability for a particular folding of a given RNA sequence is equal to a product of (different powers of the diverse) transition and emission probabilities (according to the corresponding derivation tree). This means it depends only on the exact trained parameter values of the underlying (L)SCFG, see Section 3.3.6.5.

However, note that due to the noisy ensemble distribution resulting from applying our heuristic method to an arbitrary input sequence, the most likely structures according to the exact distribution of the entire structure ensemble for that sequence might be rather unlikely to be sampled. This actually means that lots of candidate structures might be generated by the employed strategy before one of the most likely foldings for the considered input sequence is finally added to the resulting sample set. Hence, when considering rather small sample sizes[5], then extracting the MP structure from a particular sample set might not necessarily result in a reliable prediction in case of noisy distributions caused by heuristic inside-outside calculations. In fact, it seems reasonable to believe that with higher degree of approximation for the preprocessing step (that is, with decreasing values of W_e), the sample size should be increased in oder to guarantee an acceptable accuracy and reproducibility of MP structure predictions. That is, more candidate structures ought to be generated for ensuring that the resulting MP predictions are not only of sufficiently high quality but indeed also reproducible (by independent runs for the same input sequence).

Finally, it should be mentioned that one could alternatively consider a particular subset of the overall sample that contains only those candidate foldings satisfying a preliminary defined quality criterion. For example, only structures with probabilities above a specified threshold or with not less than a specific minimum number of base pairs. This means candidate folding of low quality are disregarded, such that constructing γ_{t-o}-MEA or γ_{t-o}-centroid structures might then result in reliable predictions (where only the derivation of the unique centroid is reasonable with respect to time complexity). Notably, in this context, one should apply a corresponding rejection scheme during sample composition. That is, generated structures are only added to the sample set if they meet the preliminary specified requirements, otherwise they are rejected. This is obviously more efficient than collecting all generated structures and afterwards employing a corresponding filtering process in order to identify the subset to be considered. Utilizing a reasonable rejection criterion, it actually becomes possible to generate any desired number of candidate foldings that obey to the imposed restrictions: new structures are generated until the resulting sample set is large enough. In connection with noisy ensemble distributions, choosing only moderate sizes for such filtered samples might suffice under certain circumstances, which might then result in a reduced runtime compared to the generation of large unfiltered sample sets. Anyway, in the sequel we will exclusively consider unfiltered samples and MP predictions.

9.4. Alternative Sampling Strategy

In order to enable not only meaningful but also comparative evaluations on the decline in sampling quality and predictive accuracy that results from approximative preprocessing, we decided to subsequently consider two contrary sampling methods for the traceback step: First, the simple recursive strategy described in Section 8.3.2 which has principally been originated in [DL03] and hence resembles the one used for the physics-based `Sfold` program. Second, a novel heuristic strategy that basically takes a reverse course of action and is intended to perform better in cases of heuristic preprocessing steps. Accordingly, the aim of this section is to (formally) introduce this alternative sampling variant, as well as to point out the differences between both strategies.

[5] A sample size of only 1000 structures is considered in practice for most applications, as even for a huge set of possible secondary structures of a given sequence, this generally yields statistical reproducibility of typical sampling statistics, even if samples can be entirely different (see [DL03]).

9.4.1. Overview

Recall that the well-established strategy from [DL03] that has been considered so far samples a complete secondary structure $S_{1,n}$ for a given input sequence r of length n in an *outside-to-inside* fashion, since (sub)structures are actually sampled in accordance with corresponding leftmost derivations using grammar $\mathcal{G}_s$. With respect to the goal of this thesis, the reduction of the overall time complexity of statistical (L)SCFG based sampling, it is of great advantage that when employing this particular sampling strategy, the outside values can easily be ignored (see Section 8.3.2).

Unfortunately, this common sampling strategy lacks the ability to take full advantage of the exact inside values $\widehat{\alpha}_X(i,j) = \alpha_X(i,j)$, for $X \in \mathcal{I}_{\mathcal{G}_s}$ and $j - i + 1 \leqslant W_e$, obtained by employing a particular mixed preprocessing variant according to $0 \leqslant W_e < n$. In fact, the strategy in general inevitably has to sample the first base pairs from corresponding conditional probability distributions for rather large fragments $R_{i,j}$ with $j - i + 1 > W_e$, which are indeed induced by approximated sampling probabilities rather than exact ones. Therefore, we decided to design an alternative to this well-established sampling strategy that obeys to contrary principles, resulting in a reverse sampling direction. Our concrete implementation of this strategy depends upon the following axioms:

Ax_1: As soon as possible, sample base pairs and unpaired bases from conditional probability distributions depending on inside values $\widehat{\alpha}_X(i,j)$, $X \in \mathcal{I}_{\mathcal{G}_s}$, for the shortest fragments $R_{i,j}$ – most favorably with $j - i + 1 \leqslant W_e$.

Ax_2: As late as possible, sample base pairs and unpaired bases from conditional probability distributions depending on inside values $\widehat{\alpha}_X(i,j)$, $X \in \mathcal{I}_{\mathcal{G}_s}$, for longer fragments $R_{i,j}$ – which do not satisfy $j - i + 1 \leqslant W_e$.

Note that axiom Ax_1 implies that the generation of (sub)structures starts with sampling of (moderate-sized) single-stranded regions – where possible hairpin loops. In fact, on any considered sequence fragment $R_{start,end}$, $1 \leqslant start \leqslant end \leqslant n$, one of the possible hairpin loops $R_{i,j}$, $start < i \leqslant j < end$ is drawn according to the induced sampling distribution, as $j - i + 1 \leqslant W_e$ can be satisfied by an appropriate choice of W_e (details will follow). Notably, the considered sampling distribution does not include a probability for leaving the whole fragment $R_{start,end}$ single-stranded. This is due to the fact that for long fragments $R_{start,end}$, the condition $end - start + 1 \leqslant W_e$ does usually not hold. Thus, $R_{start,end}$ will only be left unpaired if no hairpin loop can be formed on this fragment. As a consequence, some of the feasible foldings for the considered input sequence r might never be generated; these include some of the secondary structures with (rather long) unpaired regions on which an additional hairpin loop could still be formed[6]. This means that the strategy is in general only capable of producing a subset of the complete ensemble for feasible structures for a given input sequence. Therefore, it inherently samples candidate foldings according to a skewed ensemble distribution – even in connection with exact preprocessing steps. Applying this strategy in the traceback step hence in any case results in an overall heuristic rather than in a proper statistical sampling algorithm.

The implication of axiom Ax_2 is that the generation of (sub)structures proceeds with folding of more complex structural elements (that must contain at least one hairpin loop). This means that (sub)structures are not sampled in accordance with the underlying SCFG, but in an *inside-to-outside* fashion.

Accordingly, under the assumption of both axioms, a complete secondary structure $S_{1,n}$ for a given input sequence r of length n can be sampled in the following (deliberately less

[6]Specifically, (long) fragments at the ends of multiloops or the exterior loop can never remain unpaired if further hairpin loops are possible, which is basically due to the left-to-right generation of adjacent substructures within these complex loop types according to the underlying grammar $\mathcal{G}_s$ (details will follow).

controlled and hence less restrictive) way: Start with the entire RNA sequence $R_{start,end} = R_{1,n}$ and randomly construct adjacent substructures (paired substructures preceded by potentially empty single-stranded regions) of the exterior loop on the considered sequence fragment $R_{start,end}$, as long as no further paired substructure can be folded; the construction does *not* follow a particular order, for example from left to right. Any (paired) substructure on fragment $R_{start,end}$, $1 \leqslant start \leqslant end \leqslant n$, is created by sampling a random hairpin loop (with closing base pair i.j, for $start < i < j < end$) and extending it (towards the ends of $R_{start,end}$) by successively drawing closing base pairs. During this extension, basically all known substructures (stacked pairs, bulges, interior and multi-branched loops, that obey to certain restrictions which will be discussed later) may be folded, where each substructure (for instance a multiloop) has to be completed before its closing base pair is added and the corresponding helix can actually be further extended. The process of folding a particular paired substructure ends with a complete and valid paired structure (of the currently folding multiloop or of the exterior loop), either with or without a corresponding preceding unpaired region, both on the considered fragment $R_{start,end}$.

Figure 9.1 gives a schematic overview of this inside-out fashion sampling strategy. Much more detailed information on the distinct sampling steps is provided in Section 9.4.2 in the form of corresponding pseudocode procedures and formal definitions. A complete exemplary decision tree for a simple input sequence is presented in Section 9.4.3. Furthermore, by the end of Section 9.4.3, we also discuss a particular rather complex problem that inherently arises in connection with inside-to-outside sampling strategies like ours, specifically that the consideration of incorrect sampling probabilities can not be avoided under the assumption of this sampling direction. However, before getting in that much detail, we first want to discuss some of the more obvious aspects and related problems.

First, note that due to the *arbitrary* sampling order and the implied greater variety during the sampling process, there are usually different ways for sampling the same candidate structure, which unfortunately causes severe problems that will be discussed later on. In fact, any secondary structure might be sampled in more than one way with this alternative sampling strategy, whereas with the common strategy, there is always only one unique way for sampling a particular structure, namely in accordance with the generation of the corresponding unique leftmost derivation using the underlying SCFG (as shown in Example 3.2.1). It should be mentioned that rather than sampling adjacent substructures in arbitrary order, our strategy could alternatively be designed to generate substructures form left to right (or else, from right to left). However, forcing a fixed order would imply even more severe problems (see Remark 9.4.1).

Example 9.4.1 *We want to demonstrate some of the different ways for recursively sampling the (correct) secondary structure of* E.coli $tRNA^{Ala}$ *when using our novel strategy. For proper illustration of the diverse stages of the sampling process, we use an extended variant of traditional dot-bracket representation for the partially formed structures, similar to Example 3.2.1. In particular, positions that have not yet been solved (that is, determined to be unpaired or paired) are represented by symbols* *. *Additionally, positions that are predetermined to be paired and will inevitably base pair in the next sampling step(s) are represented by pairs of corresponding square brackets* [ ]. *Finally, any two positions that at a particular stage mark the ends of a currently folding multiloop (no matter if one of them or both are solved or not) are represented by a pair of corresponding curly brackets* { }. *The sampling of the correct* E.coli $tRNA^{Ala}$ *structure might then be realized as follows:*

- *Consider the overall sequence* $R_{1,n=76}$
  $\rightsquigarrow$ **********
- *Randomly draw hairpin loop on* $R_{1,76}$
  $\rightsquigarrow$ **********[[∘∘∘∘∘∘∘∘]]**********
- *Extend helix and randomly choose* $S_{8,65}$ *to imply the first substructure of a multiloop*
  $\rightsquigarrow$ *******{∘((((∘∘∘∘∘∘∘∘))))**********}**********

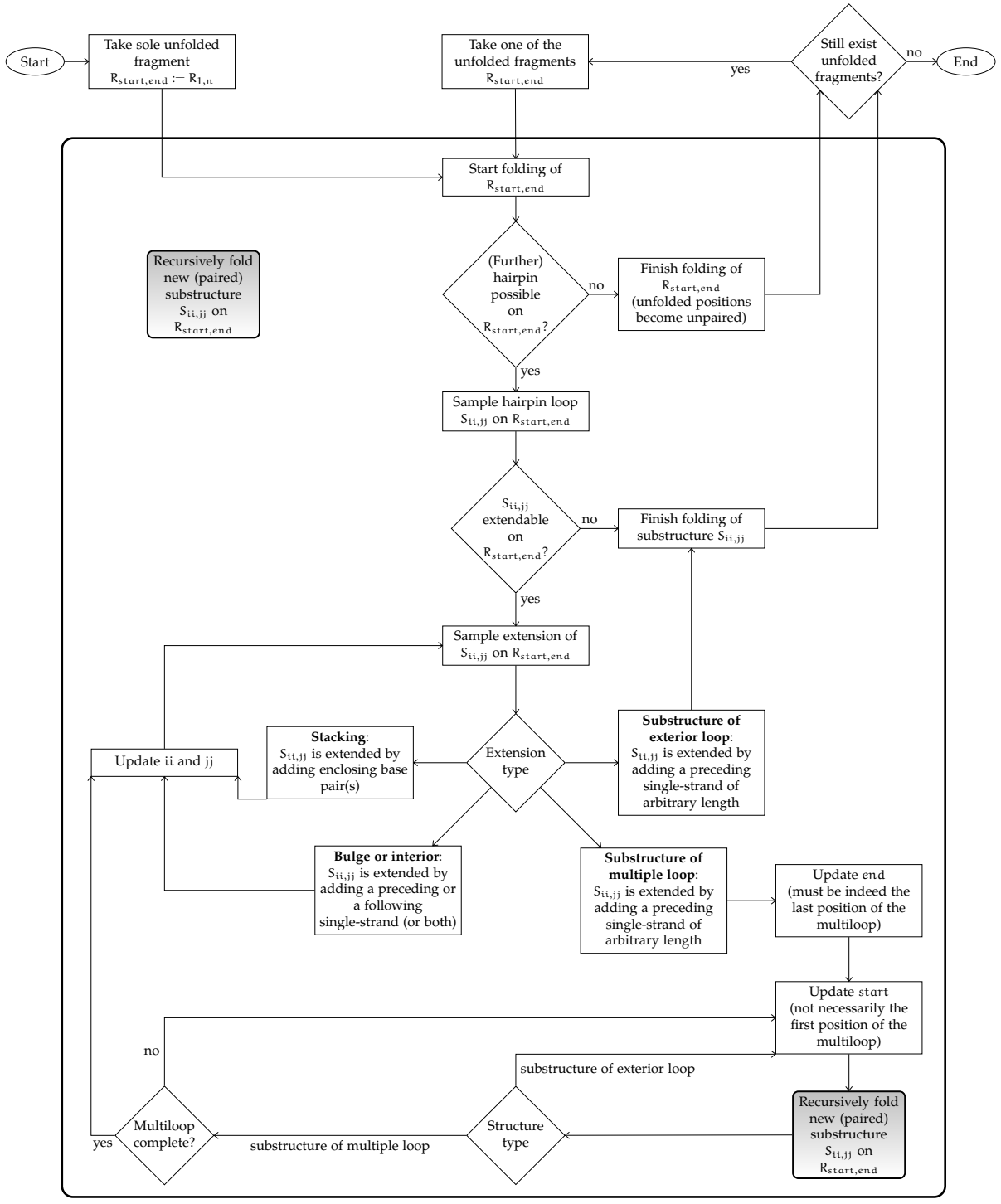


Figure 9.1.: **Flowchart for recursive sampling according to our novel strategy.** The flowchart describes the sampling of a complete secondary structure $s = S_{1,n}$ for a given input sequence $r = R_{1,n}$ of length n according to a less restrictive strategy with extensively more freedom than the common one (which therefore requires dynamic validation of possible random choices during the sampling process).

- *Randomly draw hairpin loop on* $R_{3,65}$
  $\rightsquigarrow$ *******{∘((((∘∘∘∘∘∘∘∘))))****[[∘∘∘∘∘∘∘]]**********************}**********
- *Extend helix and randomly choose* $S_{8,65}$ *to imply the first substructure of a multiloop*
  $\rightsquigarrow$ *******{∘((((∘∘∘∘∘∘∘∘))))∘(((((∘∘∘∘∘∘∘)))))********************}**********
- *Randomly draw hairpin loop on* $R_{8,65}$
  $\rightsquigarrow$ *******{∘((((∘∘∘∘∘∘∘∘))))∘(((((∘∘∘∘∘∘∘)))))********[[∘∘∘∘∘∘∘]]**}**********
- *Extend helix and complete multiloop*
  $\rightsquigarrow$ *******∘∘((((∘∘∘∘∘∘∘∘))))∘(((((∘∘∘∘∘∘∘)))))∘∘∘∘∘(((((∘∘∘∘∘∘∘)))))**********
- *Finish ongoing construction of multiloop*
  $\rightsquigarrow$ *****[[∘∘((((∘∘∘∘∘∘∘∘))))∘(((((∘∘∘∘∘∘∘)))))∘∘∘∘∘(((((∘∘∘∘∘∘∘)))))]]*********
- *Extend helix and randomly choose it to become a paired substructure of the exterior loop*
  $\rightsquigarrow$ (((((((∘∘((((∘∘∘∘∘∘∘∘))))∘(((((∘∘∘∘∘∘∘)))))∘∘∘∘∘(((((∘∘∘∘∘∘∘)))))))))))****
- *Find that it is not possible to form a hairpin loop on* $R_{73,76}$
  $\rightsquigarrow$ (((((((∘∘((((∘∘∘∘∘∘∘∘))))∘(((((∘∘∘∘∘∘∘)))))∘∘∘∘∘(((((∘∘∘∘∘∘∘)))))))))))∘∘∘∘
- *Finish folding (no unsolved regions)!*

Alternatively, this secondary structure might be sampled in the following way:

- *Consider the overall sequence* $R_{1,n=76}$
  $\rightsquigarrow$ **
- *Randomly draw hairpin loop on* $R_{1,76}$
  $\rightsquigarrow$ **************************[[∘∘∘∘∘∘∘]]*********************************
- *Extend helix and randomly choose* $S_{26,65}$ *to imply the second substructure of a multiloop*
  $\rightsquigarrow$ ************************{(((((∘∘∘∘∘∘∘)))))********************}**********
- *Randomly draw hairpin loop on* $R_{3,65}$
  $\rightsquigarrow$ **********[[∘∘∘∘∘∘∘∘]]**{(((((∘∘∘∘∘∘∘)))))********************}**********
- *Extend helix and randomly choose* $S_{8,65}$ *to imply the first substructure of a multiloop*
  $\rightsquigarrow$ *******{∘((((∘∘∘∘∘∘∘∘))))∘(((((∘∘∘∘∘∘∘)))))********************}**********
- *Randomly draw hairpin loop on* $R_{8,65}$
  $\rightsquigarrow$ *******{∘((((∘∘∘∘∘∘∘∘))))∘(((((∘∘∘∘∘∘∘)))))********[[∘∘∘∘∘∘∘]]**}**********
- *Extend helix and complete multiloop*
  $\rightsquigarrow$ *******∘∘((((∘∘∘∘∘∘∘∘))))∘(((((∘∘∘∘∘∘∘)))))∘∘∘∘∘(((((∘∘∘∘∘∘∘)))))**********
- *Finish ongoing construction of multiloop*
  $\rightsquigarrow$ *****[[∘∘((((∘∘∘∘∘∘∘∘))))∘(((((∘∘∘∘∘∘∘)))))∘∘∘∘∘(((((∘∘∘∘∘∘∘)))))]]*********
- *Extend helix and randomly choose it to become a paired substructure of the exterior loop*
  $\rightsquigarrow$ (((((((∘∘((((∘∘∘∘∘∘∘∘))))∘(((((∘∘∘∘∘∘∘)))))∘∘∘∘∘(((((∘∘∘∘∘∘∘)))))))))))****
- *Find that it is not possible to form a hairpin loop on* $R_{73,76}$
  $\rightsquigarrow$ (((((((∘∘((((∘∘∘∘∘∘∘∘))))∘(((((∘∘∘∘∘∘∘)))))∘∘∘∘∘(((((∘∘∘∘∘∘∘)))))))))))∘∘∘∘
- *Finish folding (no unsolved regions)!*

However, another way for sampling this structure might look as follows:

- *Consider the overall sequence* $R_{1,n=76}$
  $\rightsquigarrow$ **
- *Randomly draw hairpin loop on* $R_{1,76}$
  $\rightsquigarrow$ **[[∘∘∘∘∘∘∘]]**************
- *Extend helix and randomly choose* $S_{44,65}$ *to imply the* k^{th} *substructure of a multiloop,* $k > 2$
  $\rightsquigarrow$ **{∘∘∘∘(((((∘∘∘∘∘∘∘)))))}**********
- *Randomly draw hairpin loop on* $R_{3,65}$
  $\rightsquigarrow$ **********[[∘∘∘∘∘∘∘∘]]******************{∘∘∘∘(((((∘∘∘∘∘∘∘)))))}**********
- *Extend helix and randomly choose* $S_{8,65}$ *to imply the first substructure of a multiloop*
  $\rightsquigarrow$ *******{∘((((∘∘∘∘∘∘∘∘))))****************∘∘∘∘∘(((((∘∘∘∘∘∘∘)))))}**********
- *Randomly draw hairpin loop on* $R_{8,65}$
  $\rightsquigarrow$ *******{∘((((∘∘∘∘∘∘∘∘))))****[[∘∘∘∘∘∘∘]]***∘∘∘∘∘(((((∘∘∘∘∘∘∘)))))}**********
- *Extend helix and complete multiloop*
  $\rightsquigarrow$ *******∘∘((((∘∘∘∘∘∘∘∘))))∘(((((∘∘∘∘∘∘∘)))))∘∘∘∘∘(((((∘∘∘∘∘∘∘)))))**********

- *Finish ongoing construction of multiloop*
  $\rightsquigarrow$ *****[[∘∘((((∘∘∘∘∘∘∘∘))))∘(((((∘∘∘∘∘∘∘))))∘∘∘∘∘(((((∘∘∘∘∘∘∘)))))]]*********
- *Extend helix and randomly choose it to become a paired substructure of the exterior loop*
  $\rightsquigarrow$ (((((((∘∘((((∘∘∘∘∘∘∘∘))))∘(((((∘∘∘∘∘∘∘))))∘∘∘∘∘(((((∘∘∘∘∘∘∘)))))))))))****
- *Find that it is not possible to form a hairpin loop on* $R_{73,76}$
  $\rightsquigarrow$ (((((((∘∘((((∘∘∘∘∘∘∘∘))))∘(((((∘∘∘∘∘∘∘))))∘∘∘∘∘(((((∘∘∘∘∘∘∘)))))))))))∘∘∘∘
- *Finish folding (no unsolved regions)!*

It is easy to see that there are even more ways for sampling this particular secondary structure, but due to its simple composition (matching the typical cloverleaf structure of tRNAs), those are rather few and any of them actually quite similar to the presented ones.

It should be mentioned that in any step of the sampling process, one feasible substructure is randomly drawn from a corresponding probability distribution induced by all (still) possible substructures. In contrast to the common sampling algorithm employed in the preceding chapters, this distribution is not conditioned on the considered sequence fragment, but it strongly depends on the previously folded unpaired regions and helices. By successively folding additional substructures, the number of still possible alternatives to be considered in subsequent sampling steps more and more decreases, which can easily be understood when considering Example 9.4.1.

Essentially, during a particular sampling process, there are usually at several points many alternatives to be drawn that might yield to the same overall candidate structure (with corresponding probabilities that sum up), rather than only one unique choice that needs to be drawn in order to obtain that folding (with a corresponding greater probability). Thus, the strategy does not sample candidate foldings along conditional distributions over the feasible (sub)structures, but along conditional distributions over the alternative generation or decision (sub)paths corresponding to feasible (sub)structures. Hence, if a particular folding corresponds to more than one path, then any of them might be used by the strategy with the respective probability, where paths starting with choices of the most likely alternatives will be favored.

Since intuitively, at any stage of the sampling process, the probability for a particular final candidate structure equates to the sum of the probabilities of its different generation paths, we suppose that this structure is sampled with the same probability as it would have been in the case of a corresponding unique decision path – which under the assumption of proper conditional sampling distributions should be equal to the structure probability according to underlying stochastic model, such that the arbitrary sampling order does not imply a change in the distribution of the generated sample (compared to the ensemble distribution). However, this is only an intuitive suggestion and has not been proven, as the provision of a clear evidence (or an appropriate counterexample) is not trivial. Nevertheless, since the conditional sampling distributions are generally *not* proper (details will follow), the distribution of generated samples anyhow differs from the distribution induced by the applied structure model.

From the mere algorithmic point of view, a crucial aspect is that in order to ensure that all sampled substructures can be successfully folded, especially in the case of multiloops, we have to take care that at any point, the strategy may only draw such random choices that do not make it impossible to successfully finish the currently running construction process (of a particular loop). In fact, when randomly sampling the next extension of an already folded (paired) structure $S_{i,j}$ on the considered sequence fragment $R_{start,end}$, we have to make sure that only *valid* choices can actually be drawn – from a corresponding probability distribution derived for exactly all those valid choices. Obviously, the set of all valid choices for a particular extension step is not only restricted by the start and end positions of the considered fragment $R_{start,end}$ and the first and last positions of the substructure $S_{i,j}$ currently being constructed, but in general also strongly depends on the actual positions and types of all previously folded paired substructures (composed of already selected unpaired and paired bases). Therefore,

before any random choices can be drawn, the algorithm needs to *dynamically* determine the respective sets of all possible choices, during the sampling process itself.

This, unfortunately, may cause severe problems as regards the time complexity for randomly sampling the next extension (or base pair). Principally, the dynamic evaluations of possible choices during the sampling process naturally results in more computation effort for drawing random choices compared to a more controlled sampling strategy that works strictly according to a certain construction scheme (for example by folding substructures from left to right) and thus always considers basically the same choices for folding a particular sequence fragment. The additional computation effort implied by the described strategy can thus be seen as the inevitable price to be paid for gaining more freedom and reaching a greater amount of randomization during the sampling process. However, in order to guarantee that the worst-case time complexity for drawing any random choice remains in $\mathcal{O}(n)$ for n the length of the input sequence, we only need to impose a few restrictions concerning the lengths of single-stranded regions in some specific types of loops. In detail, we have to consider constant maximum allowed numbers of nucleotides in unpaired regions of hairpin loops ($\max_{\mathtt{hairpin}}$), bulge or interior loops ($\max_{\mathtt{bulge}}$), and multiloops ($\max_{\mathtt{strand}}$)[7].

For example, if $X \in \mathcal{I}_{\mathcal{G}_s}$ generates hairpin loops, then the set of all possible hairpin loops that can be validly folded on sequence fragment $R_{\mathtt{start},\mathtt{end}}$ is given by

$$\begin{aligned}\mathrm{pcHL}(\mathtt{start},\mathtt{end}) := \big\{(i,j,p) \mid\ & \mathtt{start} + \min_{\mathrm{hel}} \leqslant i \leqslant j \leqslant \mathtt{end} - \min_{\mathrm{hel}} \text{ and}\\ & i + \min_{\mathrm{HL}} - 1 \leqslant j \leqslant i + \max_{\mathtt{hairpin}} - 1 \text{ and}\\ & R_{i-\min_{\mathrm{hel}},j+\min_{\mathrm{hel}}} \text{ not folded and}\\ & p = \widehat{\beta}_X(i,j) \cdot \widehat{\alpha}_X(i,j) \neq 0\big\}.\end{aligned}$$

Obviously, $\max_{\mathtt{hairpin}}$ indeed ensures that $\mathrm{pcHL}(\mathtt{start},\mathtt{end})$ can be computed in $\mathcal{O}(n)$ time. Note that formal definitions of all corresponding sets of possible substructures that are considered by this dynamic sampling strategy, including formulæ for deriving the respective sampling probabilities, can be found in Section 9.4.2.

However, it needs to be mentioned that reasonable values for the three restrictive parameters $\max_{\mathtt{hairpin}}$, $\max_{\mathtt{bulge}}$ and $\max_{\mathtt{strand}}$ can readily be obtained along with grammar parameter estimation for any particular database. In fact, if length-dependent training procedures are applied, then suitable values can easily be read off the corresponding sets of estimated rule probabilities (for the distinct length intervals). Notably, under the assumption of the length-dependent model these restrictions are in general anyway implicitly used: the respective transition probabilities for generating longer strands are then all equal to zero, such that those substructures are in either case prohibited. When considering the plain SCFG model, however, the probabilities for longer strands are generally not equal to zero, but actually very small, such that those substructures are at least rather unlikely. Hence, in connection with our (L)SCFG based sampling method, the consideration of such RNA type specific choices for all three restrictions (readily obtained along with training) poses an adequate alternative to the commonly considered constant value of 30, respectively. For this reason, we will often use the following inherently constant choices:

$$\max_{\mathtt{hairpin}} = \min(\max^{*}_{\mathtt{hairpin}}, 30),$$

[7] Note that these restrictions are not as severe as it may seem, since for example choosing the constant value 30 for all three parameters can be expected to hardly have a negative impact on the resulting sampling quality, since longer strands are rather unrealistic and should not be contained that often in native RNA secondary structures (and trusted structural databases). Moreover, it seems worth noticing that due to the rare occurrences of unpaired regions of lengths > 30 in nature, the thermodynamic parameters for such long single-strands have not been determined by exact measurements from real-world data but by extrapolations, and many MFE based prediction algorithms also restrict the lengths of particular single-stranded regions, at least for long bulge and interior loops (where the proposed constant value of 30 is considered a common choice), see for instance [DL03].

$$\max_{bulge} = \min(\max^*_{bulge}, 30),$$
$$\max_{strand} = \min(\max^*_{strand}, 30),$$

where

$$\begin{aligned}
\max^*_{hairpin} &:= \text{ maximum length } l \geqslant \min_{HL} \text{ for which } \Pr(F \to Z^{\min_{HL}-1}H, l) \neq 0,\\
\max^*_{bulge} &:= \text{ maximum length } l \geqslant 1 \text{ for which } \Pr(B \to ZB, l) \neq 0,\\
\max^*_{strand} &:= \text{ maximum length } l \geqslant 1 \text{ for which } \Pr(U \to ZU, l) \neq 0.
\end{aligned}$$

Finally, it should be noted that the sampling strategy introduced within this section needs to additionally consider outside probabilities, for two reasons:

- First, the outside values need to be included for obtaining well-defined sampling distributions – in accordance with the underlying stochastic model. They indeed (strongly) influence the respective sampling distributions, since they are not common factors of all sampling probabilities for the relevant possible choices (mutually exclusive and exhaustive cases), respectively. Notably, the different possible choices (i, j, p) usually imply substructures $S_{i,j}$ of different lengths $j - i + 1$, such that only $p = \widehat{\alpha}_X(i, j) \cdot \widehat{\beta}_X(i, j)$ ensures that the probabilities of all possible choices are of the same order of magnitude and can hence yield a reasonable probability distribution for drawing a random choice.

- Second, the outside values are required for guaranteeing that sampled substructures can be validly extended. This means that only such hairpin loops and extensions implying a surrounding base pair $i.j$ may be sampled that can actually lead to the generation of a corresponding valid helix, consisting of at least $\min_{hel}$ valid (for example canonical) pairs $i.j, \ldots, (i - \min_{hel} + 1).(j + \min_{hel} - 1)$, with the last one $i.j$ closing the sampled substructure.

Hence, we can conclude that when employing the conventional sampling strategy from Section 8.3.2, the overall worst-case time complexity for generating a statistical sample can be reduced by considering a constant window size W_e for exact inside calculations. However, if samples sets are generated by our dynamic strategy, then only the strongest heuristic variant which approximates all of the considered inside and outside values (and performs no exact calculations at all) is capable of reducing the overall time requirements in a significant way.

Aspect	Conventional Strategy	Dynamic Strategy
Preprocessing input	1) RNA sequence r of length n 2) Structural grammar parameters: $\min_{HL} \geqslant 1$ and $\min_{hel} \geqslant 1$ 3) Probabilistic grammar parameters (trained on suitable RNA database): - transition probabilities $\Pr_{tr}(\mathit{rule})$ for $\mathit{rule} \in \mathcal{R}_{\mathcal{G}_s}$, - emission probabilities $\Pr_{em}(x)$ for $x \in \Sigma_{\mathcal{G}_r}$ and $\Pr_{em}(x_1x_2)$ for $x_1x_2 \in \Sigma^2_{\mathcal{G}_r}$	
Preprocessing results	Inside probabilities $\widehat{\alpha}_X(i, j)$, for $X \in \mathcal{I}_{\mathcal{G}_s} \cup \widehat{\mathcal{I}}_{\mathcal{G}_s}$ and $1 \leqslant i, j \leqslant n$	Inside probabilities $\widehat{\alpha}_X(i, j)$ and outside probabilities $\widehat{\beta}_X(i, j)$, for $X \in \mathcal{I}_{\mathcal{G}_s}$ and $1 \leqslant i, j \leqslant n$, as well as $(3 \times n \times n)$ matrix firstPossMLstarts and $(n \times n)$ matrix furtherPSpossible
Preprocessing time	$\mathcal{O}(n^3)$ for exact calculations $\mathcal{O}(n^2)$ for approximate variant $\mathcal{O}(n^2)$ with constant $W_e \geqslant 0$	$\mathcal{O}(n^3)$ for exact calculations $\mathcal{O}(n^2)$ for approximate variant $\mathcal{O}(n^3)$ with constant $W_e \geqslant 0$

(a) Preprocessing.

Aspect	Conventional Strategy	Dynamic Strategy
Sampling parameters	$max_{bulge} \geqslant 1$: - can be arbitrarily chosen - no restrictions are imposed	$max_{hairpin} \geqslant min_{HL}$, $max_{bulge} \geqslant 1$ and $max_{strand} \geqslant 0$: must be chosen constant to ensure that the worst-case time complexity does not increase
Constraint	min_{hel} supposed to be constant (to avoid increase of worst-case time complexity)	
Sampling properties and course of action	Inherently controlled, ordered: - substructures from left to right, - sampling proceeds "inwards": construction of substructure $S_{i,j}$ starts by considering $R_{i,j}$ and ends by generating an unpaired region (usually a hairpin loop)	Extensively more freedom, less restrictive: - substructures sampled in arbitrary order, - sampling proceeds "outwards": construction of new substructure on an unfolded fragment $R_{start,end}$ starts by choosing a random hairpin loop and continues by extending it to a completely folded and valid (paired) substructure $S_{i,j}$ on $R_{start,end}$
Benefits of sampling direction	(Sub)structures are folded in accordance with underlying SCFG	Outward direction might perform better with approximated sampling probabilities
Problems caused by sampling direction	None; proper statistical sampling from entire structure ensemble	Only sampling heuristic: 1) Different ways for sampling the same folding, but some foldings for a given sequence can never be sampled 2) Use of incorrect sampling probabilities in some cases unavoidable
Function of outside values	Do not influence resulting sampling probabilities (are common factor); are therefore not considered (in order to save time)	1) Needed for ensuring well-defined sampling distributions (are not common factor) 2) Required for guaranteeing that sampled substructures can be validly extended
Identification of valid choices	Not required: all possible choices on the considered unfolded fragment are principally valid	Dynamically checked (during sampling): - dependent on existing substructures - firstPossMLstarts and furtherPSpossible are heavily used (in case of multi- and exterior loops)
Folding time	$\mathcal{O}(n^2)$	$\mathcal{O}(n^2)$ with larger constants, due to dynamic evaluations

(b) Stochastic traceback.

Aspect	Conventional Strategy	Dynamic Strategy
Overall time complexity	$\mathcal{O}(n^3)$ for all schemes with exact variant or non-constant $W_e \geqslant 0$, $\mathcal{O}(n^2)$ for efficient schemes with constant $W_e \geqslant 0$ or complete approximation	$\mathcal{O}(n^3)$ for all schemes (MP, MF, γ-MEA and γ-centroid) in case of exact computations or mixed variants according to $W_e \geqslant 0$, $\mathcal{O}(n^2)$ for efficient schemes (MP, MF and unique centroid) only with complete approximation
Accuracy	With higher degree of approximation (decreasing values of $-1 \leqslant W_e \leqslant n$): - MF, γ-MEA and γ-centroid become less reliable (since they reflect the noisy ensemble distribution caused by heuristic preprocessing), - MP predictions gain in relevance (since those can be efficiently derived from noisy statistical samples with respect to the exact ensemble distribution implied by the underlying SCFG), - Sample size should be increased in oder to guarantee reproducibility of MP structure predictions	

(c) Extension to structure prediction.

Table 9.1.: **Differences between both sampling strategies.** Table compares the conventional sampling strategy described in Section 8.3.2 (that basically works in the same way as the sampling algorithm proposed in [DL03]) and the devised dynamic sampling strategy, according to a number of relevant aspects (concerning the recursive sampling of a random RNA secondary structure $S_{1,n}$ for a given input sequence r of length n).

Nevertheless, for $W_e \geqslant 0$, the outward sampling direction of our dynamic strategy might perform better with approximated sampling probabilities. In fact, consecutively extending a (usually) short randomly sampled hairpin loop outwards to a longer and longer (paired) substructures implies considering inside probabilities for shorter fragments – which inherently contain less approximated terms and thus less inaccuracies – first. As a consequence, the correct positions of smaller substructures, especially initial hairpin loops, might be identified more precisely. Anyway, this potential is to some extend narrowed by the corresponding outside values for which the contrary holds.

We complete this section by referring to Table 9.1 that summarizes the main differences of both sampling variants.

9.4.2. Formal Description of the Sampling Process

The intention of this section is to present all details and formal definitions connected to the dynamic sampling strategy sketched in Section 9.4.1.

Before we start, recall that at any stage of the sampling process, the strategy may only draw such random choices that do not make it impossible to successfully finish the currently running construction (of a particular loop) such that any sampled substructure can be readily completed in accordance with the previously folded substructures (or parts of substructures in case of complex multiloops). Furthermore, we already mentioned that due to the required dynamic evaluation procedures for identifying the respective valid choices, the strategy must be restricted to consider a constant maximum allowed number of nucleotides in unpaired regions of hairpin loops ($max_{hairpin}$), bulge or interior loops (max_{bulge}), and multiloops (max_{strand}), respectively, in order to ensure a $\mathcal{O}(n)$ time complexity for drawing a particular random choice.

However, for guaranteeing that the valid choices, mainly in the case of (an ongoing construction of) a particular multiloop substructure, can be identified and the corresponding extension can be randomly drawn in $\mathcal{O}(n)$ time, we need to additionally derive two different matrices in a corresponding extended preprocessing step:

First, a $(3 \times n \times n)$ matrix named firstPossMLstarts, which contains information about the smallest position on a considered sequence fragment $R_{minAllowedStartPos,endPos}$ where an arbitrary k^{th} multiloop substructure (paired substructure preceded by a potentially empty strand) can start if the corresponding multiloop ends at position $endPos$, formally

$$\text{firstPossMLstarts}[k, minAllowedStartPos, endPos] := \begin{cases} \text{firstMLstart}_{k,M}(minAllowedStartPos, endPos), & \text{if } k = 1, \\ \text{firstMLstart}_{k,O}(minAllowedStartPos, endPos), & \text{if } k = 2, \\ \text{firstMLstart}_{k,N}(minAllowedStartPos, endPos), & \text{if } k \geqslant 3, \end{cases}$$

where

$$\begin{aligned} &\text{firstMLstart}_{k,X}(minAllowedStartPos, endPos) := \\ &\min(\{startPos \mid minAllowedStartPos \leqslant startPos \leqslant endPos - (2 - (k-1)) \cdot \min_{ps} + 1 \text{ and} \\ &\qquad \left(\widehat{\beta}_X(startPos, endPos) \cdot \widehat{\alpha}_X(startPos, endPos)\right) \neq 0\}). \end{aligned}$$

The second one is a $(n \times n)$ matrix, which will be denoted by furtherPSpossible. For each pair of $startPos$ and $endPos$ (first and last position of a considered sequence fragment $R_{startPos,endPos}$), the corresponding value answers the question if a further paired substructure (consisting of at least $\min_{hel}$ consecutive base pairs with the last pair closing a hairpin loop) could actually be folded on $R_{startPos,endPos}$, formally

$$\begin{aligned} &\text{furtherPSpossible}[startPos, endPos] := \\ &\qquad \exists start, end \text{ with } startPos \leqslant start \leqslant endPos \text{ and} \\ &\qquad\qquad start + \min_{ps} - 1 \leqslant end \leqslant endPos \text{ and} \\ &\qquad\qquad \left(\widehat{\beta}_A(start, end) \cdot \widehat{\alpha}_A(start, end)\right) \neq 0. \end{aligned}$$

Both matrices can be calculated by employing simple DP routines, respectively, with $\mathcal{O}(n^2)$ time and space requirements for a given input sequence r of length n. Note that these precalculated matrices are indeed (heavily) used in Algorithms 32 to 40, which formally describe how the sampling strategy constructs a random secondary structure for a given input sequence and how it ensures that all chosen substructures are valid and complete[8]. Notably, a bunch of definitions of sets needs to be considered, which will subsequently be presented in detail.

[8]Note that in the presented pseudocode procedures, $helices[-1]$ represents the unique element that has been added most recently to the (non-empty) set $helices$.

Algorithm 32 Sampling an entire secondary structure (alternative strategy)

Input: RNA sequence r of length $n \geqslant 1$,
trained transition probabilities $\mathrm{Pr}_{tr}(\mathit{rule})$, for $\mathit{rule} \in \mathcal{R}_{\mathcal{G}_s}$,
precomputed inside and outside values $\widehat{\alpha}_X(i,j)$ and $\widehat{\beta}_X(i,j)$, for $X \in \mathcal{I}_{\mathcal{G}_s}$ and $1 \leqslant i,j \leqslant n$,
precomputed matrices firstPossMLstarts and furtherPSpossible.

Output: $\mathtt{helices} = \{(i,j,k) \mid 1 \leqslant i < j \leqslant n$ and $k \geqslant \min_{\mathrm{hel}}$ and
$i.j, (i+1).(j-1), \ldots, (i+(k-1)).(j-(k-1))$ are consecutive pairs$\}$,
$\mathtt{unpRegs} = \{(i,l) \mid 1 \leqslant i \leqslant n$ and $l \geqslant 1$ and
$i, (i+1), \ldots, (i+(l-1))$ are subsequent unpaired bases$\}$.

procedure RandomlyFoldOverallStructure()
$\mathtt{helices} = \emptyset$, $\mathtt{unpRegs} = \emptyset$
$\mathtt{unfoldedFragments} = \{(1,n)\}$
while $\mathtt{unfoldedFragments} \neq \emptyset$ **do**
$\mathtt{fragment} = \mathtt{unfoldedFragments}[1]$
$\mathtt{unfoldedFragments} = \mathtt{unfoldedFragments} \setminus \{\mathtt{fragment}\}$
$\mathtt{res} =$ RandomlyFoldPairedSubstructure($\mathtt{fragment}[1], \mathtt{fragment}[2], \mathtt{helices}, \mathtt{unpRegs}$)
$\mathtt{helices} = \mathtt{res}[2]$, $\mathtt{unpRegs} = \mathtt{res}[3]$
$\mathtt{unfoldedFragments} =$ remainingFragments($1, n, \mathtt{helices}, \mathtt{unpRegs}$)
end while
return $(\mathtt{helices}, \mathtt{unpRegs})$
end procedure

Algorithm 33 Sampling a new paired substructure on the considered sequence fragment

procedure RandomlyFoldPairedSubstructure($\mathtt{start}, \mathtt{end}, \mathtt{helices}, \mathtt{unpRegs}$)
$\mathtt{tmp} =$ RandomlyFoldHairpinLoop($\mathtt{start}, \mathtt{end}, \mathtt{helices}, \mathtt{unpRegs}$)
$\mathtt{structureCompleted} = \mathtt{tmp}[1]$
while $\mathtt{structureCompleted} = \mathtt{False}$ **do**
$\mathtt{helices} = \mathtt{tmp}[2]$, $\mathtt{unpRegs} = \mathtt{tmp}[3]$, $\mathtt{ii} = \mathtt{tmp}[4,1]$, $\mathtt{jj} = \mathtt{tmp}[4,2]$
$\mathtt{tmp} =$ RandomlyExpandPairedSubstructure($\mathtt{start}, \mathtt{end}, \mathtt{ii}, \mathtt{jj}, \mathtt{helices}, \mathtt{unpRegs}$)
$\mathtt{structureCompleted} = \mathtt{tmp}[1]$
end while
return $\mathtt{tmp}$
end procedure

Algorithm 34 Sampling a new hairpin loop on the considered sequence fragment

procedure RandomlyFoldHairpinLoop($\mathtt{start}, \mathtt{end}, \mathtt{helices}, \mathtt{unpRegs}$)
/*Randomly draw a new hairpin loop on the considered sequence fragment $r_{\mathtt{start},\mathtt{end}}$:*/
$\mathtt{possChoices} =$ pcHL($\mathtt{start}, \mathtt{end}, \mathtt{helices}, \mathtt{unpRegs}$)
if $\mathtt{possChoices} = \emptyset$ **then**
/*No (more) hairpin possible, so the folding of the currently considered (composed) structure is finished:*/
return CompleteRandomlyFoldedStructure($\mathtt{start}, \mathtt{end}, \mathtt{helices}, \mathtt{unpRegs}$)
else
Sample $\mathtt{randChoice}$ from $\mathtt{possChoices}$ (according to the induced probability distribution)
$\mathtt{unpRegs} = \mathtt{unpRegs} \cup \{(\mathtt{randChoice}[1], (\mathtt{randChoice}[2] - \mathtt{randChoice}[1] + 1))\}$
$\mathtt{ii} = \mathtt{randChoice}[1]$, $\mathtt{jj} = \mathtt{randChoice}[2]$
$\mathtt{helices} = \mathtt{helices} \cup \{(\mathtt{ii}, \mathtt{jj}, 0)\}$
return $(\mathtt{False}, \mathtt{helices}, \mathtt{unpRegs}, (\mathtt{ii}, \mathtt{jj}))$
end if
end procedure

Algorithm 35 Sampling a valid extension of the currently considered paired substructure

procedure RandomlyExpandPairedSubstructure($start, end, ii, jj, helices, unpRegs$)
/*Randomly draw a valid extension (surrounding substructure) of the just folded paired structure $S_{ii,jj}$ on the considered sequence fragment $R_{start,end}$:*/
if $0 \leqslant helices[-1,3] < \min_{hel}$ /*length of last added helix is still too small*/ **then**
$possChoices = pcSP(start, end, ii, jj, helices, unpRegs)$
if $possChoices = \emptyset$ **then**
/*If $\min_{hel} \geqslant 2$, we may have sampled base pairs according to an effectively "wrong" rule ($P \rightarrow (L$ instead of $A \rightarrow (^{\min_{hel}} L)^{\min_{hel}}$) in the previous helix extension step, such that at this point, no extension of the considered (still incomplete) helix down from the currently first base pair ii.jj of the helix is possible according to the considered sampling probabilities (the corresponding inside and outside values derived according to the grammar model), which is due to the fact that the sampling strategy does not work conform with the underlying grammar (which constructs helices the other way round, by extending minimum length helices from the respective last base pair upwards).
Due to this fact, we actually have to sample base pairs successively in order to ensure that all possible helices (of all different lengths) might be folded by the algorithm, since always starting a new helix with $\min_{hel} \geqslant 2$ base pairs without sampling might eventually prohibit longer helices (for some similar reason as the above mentioned one). Nevertheless, due to the overall sampling strategy (and the used sampling probabilities), once a helical region is initiated, it is guaranteed that at least $\min_{hel}$ consecutive base pairs can be folded, so we should now force the remaining necessary pairs to be formed:*/
$missingBPs = \min_{hel} - helices[-1,3]$
$possChoices = \{((ii - missingBPs), (jj + missingBPs), 0)\}$
end if
else
$possChoices = pcSubStruct(start, end, ii, jj, helices, unpRegs)$
end if
if $possChoices = \emptyset$ **then**
/*No extension possible, so the folding of the currently considered (paired) substructure $S_{ii,jj}$ is finished:*/
return $(True, helices, unpRegs, (ii, jj))$
else
Sample $randChoice$ from $possChoices$ (according to the induced probability distribution)
/*Identify the type of extension (surrounding substructure type) implied by $randChoice$ and construct the corresponding (complete and valid) substructure on $R_{start,end}$ (containing the already folded structure $S_{ii,jj}$):*/
if $randChoice \in pcSP(start, end, ii, jj, helices, unpRegs)$ **or** $randChoice[3] = 0$ **then**
$tmp = ConstructBasePairs(randChoice, possChoices, ii, jj, helices, unpRegs)$
else if $randChoice \in pcBI(start, end, ii, jj, helices, unpRegs)$ **then**
$tmp = ConstructBulgeOrInteriorLoop(randChoice, possChoices, ii, jj, helices, unpRegs)$
else if $randChoice \in pcML(start, end, ii, jj, helices, unpRegs)$ **then**
$tmp = ConstrucMultiloop(randChoice, possChoices, start, end, ii, jj, helices, unpRegs)$
else if $randChoice \in pcEL(start, end, ii, jj, helices, unpRegs)$ **then**
$tmp = ConstructExtLoopSubstruct(randChoice, possChoices, ii, jj, helices, unpRegs)$
end if
return tmp
end if
end procedure

Algorithm 36 Construct base pair(s)

procedure ConstructBasePairs($randChoice, possChoices, ii, jj, helices, unpRegs$)
if $randChoice \in pcSP_{A\to(^{min_{hel}}L)^{min_{hel}}}(start, end, ii, jj, helices, unpRegs)$ **then**
  $helices[-1] = (randChoice[1], randChoice[2], helices[-1, 3] + min_{hel})$
else if $randChoice \in pcSP_{P\to(L)}(start, end, ii, jj, helices, unpRegs)$ **then**
  $helices[-1] = (randChoice[1], randChoice[2], helices[-1, 3] + 1)$
else if $randChoice[3] = 0$ /*special case (where no regular choice is possible)*/ **then**
  $helices[-1] = (randChoice[1], randChoice[2], min_{hel})$
end if
$ii = randChoice[1]$, $jj = randChoice[2]$
return $(False, helices, unpRegs, (ii, jj))$
end procedure

Algorithm 37 Construct bulge or interior loop

procedure ConstructBulgeOrInteriorLoop($randChoice, possChoices, ii, jj, helices, unpRegs$)
if $randChoice \in pcBI_{G\to BA}(start, end, ii, jj, helices, unpRegs)$ **then**
  $unpRegs = unpRegs \cup \{(randChoice[1], (ii-1) - randChoice[1] + 1)\}$
else if $randChoice \in pcBI_{G\to AB}(start, end, ii, jj, helices, unpRegs)$ **then**
  $unpRegs = unpRegs \cup \{(jj+1, randChoice[2] - (jj+1) + 1)\}$
else if $randChoice \in pcBI_{G\to BAB}(start, end, ii, jj, helices, unpRegs)$ **then**
  $unpRegs = unpRegs \cup \{(randChoice[1], (ii-1) - randChoice[1] + 1)\}$
  $unpRegs = unpRegs \cup \{(jj+1, randChoice[2] - (jj+1) + 1)\}$
end if
$ii = randChoice[1]$, $jj = randChoice[2]$
$helices = helices \cup \{(ii, jj, 0)\}$
return $(False, helices, unpRegs, (ii, jj))$
end procedure

Algorithm 38 Construct a substructure of the exterior loop

procedure ConstructExtLoopSubstruct($randChoice, possChoices, ii, jj, helices, unpRegs$)
if $randChoice \in pcEL_{T\to x}(start, end, ii, jj, helices, unpRegs)$ for $x \in \{C, CA, CAT\}$ **then**
  $unpRegs = unpRegs \cup \{(randChoice[1], (ii-1) - randChoice[1] + 1)\}$
else if $randChoice \in pcEL_{T\to x}(start, end, ii, jj, helices, unpRegs)$ for $x \in \{A, AT\}$ **then**
  nothing to do
end if
$ii = randChoice[1]$, $jj = randChoice[2]$
/*Since substructures of the exterior loop are not accessible from a closing base pair (except for the imaginary base pair $0.(n+1)$), the construction is already finished!*/
return $(True, helices, unpRegs, (ii, jj))$
end procedure

Algorithm 39 Construct a (complete and valid) multiloop

procedure ConstructMultiloop($randChoice, possChoices, start, end, ii, jj, helices, unpRegs$)
$unpRegs = unpRegs \cup \{(randChoice[1], (ii-1) - randChoice[1] + 1)\}$
$ii = randChoice[1]$, $jj = randChoice[2]$
if $(-ii, jj, -1) \in helices$ **then**
/*The structure $S_{ii,jj}$ is ought to become the last substructure (the k^{th} one, $k \geqslant 2$) of the currently folding multiloop (our random choice corresponds to $O \to UAN$ or $N \to UAN$, with $U \to \epsilon$ and $N \Rightarrow^* \epsilon$), but this has already been decided in the last step, so we can finish the construction:*/
return $(True, helices, unpRegs, (ii, jj))$
end if
/*Help restriction for k^{th} multiloop substructure ($k \geqslant 1$) is added, since at this point, we definitely know only the end of the currently folding multiloop, but not necessarily where it actually starts:*/
$helices = helices \cup \{(-ii, jj, -1)\}$
if $randChoice \in pcML_{k=1,M\to UAO}(start, end, ii, jj, helices, unpRegs)$ **then**
$mlStart = ii$, $mlEnd = jj$
else
/*$randChoice$ corresponds to rule $O \to UAN$ (and $k = 2$) or $N \to UAN$ (and $k = 3$):*/
$mlEnd = jj$
$mlStart = \max(\text{last paired position before } ii, 0)$ /*= end of preceding paired substructure*/ $+1$
$mlStart = mlStart + \min_{hel}$ /*space for multiloop foundation*/
$mlStart = \text{firstPossMLstarts}[1, mlStart, mlEnd]$
end if
if $mlEnd = end$ **then**
/*The construction of the currently folding multiloop has not been started just now, but that multiloop has already been partially formed (there exists at least one further multiloop substructure on $R_{start,mlEnd}$ in addition to the just folded substructure $S_{ii,mlEnd}$), which means we have to go back to the point where the construction of this multiloop was initiated, such that it can be completed there:*/
return $(True, helices, unpRegs, (ii, jj), k)$
end if
/*At this point, we obviously know that the construction of the considered multiloop has just started, which means that an arbitrary paired substructure (along with a potentially empty preceding unpaired region) of a new multiloop has been folded according to our random choice, and we now have to recursively fold further (paired) substructures in order to complete this loop:*/
while not $ii = 0$ **and** $jj = n + 1$ **do**
$res = \text{RandomlyFoldPairedSubstructure}(mlStart, mlEnd, helices, unpRegs)$
$helices = res[2]$, $unpRegs = res[3]$, $ii = res[4, 1]$, $jj = res[4, 2]$
$mlSubstructStarts = \{i \mid (-i, mlEnd, -1) \in helices\}$
if $jj = mlEnd$ **and** $res[5] = 1$ **then**
/*The recursively folded structure is ought to be the first substructure of the considered multiloop:*/
if $ii = \min(mlSubstructStarts)$ **then**
$mlStart = ii$
end if
end if
if $jj = n$ **then**
/*The recursively folded (paired) substructure has been determined to be one of the adjacent substructures of the exterior loop, which means it actually represents an independent paired structure on $R_{mlStart,mlEnd}$ preceding the considered multiloop:*/
$mlStart = helices[-1, 2]$ /*= end of just constructed preceding exterior loop substructure*/ $+1$
$mlStart = mlStart + \min_{hel}$ /*space for multiloop foundation*/
$mlStart = \min(\text{firstPossMLstarts}[1, \min(mlStart, n), mlEnd], \min(mlSubstructStarts))$
end if
end while
/*The considered multiloop is now complete (contains $k \geqslant 2$ recursively folded paired substructures radiating out from this loop), which means the overall folding of the entire sequence fragment $R_{mlStart,mlEnd}$ is finished, such that we only have to add the foundation:*/
$ii = mlStart$, $jj = mlEnd$
$helices = helices \cup \{(ii, jj, 0)\}$
return $(False, helices, unpRegs, (ii, jj))$
end procedure

Algorithm 40 Finishing the folding of the considered sequence fragment

procedure CompleteRandomlyFoldedStructure($start, end, helices, unpRegs$)
$unpRegs = unpRegs \cup \text{unfoldedRegions}(start, end, helices, unpRegs)$
$helices = helices \setminus \text{mlSubstructures}(start, end, helices)$
$unpRegs = unpRegs \setminus \text{emptyStrands}(start, end, unpRegs)$
return $(\text{True}, helices, unpRegs, (0, n+1))$
end procedure

In particular, besides the strongly relevant sets of all possible substructures that can be validly folded on a particular sequence fragment depending on the previously formed substructures, the following simple definitions are employed, which require no further explanation:

$$\begin{aligned}&\text{remainingFragments}(start, end, helices, unpRegs) :=\\ &\qquad \big\{(i,j) \mid start \leqslant i < j \leqslant end \text{ and } R_{i,j} \text{ not folded and}\\ &\qquad\qquad (r_{i-1} \text{ folded or } i = start) \text{ and } (r_{j+1} \text{ folded or } j = end)\big\},\end{aligned}$$

$$\begin{aligned}&\text{unfoldedRegions}(start, end, helices, unpRegs) :=\\ &\qquad \big\{(i, j-i+1) \mid start \leqslant i < j \leqslant end \text{ and } R_{i,j} \text{ not folded and}\\ &\qquad\qquad (r_{i-1} \text{ folded or } i = start) \text{ and } (r_{j+1} \text{ folded or } j = end)\big\},\end{aligned}$$

and

$$\begin{aligned}\text{mlSubstructures}(start, end, helices) &:= \{(i, j, -1) \in helices \mid start \leqslant i < j \leqslant end\},\\ \text{emptyStrands}(start, end, unpRegs) &:= \{(i, 0) \in unpRegs \mid start \leqslant i \leqslant end\}.\end{aligned}$$

However, the key point is that the type (or shape) and actual composition (of base pairs and unpaired bases) of a particular extension of the already folded substructure $S_{i,j}$ on the considered fragment $R_{start,end}$ of the input sequence r are randomly drawn according to the probability distributions induced by the respective sets of all valid choices. Actually, the presented sampling strategy relies on the following (more or less complex) formal set definitions for the respective relevant (valid) mutually exclusive and exhaustive cases in order to perform the needed random choices:

First, the set of all possible hairpin loops that can be folded on sequence fragment $R_{start,end}$ is given by

$$\begin{aligned}&\text{pcHL}(start, end, helices, unpRegs) :=\\ &\qquad \big\{(i, j, prob) \mid start + \min_{hel} \leqslant i \leqslant j \leqslant end - \min_{hel} \text{ and}\\ &\qquad\qquad i + \min_{HL} - 1 \leqslant j \leqslant i + \max_{hairpin} - 1 \text{ and}\\ &\qquad\qquad R_{i-\min_{hel}, j+\min_{hel}} \text{ not folded and}\\ &\qquad\qquad prob = \Big(\widehat{\beta}_L(i,j) \cdot (\widehat{\alpha}_F(i,j) \cdot \Pr(L \to F))\Big) \neq 0\big\}.\end{aligned}$$

It is important to mention that $\max_{hairpin}$ ensures that $\text{pcHL}(start, end, helices, unpRegs)$ can be computed in $\mathcal{O}(n)$ time, since in the worst-case we have to consider $\mathcal{O}(n)$ possible start positions i, and for each i there are at most $\max_{hairpin}$, that is $\mathcal{O}(1)$, possible end positions for the corresponding hairpin loop. However, for stacked pairs, the algorithm considers

$$\begin{aligned}&\text{pcSP}(start, end, ii, jj, helices, unpRegs) :=\\ &\qquad \bigcup_{rule \in \{A \to (^{\min_{hel}} L)^{\min_{hel}}, P \to (L)\}} \text{pcSP}_{rule}(start, end, ii, jj, helices, unpRegs),\end{aligned}$$

where

$$\begin{aligned}
&\mathrm{pcSP}_{A\to(^{\min_{\mathrm{hel}}}L)^{\min_{\mathrm{hel}}}}(start, end, ii, jj, helices, unpRegs) := \\
&\qquad \big\{(i,j,prob) \mid start \leqslant i \leqslant j \leqslant end \text{ and} \\
&\qquad\qquad i = ii - \min_{\mathrm{hel}} \text{ and } j = jj + \min_{\mathrm{hel}} \text{ and} \\
&\qquad\qquad R_{i,i+\min_{\mathrm{hel}}-1} \text{ and } R_{j-\min_{\mathrm{hel}}+1,j} \text{ not folded and} \\
&\qquad\qquad prob = \left(\widehat{\beta}_A(i,j)\cdot\widehat{\alpha}_A(i,j)\right) \neq 0\big\},
\end{aligned}$$

$$\begin{aligned}
&\mathrm{pcSP}_{P\to(L)}(start, end, ii, jj, helices, unpRegs) := \\
&\qquad \big\{(i,j,prob) \mid start \leqslant i \leqslant j \leqslant end \text{ and} \\
&\qquad\qquad i = ii - 1 \text{ and } j = jj + 1 \text{ and} \\
&\qquad\qquad r_i \text{ and } r_j \text{ not folded and} \\
&\qquad\qquad prob = \left(\widehat{\beta}_L(i,j)\cdot(\widehat{\alpha}_P(i,j)\cdot Pr(L\to P))\right) \neq 0\big\}.
\end{aligned}$$

As for bulge and interior loops, the set of all possible choices according to a just folded paired structure $S_{ii,jj}$ and a currently considered sequence fragment $R_{start,end}$ is defined as

$$\begin{aligned}
&\mathrm{pcBI}(start, end, ii, jj, helices, unpRegs) := \\
&\qquad\qquad \bigcup_{x\in\{BA,AB,BAB\}} \mathrm{pcBI}_{G\to x}(start, end, ii, jj, helices, unpRegs),
\end{aligned}$$

with

$$\begin{aligned}
&\mathrm{pcBI}_{G\to BA}(start, end, ii, jj, helices, unpRegs) := \\
&\qquad \big\{(i,j,prob) \mid start + \min_{\mathrm{hel}} \leqslant i \leqslant j \leqslant end - \min_{\mathrm{hel}} \text{ and} \\
&\qquad\qquad ii - \max_{bulge} \leqslant i \leqslant ii - 1 \text{ and } j = jj \text{ and} \\
&\qquad\qquad R_{i-\min_{\mathrm{hel}},ii-1} \text{ and } R_{j+1,j+\min_{\mathrm{hel}}} \text{ not folded and} \\
&\qquad\qquad prob = \left(\widehat{\beta}_G(i,j)\cdot(\widehat{\alpha}_B(i,ii-1)\cdot\widehat{\alpha}_A(ii,j)\cdot Pr(G\to BA))\right) \neq 0\big\},
\end{aligned}$$

$$\begin{aligned}
&\mathrm{pcBI}_{G\to AB}(start, end, ii, jj, helices, unpRegs) := \\
&\qquad \big\{(i,j,prob) \mid start + \min_{\mathrm{hel}} \leqslant i \leqslant j \leqslant end - \min_{\mathrm{hel}} \text{ and} \\
&\qquad\qquad i = ii \text{ and } jj + 1 \leqslant j \leqslant jj + \max_{bulge} \text{ and} \\
&\qquad\qquad R_{i-\min_{\mathrm{hel}},i-1} \text{ and } R_{jj+1,j+\min_{\mathrm{hel}}} \text{ not folded and} \\
&\qquad\qquad prob = \left(\widehat{\beta}_G(i,j)\cdot(\widehat{\alpha}_A(i,jj)\cdot\widehat{\alpha}_B(jj+1,j)\cdot Pr(G\to AB))\right) \neq 0\big\},
\end{aligned}$$

$$\begin{aligned}
&\mathrm{pcBI}_{G\to BAB}(start, end, ii, jj, helices, unpRegs) := \\
&\big\{(i,j,prob) \mid start + \min_{\mathrm{hel}} \leqslant i \leqslant j \leqslant end - \min_{\mathrm{hel}} \text{ and} \\
&\qquad ii - \max_{bulge} \leqslant i \leqslant ii - 1 \text{ and} \\
&\qquad jj + 1 \leqslant j \leqslant jj + \max_{bulge} \text{ and} \\
&\qquad R_{i-\min_{\mathrm{hel}},ii-1} \text{ and } R_{jj+1,j+\min_{\mathrm{hel}}} \text{ not folded and} \\
&\qquad prob = \left(\widehat{\beta}_G(i,j)\cdot(\widehat{\alpha}_B(i,ii-1)\cdot\widehat{\alpha}_A(ii,jj)\cdot\widehat{\alpha}_B(jj+1,j)\cdot Pr(G\to BAB))\right) \neq 0\big\}.
\end{aligned}$$

It is easy to see that $\max_{bulge}$ is indeed necessary to ensure that any subset $\mathrm{pcBI}_{rule}(\cdot)$ can be computed in $\mathcal{O}(n)$ time. The same holds for $\max_{strand}$ in the case of multiloops, as can be observed from the following definitions:

$$\begin{aligned}\text{pcML}(&\mathit{start},\mathit{end},ii,jj,\mathit{helices},\mathit{unpRegs}) :=\\ &\text{pcML}_{k=1,M\to UAO}(\mathit{start},\mathit{end},ii,jj,\mathit{helices},\mathit{unpRegs}) \cup\\ &\text{pcML}_{k=2,O\to UAN}(\mathit{start},\mathit{end},ii,jj,\mathit{helices},\mathit{unpRegs}) \cup\\ &\text{pcML}_{k=3,N\to UAN}(\mathit{start},\mathit{end},ii,jj,\mathit{helices},\mathit{unpRegs}),\end{aligned}$$

where

$$\begin{aligned}&\text{pcML}_{k,X\to UAY}(\mathit{start},\mathit{end},ii,jj,\mathit{helices},\mathit{unpRegs}) :=\\ &\{(i,j,\mathit{prob}) \mid \mathit{start} \leqslant i \leqslant j \leqslant \mathit{end} - \min_{\text{hel}} \text{ and}\\ &\quad ii - \max_{\mathit{strand}} \leqslant i \leqslant ii - 0 \text{ and}\\ &\quad ([\text{last folded position } b \text{ before } ii \text{ is end of multiloop substructure and}\\ &\quad\ \text{validChoice}_{k,X}(\max(b+1,\mathit{start}),jj,i,j) = \mathit{True}] \text{ or}\\ &\quad\ [\text{last folded position } b \text{ before } ii \text{ belongs to exterior loop and}\\ &\quad\ \text{validChoice}_{k,X}(\max(b+1+\min_{\text{hel}},\mathit{start}),jj,i,j) = \mathit{True}]) \text{ and}\\ &\quad \mathit{prob} = \left(\widehat{\beta}_X(i,j)\cdot(\widehat{\alpha}_U(i,ii-1)\cdot\widehat{\alpha}_A(ii,jj)\cdot\widehat{\alpha}_Y(jj+1,j)\cdot\Pr(X\to UAY))\right) \neq 0\},\end{aligned}$$

and[9]

$$\begin{aligned}&\text{validChoice}_{k,X}(\mathit{start},jj,i,j) :=\\ &\quad [i \geqslant \mathit{start} + (k-1)\cdot\min_{\text{ps}} \text{ and } (j-i+1) \geqslant (2-(k-1))\cdot\min_{\text{ps}}] \text{ and}\\ &\quad \left(\widehat{\beta}_X(i,j)\cdot\widehat{\alpha}_X(i,j)\right) \neq 0 \text{ and}\\ &\quad i \geqslant \text{firstPossMLstarts}[k,\min(\mathit{start}+(k-1)\cdot\min_{\text{ps}},n),j] \text{ and}\\ &\quad j - \text{firstPossMLstarts}[1,\mathit{start},j] + 1 \geqslant 2\cdot\min_{\text{ps}} \text{ and}\\ &\quad R_{\text{firstPossMLstarts}[1,\mathit{start},j]-\min_{\text{hel}},\text{firstPossMLstarts}[1,\mathit{start},j]-1} \text{ and } R_{j+1,j+\min_{\text{hel}}} \text{ not folded and}\\ &\quad ([k=1 \text{ and furtherPSpossible}[jj+1,j]] \text{ or}\\ &\quad\ [k \geqslant 2 \text{ and furtherPSpossible}[\text{firstPossMLstarts}[1,\mathit{start},j],i-1]]).\end{aligned}$$

Finally, for exterior loops the strategy initially considers the formal set definition

$$\begin{aligned}\text{pcEL}(&\mathit{start},\mathit{end},ii,jj,\mathit{helices},\mathit{unpRegs}) :=\\ &\bigcup_{x\in\{C,A,CA,AT,CAT\}} \text{pcEL}_{T\to x}(\mathit{start},\mathit{end},ii,jj,\mathit{helices},\mathit{unpRegs}),\end{aligned}$$

where

$$\begin{aligned}&\text{pcEL}_{T\to C}(\mathit{start},\mathit{end},ii,jj,\mathit{helices},\mathit{unpRegs}) :=\\ &\quad \{(i,n,\mathit{prob}) \mid i \leqslant ii \text{ and } jj = n \text{ and}\\ &\qquad R_{i,n} \text{ completely unpaired (contains no base pairs) and}\\ &\qquad \text{validChoice}_{T\to C}(\mathit{end},ii,jj,i,\mathit{helices}) = \mathit{True} \text{ and}\\ &\qquad \mathit{prob} = \left(\widehat{\beta}_T(i,n)\cdot(\widehat{\alpha}_C(i,n)\cdot\Pr(T\to C))\right) \neq 0\} \;=\; \emptyset,\end{aligned}$$

$$\begin{aligned}&\text{pcEL}_{T\to A}(\mathit{start},\mathit{end},ii,jj,\mathit{helices},\mathit{unpRegs}) :=\\ &\quad \{(i,n,\mathit{prob}) \mid i = ii \text{ and } jj = n \text{ and}\end{aligned}$$

[9] Note that $\text{validChoice}_{k,X}$ (and also $\text{validChoice}_{\mathit{rule}}$ which will be defined later) is a boolean function, returning either *True* or *False*. It checks most of the complex restrictions that need to be evaluated dynamically with respect to the corresponding sampling step.

$$\text{validChoice}_{T\to A}(end, ii, jj, i, helices) = \text{True and}$$
$$prob = \left(\widehat{\beta}_T(i,n)\cdot(\widehat{\alpha}_A(i,n)\cdot Pr(T\to A))\right) \neq 0\},$$

$$pcEL_{T\to CA}(start, end, ii, jj, helices, unpRegs) :=$$
$$\{(i,n,prob) \mid i \leqslant ii-1 \text{ and } jj = n \text{ and}$$
$$R_{i,ii-1} \text{ not folded and}$$
$$\text{validChoice}_{T\to CA}(end, ii, jj, i, helices) = \text{True and}$$
$$prob = \left(\widehat{\beta}_T(i,n)\cdot(\widehat{\alpha}_C(i,ii-1)\cdot\widehat{\alpha}_A(ii,n)\cdot Pr(T\to CA))\right) \neq 0\},$$

$$pcEL_{T\to AT}(start, end, ii, jj, helices, unpRegs) :=$$
$$\{(i,n,prob) \mid i = ii \text{ and } jj \leqslant n-1 \text{ and}$$
$$\text{validChoice}_{T\to AT}(end, ii, jj, i, helices) = \text{True and}$$
$$prob = \left(\widehat{\beta}_T(i,n)\cdot(\widehat{\alpha}_A(i,jj)\cdot\widehat{\alpha}_T(jj+1,n)\cdot Pr(T\to AT))\right) \neq 0\},$$

$$pcEL_{T\to CAT}(start, end, ii, jj, helices, unpRegs) :=$$
$$\{(i,n,prob) \mid i \leqslant ii-1 \text{ and } jj \leqslant n-1 \text{ and}$$
$$R_{i,ii-1} \text{ not folded and}$$
$$\text{validChoice}_{T\to CAT}(end, ii, jj, i, helices) = \text{True and}$$
$$prob = \left(\widehat{\beta}_T(i,n)\cdot(\widehat{\alpha}_C(i,ii-1)\cdot\widehat{\alpha}_A(ii,jj)\cdot\widehat{\alpha}_T(jj+1,n)\cdot Pr(T\to CAT))\right) \neq 0\},$$

and

$$\text{validChoice}_{rule}(end, ii, jj, i, helices) :=$$
$S_{ii,jj}$ is not accessible from any base pair (except from the imaginary pair $0.(n+1)$) and
([last folded pos. b before ii is end of m.l. substruct. and $i \geqslant \max(b+1+\min_{hel}, 1) =: start$] or
[last folded pos. b before ii belongs to exterior loop and $i \geqslant \max(b+1, 1) =: start$]) and
([first folded pos. b after jj is start of multiloop substruct. and $jj \leqslant \min(b-1-\min_{hel}, end)$] or
[first folded pos. b after jj belongs to exterior loop and $jj \leqslant \min(b-1, end)$]) and
([*rule* $= T\to A$ or *rule* $= T\to AT$] and
[a paired substruct. ending at pos. $ii-1$ can be folded on $R_{start,ii-1}$ or already exists]) and
([currStarts $= \{i \mid (-i, end, -1) \in helices\} = \emptyset$] or
[nextPossStart $=$ firstPossMLstarts$[1, \min(jj+1+\min_{hel}, n), end] < \min(\text{currStarts}) - \min_{ps} + 1$
and furtherPSpossible$[\text{nextPossStart}, \min(\text{currStarts})-1]$]).

Last but not least, the set of all possible extensions of a paired substructure $S_{ii,jj}$ on a considered sequence fragment $R_{start,end}$ is obviously given by

$$pcSubStruct(start, end, ii, jj, helices, unpRegs) :=$$
$$pcSP(start, end, ii, jj, helices, unpRegs) \cup pcBI(start, end, ii, jj, helices, unpRegs) \cup$$
$$pcML(start, end, ii, jj, helices, unpRegs) \cup pcEL(start, end, ii, jj, helices, unpRegs).$$

It remains to mention that after a preprocessing of the given input sequence – including the complete DP method for deriving all inside and outside values $\widehat{\alpha}_X(i,j)$ and $\widehat{\beta}_X(i,j)$, for $X \in \mathcal{I}_{\mathcal{G}_s}$ and $1 \leqslant i,j \leqslant n$, as well as the subsequent calculation of the additionally needed matrices firstPossMLstarts and furtherPSpossible – each of the probabilities prob defined for

a particular choice of either a new hairpin loop or a possible valid extension of the just folded (paired) substructure $S_{i,j}$ on sequence fragment $R_{start,end}$ in the respective set of all possible choices (pcHL$(\cdot)$ or a subset of pcSubStruct$(\cdot)$) can be derived in constant time[10]. Furthermore, according to the implied restrictions for particular loops (only constant values for $max_{hairpin}$, max_{bulge} and max_{strand} are allowed), each of the previously defined sets of possible choices contains only $\mathcal{O}(n)$ elements in the worst-case, that is $\texttt{card}(\text{pcHL}(\cdot)) \in \mathcal{O}(n)$ and $\texttt{card}(\text{any subset of pcSubStruct}(\cdot)) \in \mathcal{O}(n)$. Hence, the sampling strategy needs $\mathcal{O}(n)$ time for (dynamically) deriving a particular probability distribution and drawing a corresponding random choice.

Moreover, according to the employed strategy for sampling (complete and valid) paired substructures on a considered sequence fragment $R_{start,end}$, it is guaranteed that after a constant number of random choices[11], at least one new base pair (and an arbitrary number of single-stranded bases) is folded on $R_{start,end}$ and can thus be added to the overall secondary structure $S_{1,n}$. Consequently, there results $\mathcal{O}(n)$ time complexity for sampling (at least) one random base pair, and since any structure of size n can have at most $\lfloor \frac{n-min_{HL}}{2} \rfloor \in \mathcal{O}(n)$ base pairs, the time requirements of the sampling strategy for constructing the entire secondary structure $S_{1,n}$ is bounded by $\mathcal{O}(n^2)$.

9.4.3. Discussion

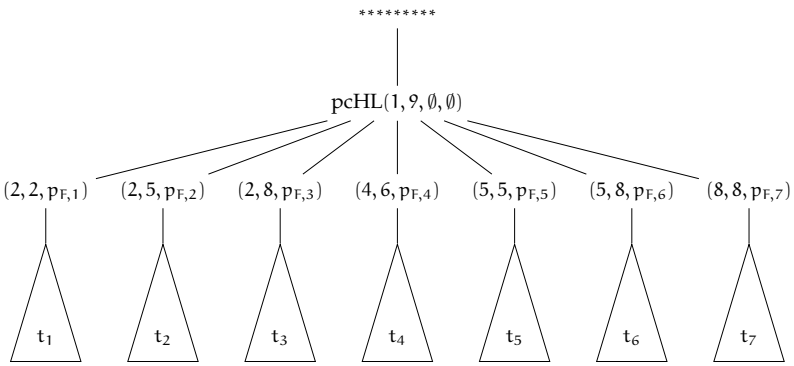
On the basis of the formal definitions introduced in Section 9.4.2, we can easily construct a corresponding decision tree for any particular input sequence. However, due to the high variety during the sampling process and the fact that most of the feasible structures are often generated in many different ways, even short sequences imply rather complex trees. Nevertheless, considering such a decision tree does not only contribute to the understanding of the course of action of the sampling strategy, but also reveals where and why some of the previously mentioned problems connected to inside-out sampling directions arise. Therefore, we decided to construct the corresponding decision tree for the rather short exemplary input sequence cagcagcag under the assumption of the structural parameters $min_{hel} = min_{HL} = 1$. Additionally, in contrast to all evaluations performed within this thesis, we here assume that only canonical base pairs are allowed in order to keep the number of feasible secondary structures low. The resulting tree is presented in Figure 9.2.

Obviously, any path from the root node to a leaf corresponds to a particular sampling process[12]. Furthermore, we immediately observe that there are indeed structures that correspond to different paths, that is which can be generated in more than one way. Looking at the different leaves, it is easy to see that five secondary structures that are conform with the considered sequence are missing: (∘)(∘)∘∘∘, (∘)∘∘∘∘∘∘, (∘∘∘∘)∘∘∘, ∘∘∘(∘)∘∘∘ and ∘∘∘∘∘∘∘∘∘. Common to all these structures is that the last component of the exterior loop is a single-stranded region, where in all cases an additional canonical base pair is possible on the corresponding sequence fragment. Notably, structures in which any component of the exterior loop but the last is unpaired can actually be sampled by the strategy, no matter if an additional canonical pair is

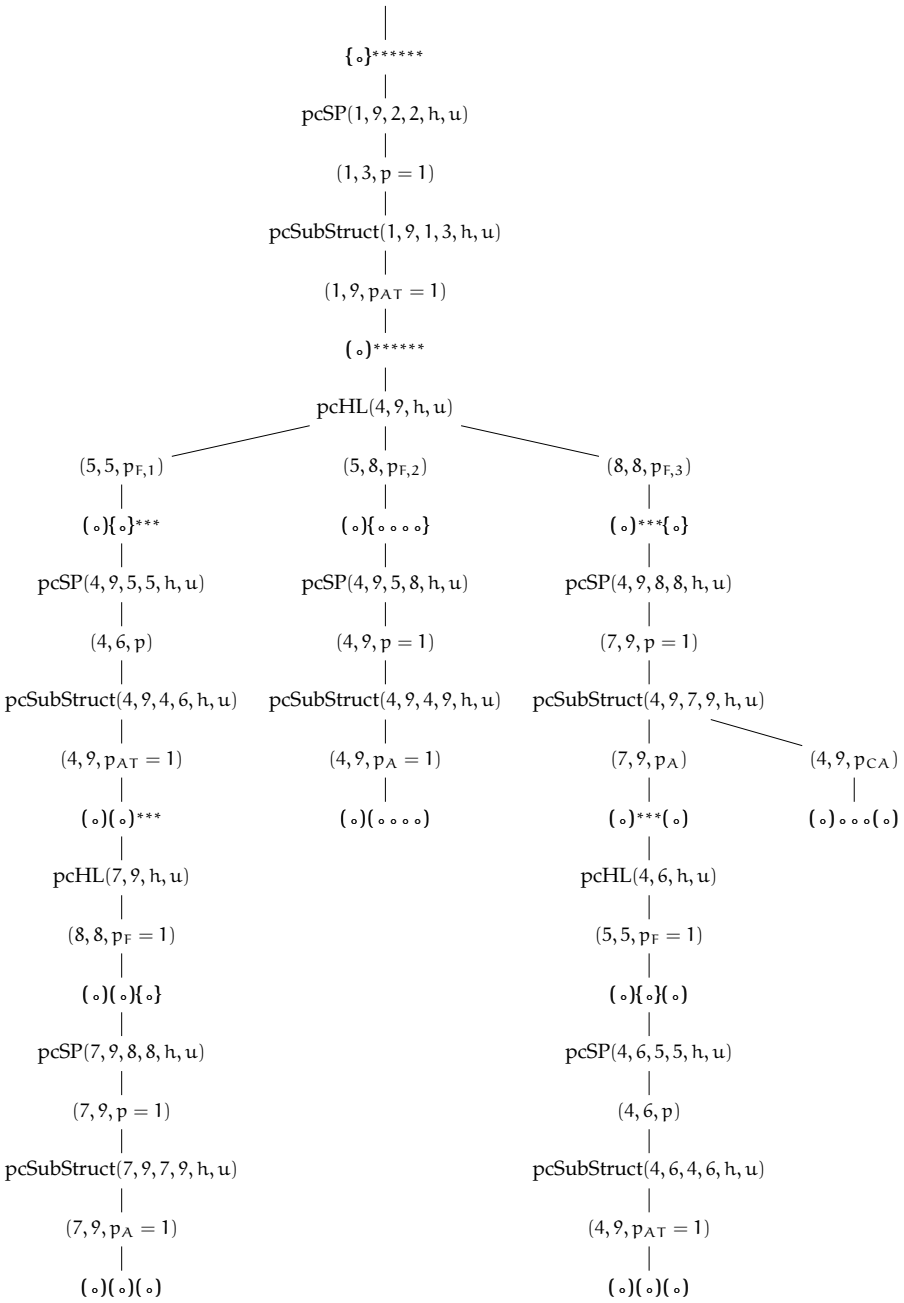
[10] By using a clever implementation, which for example preliminary computes the last folded position lowerBound before ii and the first folded position upperBound after jj (dynamically according to the current state of helices and unpRegs), which requires $\mathcal{O}(n)$ time for sequence length n. Since then, the probabilities prob of the possible choices $(i, j, prob)$ in a particular set pc can be efficiently derived by checking whether $lowerBound < i$ and $j < upperBound$; this can then actually be done in $\texttt{card}(pc) \cdot \mathcal{O}(1)$ time. In fact, we do not need to go strictly conform with the definition of the set pc and explicitly validate the formulated condition "$R_{i,j}$ not folded", as this would indeed require $\texttt{card}(pc) \cdot \mathcal{O}(n)$ time.

[11] For example sampling a new hairpin, then drawing a valid extension, or else additionally sampling the length of the preceding unpaired region and corresponding substructure type after having drawn the previous extension of a paired substructure.

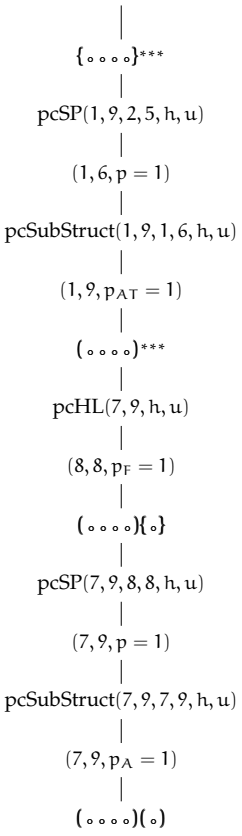
[12] Note that in Figure 9.2g, the probabilities 1/2 for using the left or the right path in order to sample the resulting identical structures is due to the arbitrary order of elements in $unfoldedFragments = \{(1,3),(7,9)\}$, see Algorithm 32.



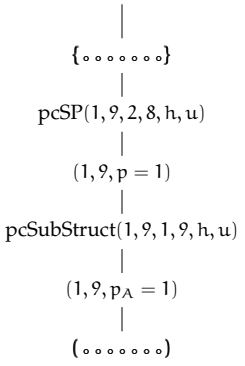
(a) Overall decision tree.



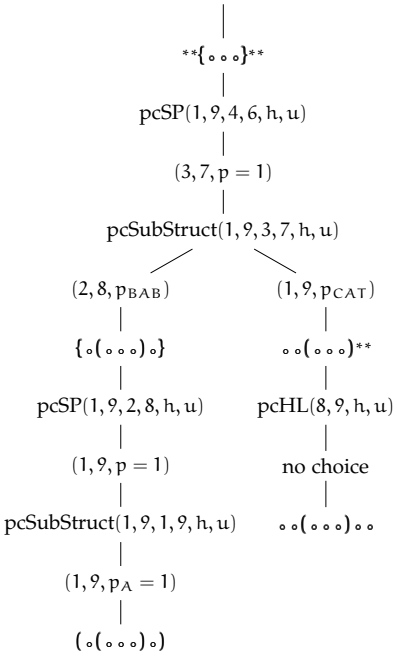
(b) Decision tree t_1.



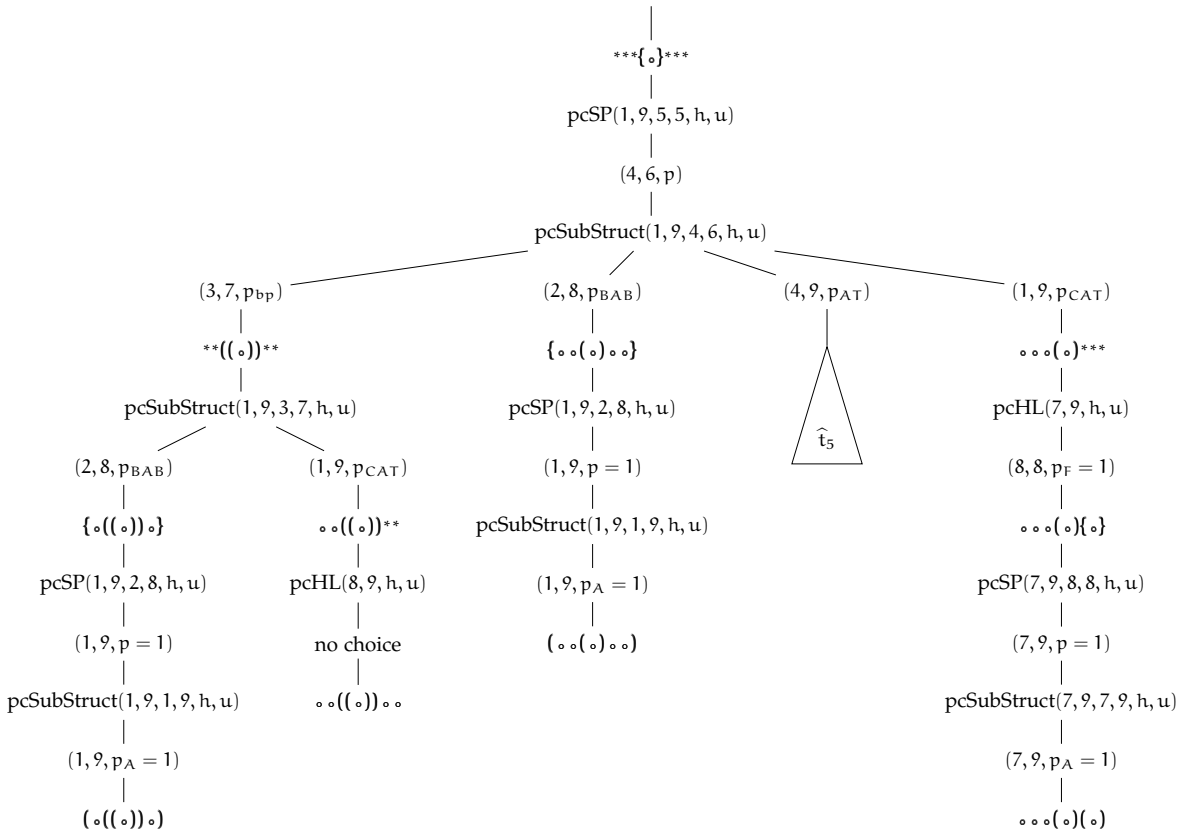
(c) Decision tree t_2.



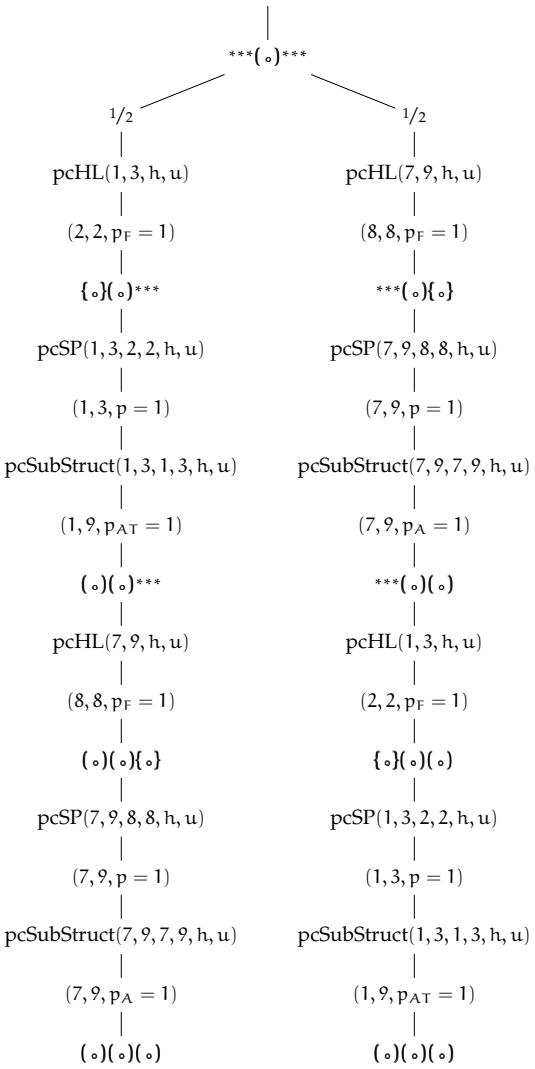
(d) Decision tree t_3.



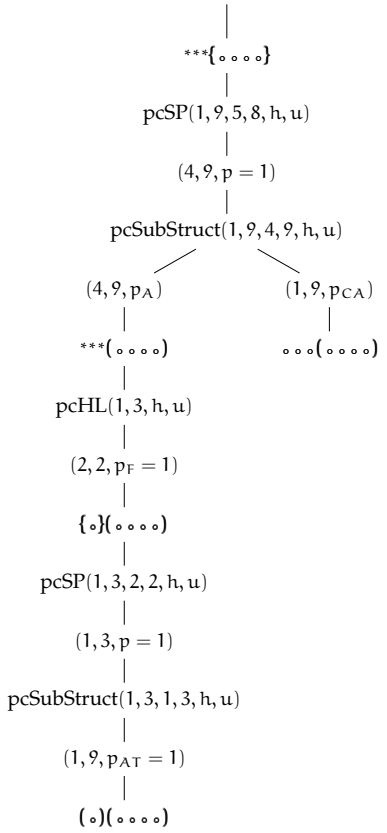
(e) Decision tree t_4.



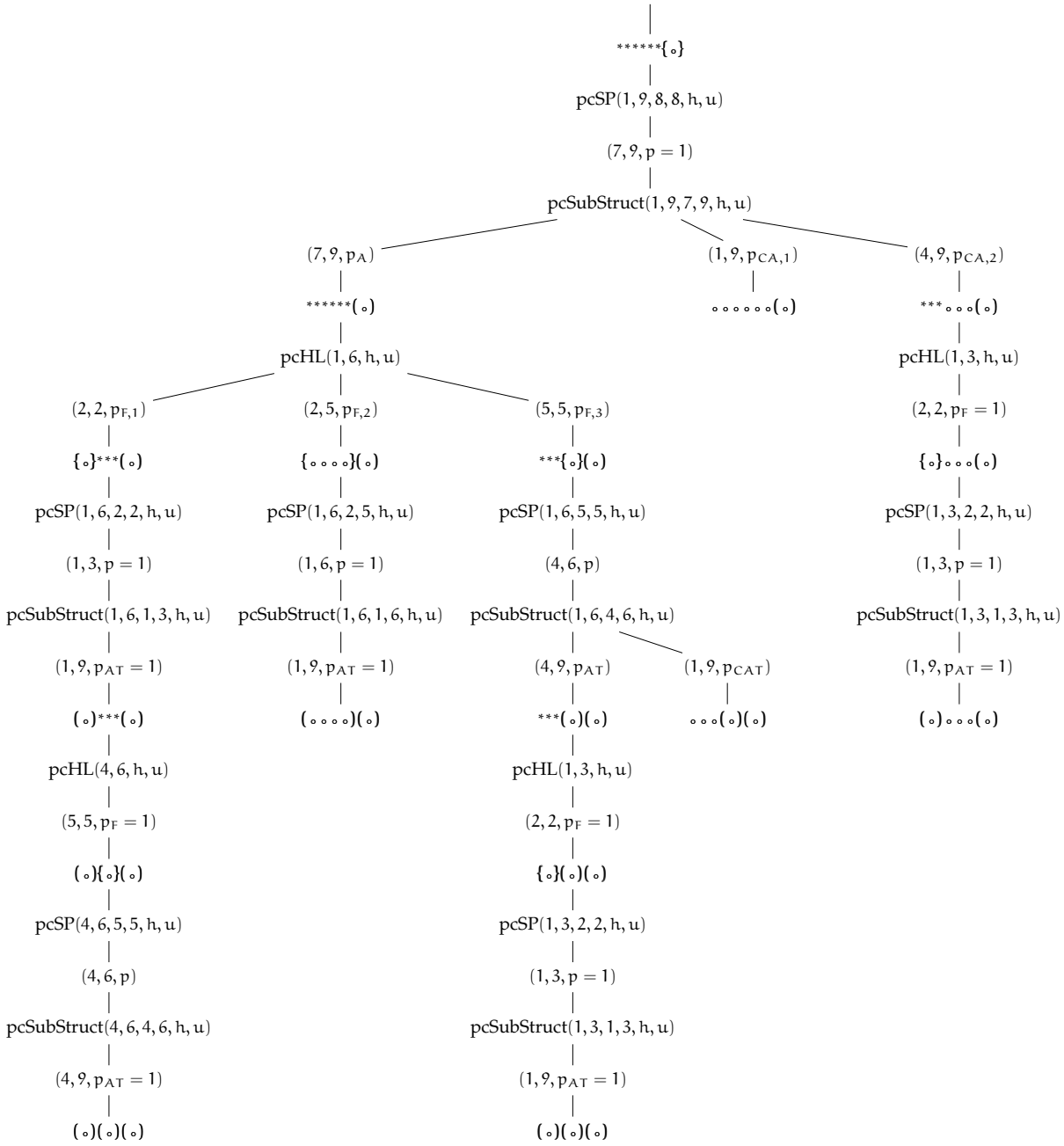
(f) Decision tree t_5.



(g) Decision tree $\widehat{t}_5$.



(h) Decision tree t_6.



(i) Decision tree t_7.

Figure 9.2.: **Exemplary decision trees for the dynamic sampling strategy.** Figures show (particular subtrees of) the overall decision tree associated with the dynamic sampling strategy for the input sequence $cagcagcag$. They were created under the assumptions of $\min_{hel} = \min_{HL} = 1$ and that only canonical base pairs are allowed in the underlying stochastic model. Note that h and u denote the sets $helices$ and $unpRegs$ that represent the currently folded secondary structure at any point of the sampling process. Sampling probabilities are denoted by p, where we used p_F and p_x, $x \in \{C, A, CA, AT, CAT\}$, to explicitly mark the conclusion of the production rule $L \to F$ or $T \to x$ that corresponds to the respective randomly drawn choice. Furthermore, we used the same representation for the partially formed structures as in Example 9.4.1.

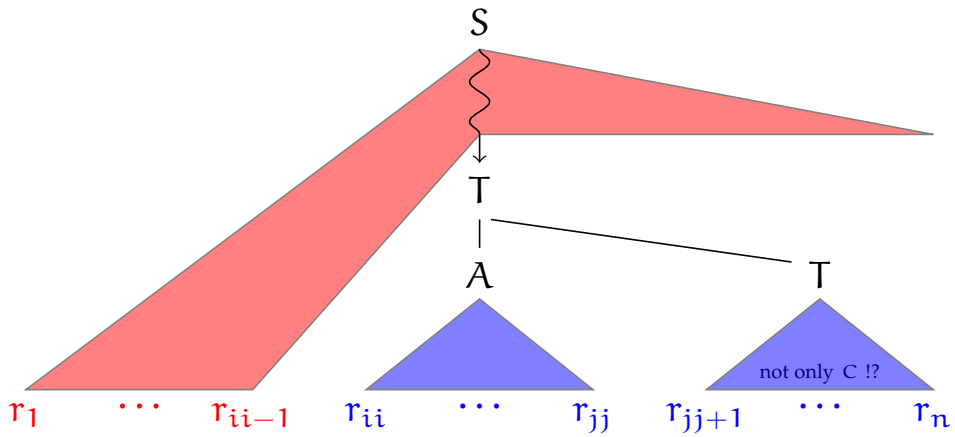
possible on the corresponding fragment. This is due to the fact that the strategy (outwardly) extends substructures according to sampling probabilities induced by the productions of the underlying grammar $\mathcal{G}_s$, which generates adjacent substructures (in exterior and multiple loops) from left to right.

For instance, in case of exterior loops the adjacent substructures are generated by productions with premise T using the grammar $\mathcal{G}_s$, where any paired substructure (derived from symbol A) might be constructed along with a preceding unpaired region (derived from C), and the subsequent substructures (if any) are generated thereafter (again from symbol T). Hence, if for $x \in \{CA, CAT\}$, an extension $(i, n, p_x) \in \text{pcSubStruct}(start, end, ii, jj, helices, unpRegs)$ of a just folded paired substructure $S_{ii,jj}$ on a considered sequence fragment $R_{start,end}$ is sampled, then $S_{i,ii-1}$ becomes unpaired and $R_{i,ii-1}$ will not be considered again by the strategy. However, after an extension $(i, n, p_x) \in \text{pcSubStruct}(start, end, ii, jj, helices, unpRegs)$ for $x \in \{AT, CAT\}$, fragment $R_{jj+1,n}$ will again be considered in subsequent sampling steps. If possible, the sampling strategy then inherently generates an additional hairpin loop on this fragment, as discussed in Section 9.4.1 and illustrated by Figure 9.3a.

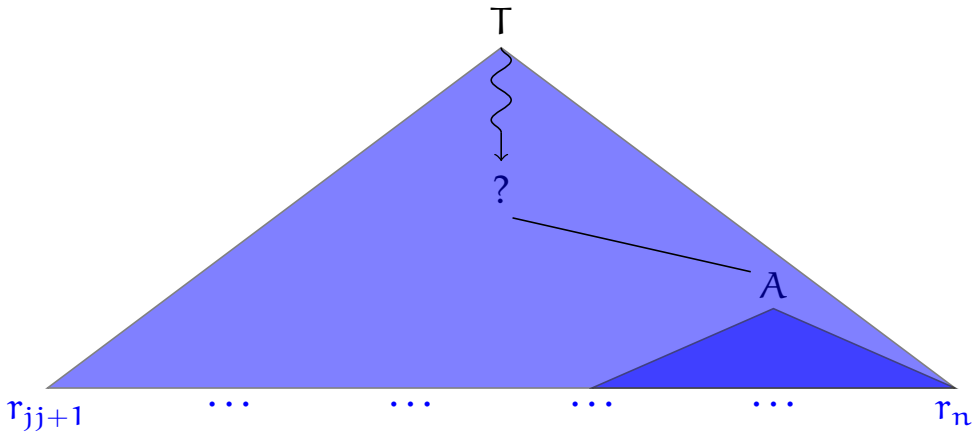
In fact, our strategy deliberately excludes the probability for leaving $R_{jj+1,n}$ single-stranded from the corresponding sampling distribution due to axiom Ax_1. Note that in the absence of this axiom, for instance in cases of exact preprocessing, we might additionally consider the corresponding probability $\widehat{\beta}_C(jj+1, n) \cdot \widehat{\alpha}_C(jj+1, n)$ when deriving that sampling distribution. In this context, it is worth mentioning that when considering the fragment $R_{jj+1,n}$, the strategy easily recognizes that the unpaired region must be generated from symbol C rather than from symbol U which produces single-strands in multiloops. However, when considering other fragments $R_{i,j}$, $1 < i, j < n$, it is not always obvious if the unpaired region $S_{i,j}$ must be generated from C or U. This must then be dynamically checked depending on the partially formed structure. It should be clear that equivalent problems concerning single-stranded regions as rightmost structural components occur in case of multiloops, since the adjacent substructures are generated in basically the same way as for exterior loops, using similar production rules which principally only differ in the corresponding intermediate symbols.

Anyway, as already mentioned, inside-to-outside strategies conceptually imply the consideration of fuzzy conditional sampling distributions. The reason for this basically lies in the fact that the inside (and outside) values include probabilities for all possible foldings, but due to the sampling direction, in many cases the probabilities of some foldings need to be excluded. For the sake of simplicity and to enable comparisons to the presented decision tree, we decided to exemplarily consider extensions of a paired substructure $S_{ii,jj}$ generated by intermediate symbol A to a substructure $S_{i,jj}$, $i \leqslant ii$, induced by production rules $T \to AT$ or $T \to CAT$. First, note that in comparison to $T \to AT$, the probabilities for the respective mutually exclusive and exhaustive cases corresponding to $T \to CAT$ additionally depend on a specific inside value $\widehat{\alpha}_C(i, ii-1)$ and are only considered if $R_{i,ii-1}$ has not yet been folded (according to the definition of $\text{pcEL}_{T \to CAT}(start, end, ii, jj, helices, unpRegs)$). Since in these cases, $S_{i,ii-1}$ becomes unpaired by deriving C in a unique way, $\widehat{\alpha}_C(i, ii-1)$ is a proper contribution to the respective sampling probability, and we may without loss of generality consider only the rule $T \to AT$ for our argumentation. The interrelationship of the corresponding inside and outside values is illustrated in Figure 9.3a.

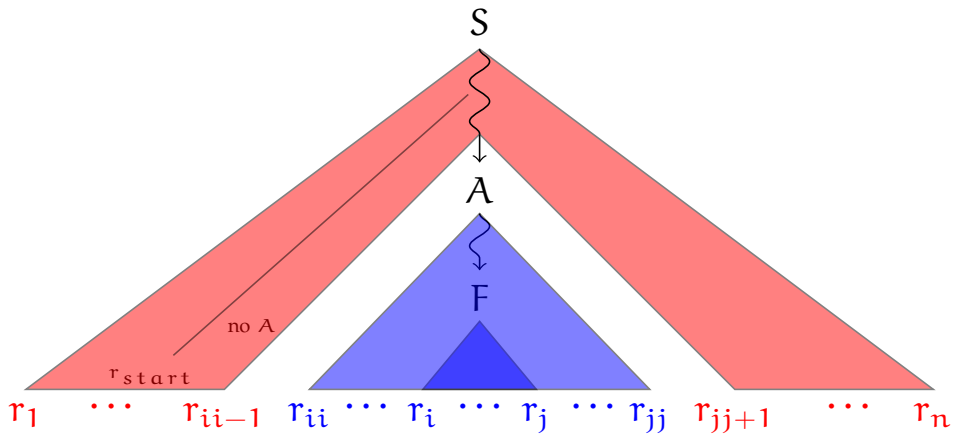
Essentially, whenever the strategy calculates a sampling probability corresponding to production $T \to AT$, a particular paired substructure $S_{ii,jj}$ which can be generated from the nonterminal symbol A has already been folded in preceding sampling steps. Hence, if other paired substructures $S'_{ii,jj}$ could be generated from A, then the inside value $\widehat{\alpha}_A(ii, jj)$ falsely includes the corresponding probabilities. In these cases, $\widehat{\alpha}_A(ii, jj)$ thus provides no proper contribution since it does not equate to the probability of the sampled paired substructure $S_{ii,jj}$. Note that such inconsistencies in the used inside values can not occur in the context of the common sampling strategy, as sampling the next base pair or substructure type corresponding



(a) Extensions induced by $T \rightarrow AT$.



(b) Problematic situation caused by arbitrary sampling order.



(c) Problematic situation caused by left to right sampling order.

Figure 9.3.: **Critical inside and outside values.** Figures illustrate particular situations that generally yield inconsistencies in the context of inside-out sampling processes based on the SCFG $\mathcal{G}_s$.

to a particular production rule there implies that the considered *unfolded* sequence fragment is folded to *one of the possible* substructures that can be generated from the conclusion.

Furthermore, note that if substructures are sampled in arbitrary order, there are different situations depending on the previously folded fragments. First, consider the inside probability $\widehat{\alpha}_T(jj+1,n)$ which includes to probability of any substructure that can be derived from symbol T on $R_{jj+1,n}$. It hence provides a proper contribution if and only if no substructure has been folded so far on the entire fragment $R_{jj+1,n}$. Otherwise, if one or more (paired) substructures on this sequence fragment have already been sampled, then $\widehat{\alpha}_T(jj+1,n)$ represents an incorrect contribution, as it includes too many possibilities. This situation is illustrated in Figure 9.3b. In fact, the probabilities of any folding on $R_{jj+1,n}$ that can not be derived from T under the restriction that the existing substructures must be folded have to be excluded now. This, however, could only easily and efficiently be done in the rare case where the *complete* fragment $R_{jj+1,n}$ has already been folded, since then we could simply compute the probability that T generates the structure $S_{jj+1,n}$ and use it instead of $\widehat{\alpha}_T(jj+1,n)$ for calculating the corresponding sampling probability.

Finally, it should be clear that using the traditional outside probability $\widehat{\beta}_T(i,n)$ implies similar inconsistencies in the corresponding sampling distributions, as intuitively it provides the probability of every possible folding of the exterior loop on the sequence fragment $R_{1,i-1}$ preceding the nonterminal symbol T in the corresponding derivation tree. Accordingly, if that fragment is only partially but not yet completely folded due to the arbitrary sampling order, then $\widehat{\beta}_T(i,n)$ incorrectly contains probabilities for structures that can not be sampled in this iteration (on this path in the decision tree) according to the previously folded elements. Note that principally, similar inconsistencies in the used outside values can also occur in case of the common sampling strategy, but there it does not matter due to the fact that in all conditional sampling distributions, the outside values are consistently cancelled out.

Remark 9.4.1 *Intuitively, as regards inside-to-outside strategies, one might assume that these problems, especially the one illustrated in Figure 9.3b, could be reduced by sampling adjacent substructures from left to right*[13]*. This means after one of the possible hairpin loops $S_{i,j}$ on a considered sequence fragment $R_{start,end}$ has been sampled and afterwards has been validly extended to a paired substructure $S_{ii,jj}$, the then still unfolded fragment $R_{start,ii-1}$ will not be folded by the corresponding strategy in subsequent sampling steps, but it will immediately be saved as unpaired region. This more restrictive sampling variant obviously stronger resembles the outside-in statistical sampling algorithm applied in the previous chapters, but nevertheless the reverse sampling direction again implies fundamental problems. Specifically, in order to ensure proper sampling distributions for drawing a random hairpin loop $S_{i,j}$ on fragment $R_{start,end}$, the sampling probabilities for all possible choices must of course still depend on the inside values $\widehat{\alpha}_F(i,j)$. In contrast to the definition of* $\mathrm{pcHL}(start,end,helices,unpRegs)$*, however, they may not depend on the traditional outside values $\widehat{\beta}_L(i,j)$, as the fragment $R_{start,ii-1}$ preceding the final paired substructure $S_{ii,jj}$ that contains the hairpin loop $S_{i,j}$ is predetermined to become single-stranded. This situation is illustrated in Figure 9.3c. Presumably, computing such adapted outside values – either dynamically during the sampling process or in a corresponding preprocessing step – is a challenging if not unsolvable task, at least without increasing the time and/or space complexity.*

In summary, even in connection with exact preprocessing, the inside-out sampling direction – contradicting the top-down structure generation according to the SCFG model – in many cases inherently implies the consideration of fuzzy sampling distributions. This, together with the fact that in general a specific subset of the ensemble of all feasible structures can not be sampled at all, yields the conclusion that the dynamic sampling method introduced in this section is only a sampling heuristic, but this is indeed unavoidable under the assumption of the induced bottom-up structure generation. In the context of approximative preprocessing,

[13] Or more generally, by sampling substructures in the same order as they are generated by the underlying grammar, that is alternatively from right to left.

the heuristic character of our strategy, mainly due to the use of often inconsistent inside and outside values for deriving the respective sampling distributions, is not necessarily of importance since the considered inside and outside probabilities and resulting distributions are anyhow fuzzy. Nevertheless, when attempting to employ this alternative strategy on the basis of precise inside and outside values, one has to keep in mind that the distribution of sampled candidate structures can not exactly correspond to the ensemble distribution induced by the SCFG model.

9.5. Analysis of the Effect of Approximative Preprocessing

The proclaimed goal of this section is to quantify the decline in sampling quality that results from applying different stages (by means of parameter W_e) of the approximative preprocessing variant proposed in Section 9.3 rather than the exact one (as introduced in Section 7.3.1, corresponding to the choice of $W_e = \infty$). To achieve this, we will perform a comprehensive experimental analysis on the effect of increasing the approximation level on the quality of results obtained by (L)SCFG based statistical sampling. Particularly, we want to explore to what extend the quality of produced secondary structure samples and the corresponding predictive accuracy decreases when different degrees of approximation are used for deriving the needed sampling probabilities considered by the conventional strategy as presented in Section 8.3.2 and by the contrary dynamic variant described in Section 9.4, respectively.

9.5.1. RNA Structure Data

Aiming at all-embracing examinations, we decided to consider particular instances of two opponent cases of RNA data that may be chosen for parameter estimation (see Section 3.3.6.4 for details): First, two different (rich and reliable) training sets where only structures of a single biological class are contained, respectively. In particular, these are exactly the same tRNA database and the identical 5S rRNA data set that were considered in the preceding chapters. Second, a training set of mixed biological classes, here represented by the (comparably sparse) mixed structural S-151Rfam database.

Hence, the corresponding three different induced (L)SCFG models – obtained by training the sophisticated grammar $\mathcal{G}_s$ on all structures contained in the respective training set – will be used as bases for our diverse (heuristic) sampling methods. Recall that in Sections 6.6 and 7.4, the quality of the respective induced (L)SCFG models has been evaluated from different perspectives. Amongst others, we found that overfitting is not really an issue in connection with the underlying grammar $\mathcal{G}_s$ and the training sets used (at least for the rich tRNA and 5S rRNA data), even in the case of length-dependent parameter estimation procedures. Furthermore, by a series of k-fold cross-validation experiments[14], we showed that for the tRNA and 5S rRNA sets, the (L)SCFG based sampling approach yields good quality results with respect to all considered applications of the generated sample sets if no uncertainties are included in the respective sampling distributions for a given sequence (that is, in case of exact preprocessing according to $W_e = \infty$). However, for the lean S-151Rfam set, worse results are observed, especially in the length-dependent case.

Anyway, our declared aim is to investigate the extend to which particular choices of (constant) W_e to be considered for our heuristic sampling methods lower the resulting sampling accuracy (compared to the exact variant), which does actually not depend on the quality of the underlying (L)SCFG model. Thus, it is not necessary to perform comprehensive k-fold cross-validation

[14]It should be mentioned that for any data set, similar results have been observed for all of the k considered randomly chosen folds, respectively, providing further evidence on the reliability of the estimated parameters and especially of the overall data set.

procedures based on partitions of our three data sets of RNA structures into k approximately equal-sized folds, respectively. In fact, for our purpose it obviously suffices to use one particular randomly chosen subset (benchmark set) of any training set for deriving reliable evaluation results, where the corresponding employed grammar parameters can readily be estimated from the corresponding complete training set, together with adequate choices for the restrictive parameters $max_{hairpin}$, max_{bulge} and max_{strand} considered for the dynamic sampling strategy. In fact, by considering all structures in our three databases that obey to the commonly considered structural parameters $min_{HL} = 3$ and $min_{hel} = 2$, respectively, we found

- $max^*_{hairpin} = 15$, $max^*_{bulge} = 26$ and $max^*_{strand} = 23$ in case of tRNAs,
- $max^*_{hairpin} = 15$, $max^*_{bulge} = 7$ and $max^*_{strand} = 6$ for 5S rRNAs and
- $max^*_{hairpin} = 73$, $max^*_{bulge} = 31$ and $max^*_{strand} = 181$ for the S-151Rfam set.

Note that this somehow indicates that the quality of the S-151Rfam set of mixed structural RNAs is not as high as that of the trusted tRNA and 5S rRNA sets.

Nevertheless, for our purpose, we decided to consider random (slightly less than) 5% portions of the comprehensive tRNA and 5Sr RNA data sets and a random 10% portion of the S-151Rfam database as benchmarks. Particularly,

- our tRNA benchmark consists of 100 distinct sequences (with lengths in $[70, 90]$ and about 76 on average),
- our 5S rRNA benchmark contains 50 distinct elements (with lengths in $[104, 122]$ and about 118 on average), and finally
- our S-151Rfam benchmark set is given by 15 distinct RNA molecules (with lengths in $[76, 145]$ and an average length of about 100).

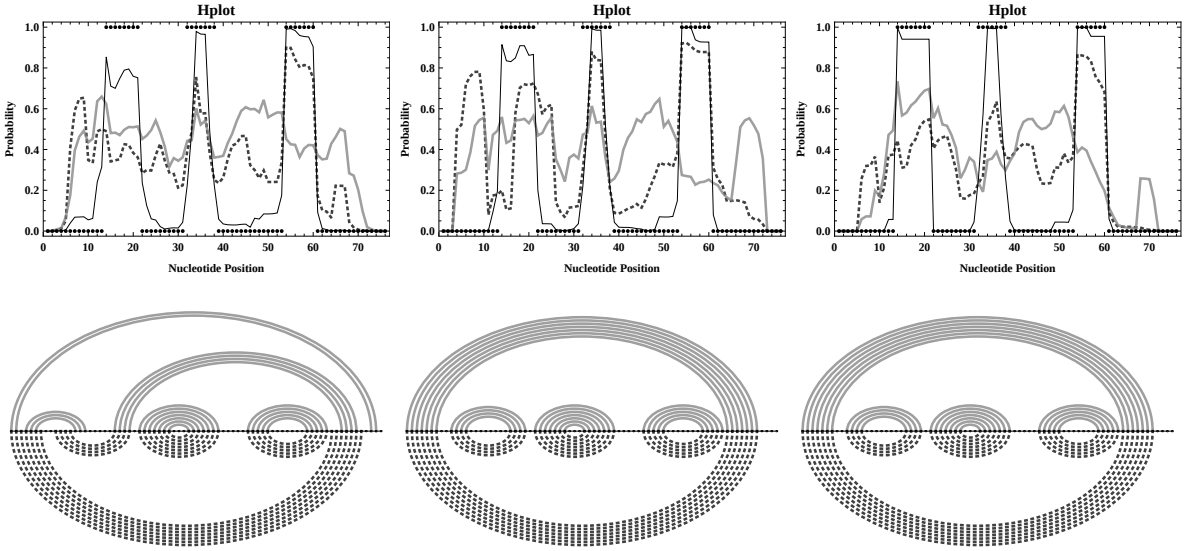
Essentially, for any (L)SCFG model implied by one of our three training sets, we will indeed consider the respective complete benchmark set as foundation for the desired examinations on the negative impact of decreasing values of W_e on the resulting accuracy of statistical sampling. The observed folding results can also readily be opposed to corresponding ones obtained by employing several popular RNA secondary structure prediction tools (that implement competing approaches for deriving a particular folding). This will be done in Section 9.6.

However, in order to get a first impression on the influence of approximative preprocessing variants on the quality of generated sample sets, we decided to initially study probability profiling results for a single RNA sequence in the following section, before we will continue with examinations of complete benchmarking results for the different data sets.

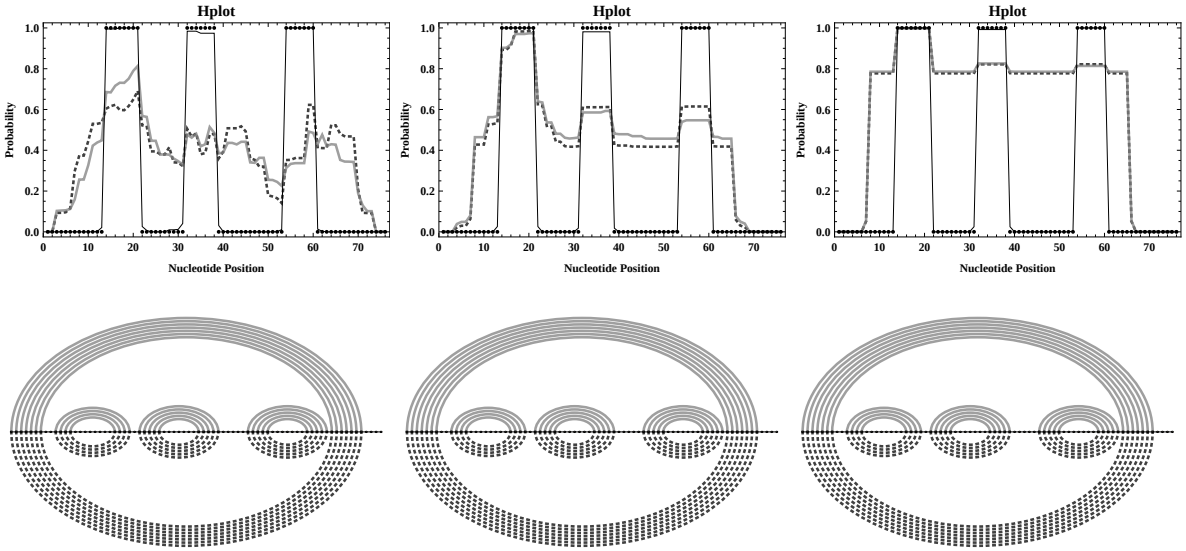
9.5.2. Probability Profiling for Specific Loop Types

In analogy to the preceding chapters, we decided to consider the well-known *Escherichia coli* $tRNA^{Ala}$ sequence for probability profiling of unpaired bases located within particular types of loops. Since this molecule folds into the typical tRNA cloverleaf structure, this makes it very easy to judge the accuracy of the resulting profiles (and predictions) and hence builds a well-suited starting point for our analysis.

In principle, we calculated a collection of probability profiles from different statistical samples for the considered sequence, where any sample has been derived by applying one of our two sampling strategies (as devised in Sections 8.3.2 and 9.4) on the basis of a particular set of inside and outside probabilities (as defined in Section 9.3.4), computed for a specific reasonable value of W_e. All relevant results are displayed in Figures E.1 to E.8 of Section E. Some of the most interesting ones are presented in Figures 9.4 and 9.5.

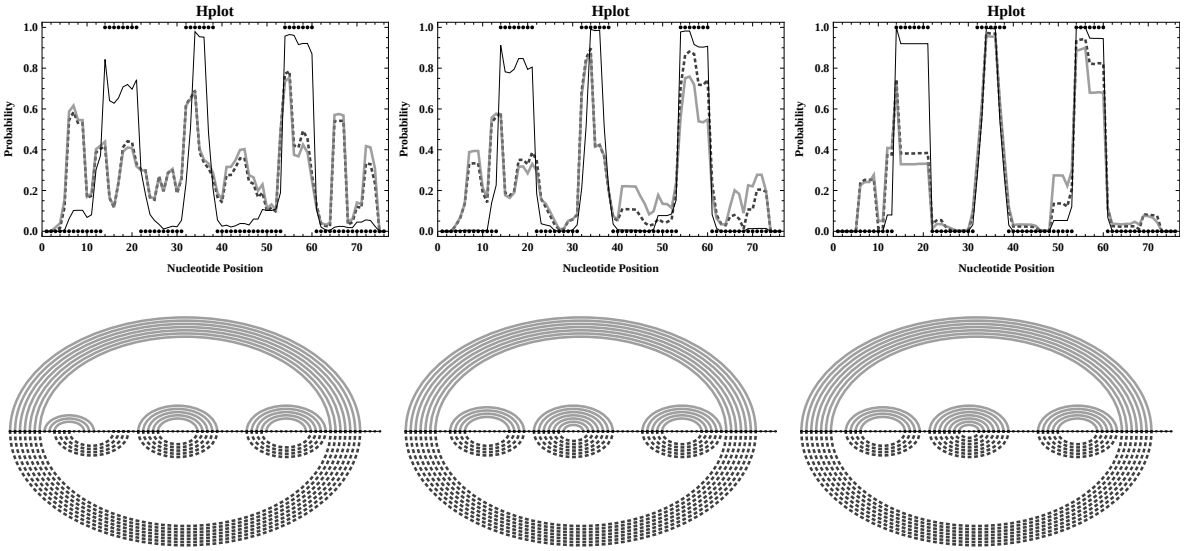


(a) Results for traditional SCFG model.

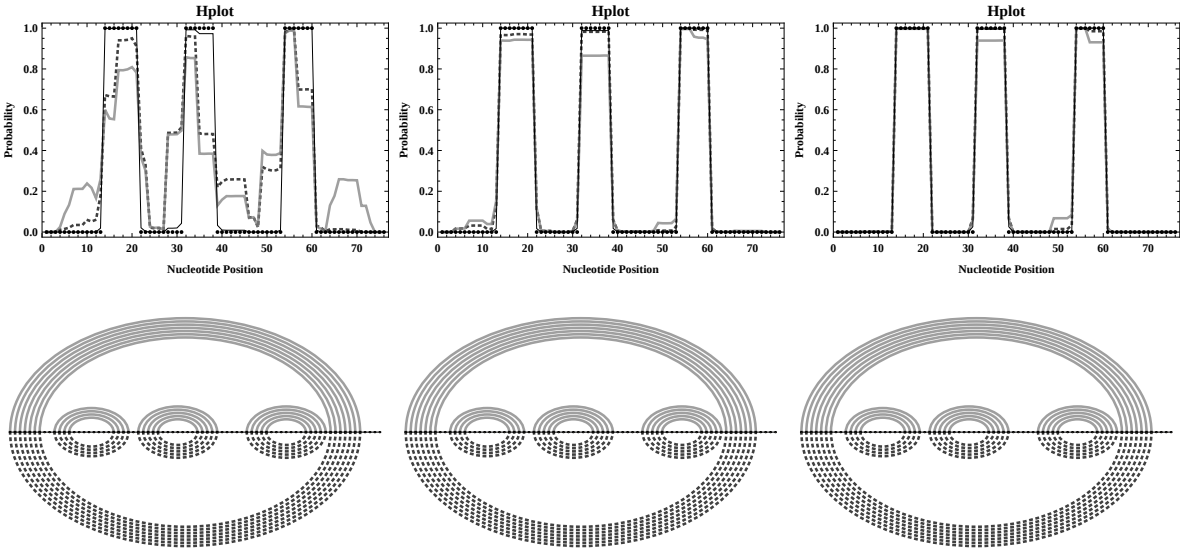


(b) Results for LSCFG model.

Figure 9.4.: **Profiling results obtained with approximated probabilities and common strategy.** Figures show hairpin loop profiles and MP predictions for *E.coli* $\mathrm{tRNA}^{\mathrm{Ala}}$, calculated from random samples of 100 000, 10 000 and 1 000 structures generated with the common sampling strategy for $W_e = -1$ (no window, thick gray lines), $W_e = 30$ (moderate window, thick dotted darker gray lines) and $W_e = +\infty$ (complete window, thin black lines), respectively, under the assumption of $\min_{\mathrm{HL}} = 3$ and $\min_{\mathrm{hel}} \in \{2, 3, 4\}$ (figures from left to right). The correct hairpin loops in *E.coli* $\mathrm{tRNA}^{\mathrm{Ala}}$ are illustrated by the black points.



(a) Results for traditional SCFG model.



(b) Results for LSCFG model.

Figure 9.5.: **Profiling results obtained with approximated probabilities and dynamic strategy.** Figures show hairpin loop profiles and MP predictions corresponding to those presented in Figure 9.4, but obtained by employing the alternative sampling strategy (under the assumption of the common restrictions $\max_{\mathtt{hairpin}} = \max_{\mathtt{bulge}} = \max_{\mathtt{strand}} = 30$).

9.5.2.1. General Observations

First, the sampling results displayed in Figure 9.4 indicate that for the common sampling strategy, considering a window of constant size W_e (chosen to cover the size of hairpin loops), that is using a mixed preprocessing variant, actually yields a slight improvement of the resulting sampling quality, where the same time requirements are needed for generating the respective sample sets.

Contrary to this observation, Figure 9.5 demonstrates that when employing our alternative sampling strategy, the corresponding results are not significantly different for the completely approximate preprocessing variant and for a mixed version on the basis of a constant value for W_e. Thus, to our surprise it does not matter if we consider a constant window for exact calculations or simply approximate all inside and outside values. This is not only an interesting observation itself, but also fortunately prevents us from having to deal with an undesirable trade-off between reducing the worst-case time complexity and sacrificing less of the resulting sampling quality. In fact, this means we may without resulting significant quality losses always use the more efficient completely approximative preprocessing variant in order to reduce the worst-case time complexity of the overall sampling algorithm.

Furthermore, comparing the profile plots presented in Figure 9.4 to those displayed in Figure 9.5, we find that in case of heuristic preprocessing variants, basically all samples generated by the dynamic strategy are significantly more accurate than the corresponding ones produced with the common sampling method. Hence, as intended, the inside-to-outside course of action of our alternative sampling strategy seems to indeed handle uncertainties in the precalculated inside and outside values for a given input sequence better than the reverse outside-to-inside sampling process employed by the well-established strategy. Moreover, when using the more favorable dynamic sampling strategy in connection with any of our heuristic methods, then the consideration of more realistic (yet more restrictive) longer minimum allowed helix lengths (that is, increasing values of $\min_{\text{hel}}$ up to a certain constant value) might results in a noticeable quality improvement of the respective sample sets for both the traditional and the length-dependent sampling variant.

Altogether, the presented profiles (at least those for the common choice of $\min_{\text{hel}} = 2$) perfectly demonstrate that due to the noisy ensemble distribution caused by approximating the highly relevant sequence-dependent emission probabilities, the resulting sample sets are in many cases of humble overall quality, since they contain many foldings that are rather unlikely according to the exact ensemble distribution for the considered input sequence, especially when the common sampling strategy is used to generate them. Thus, statistical samples produced on the basis of our heuristic method might generally rather not be used for estimating the probabilities of particular structural motifs by profiling. Nevertheless, by identifying the MP structure of a particular random sample, we can still get high quality prediction results, especially under the assumption of the length-dependent grammar model.

9.5.2.2. Relevant Sampling Probabilities

In order to affirm the intuitive conclusions and assumptions resulting from the presented unquantified profile plots and corresponding predictions, as well as to get more detailed insight into the sources of problems that arise by switching from exact inside-outside calculations to heuristic variants, we decided to study an important aspect concerning the appropriateness of approximated values. Particularly, as already pointed out in Section 8.4, in order to prevent a decline in accuracy of generated structures and a reduction of the overall sampling quality, it seems to be of great importance that the employed sampling strategy is capable of distinguishing between inside (and outside) values and especially sampling probabilities that

are equal and unequal to zero according to the exact ensemble distribution for the given input sequence.

Motivated by this suggestion, we decided to count the relevant (that is, greater than zero) inside, if needed also outside, and sampling probabilities that were considered for obtaining the profiles presented in Figures 9.4 and 9.5 (on the left, that is for $\min_{\text{hel}} = 2$, respectively). The results are collected in Tables E.1 to E.5 of Section E.

Basically, the tabulated values yield the observation that in most cases, the relevant inside, outside and most importantly sampling probabilities obtained from exact preprocessing (using $W_e = \infty$) coincide with the respective relevant ones when applying any of the investigated heuristic methods (for $W_e \in \{-1, 30\}$). In particular, we observe two important relations: On the one hand, for both strategies, some sampling probabilities are only relevant in the case of approximative calculations. That is, these probabilities are not relevant if they are derived on the basis of corresponding exact inside (and outside) values for the input sequence, but they actually become relevant if the respective inside (and outside) probabilities are only approximated. This means that when applying our heuristic method, both sampling strategies are at some points allowed to fold substructures that are principally impossible according to the actual input sequence. This inevitably results in low quality, otherwise prohibited, overall candidate structures that indeed have probability 0 according to the exact ensemble distribution.

On the other hand, a moderate number of outside probabilities as considered by the dynamic strategy are only relevant in the exact case. This means they unfortunately became irrelevant due to the heuristic inside outside calculations. As a consequence, it must be assumed that at least a few of the sampling probabilities considered by this alternative strategy, namely the ones that depend on these incorrectly irrelevant outside values, are also no longer relevant in connection with the respective heuristic variant[15]. However, under the condition that the number of falsely irrelevant sampling probabilities resulting from heuristic preprocessing is rather small, the employed strategy is only in some extremely rare cases prevented from folding principally possible substructures; notably, this seems to be satisfied in connection with both considered sampling strategies. This eventually means that the employed strategy should have the chance to randomly sample (at least one of) the most likely structures for the considered input sequence according to the exact distribution on all feasible foldings.

All these observations exemplarily prove the intuitive assumption already formulated at the end of Section 9.3.5, namely that potentially significantly larger numbers of candidate foldings need to be sampled when applying our heuristic methods rather than the exact algorithm for ensuring that the MP structure of the generated sample is identical to one of the most likely foldings of the input sequence and hence represents a high quality prediction. A corresponding investigation on reasonable sample sizes depending on parameter W_e will follow in the next section. Nevertheless, it should already at this point be noted that due to the results listed in Tables E.1 to E.5 of Section E, it must be assumed that the consideration of rather large sample sizes is especially required in case of complete approximation, that is when using $W_e = -1$ for the preprocessing step, or when structures are sampled according to the alternative strategy introduced in Section 9.4.

[15] We could not collect and compare the actual counts of all relevant sampling probabilities (corresponding to valid choices for substructures) that need to be considered for any particular sampling decision due to the fact that these counts (cardinalities of the respective sets of possible choices as defined in Section 9.4.2) depend on the previously folded substructures which are obviously not known in general but only during one particular overall sampling process. In fact, only the number of relevant sampling probabilities for the initial sampling decision (that is, for randomly drawing the first hairpin loop if no substructures have been folded yet) does not need to be dynamically evaluated during a particular sampling process, and could hence easily be derived under the assumption of different values of $\min_{\text{hel}}$, see Table E.5.

9.5.3. Reproducibility of Predictions in Connection with Sample Size

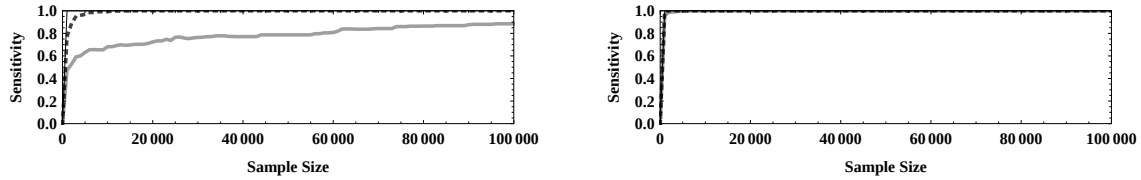
The aim of this section is to investigate to what extend the stability in MP structure predictions and the resulting accuracy depends on the sample size considered when applying different variants of our statistical sampling method.

Obviously, one straightforward approach for reaching this goal is to consider (one or more) relevant measures for assessing the predictive accuracy and to plot the corresponding values obtained from MP structures of randomly generated sample sets as functions in the considered sample sizes. For any considered measure, a corresponding plot in the sample size can easily be derived by continuously sampling secondary structures (up to a preliminary chosen reasonable number) and always taking the so far most probable one as the actual prediction in order to determine the corresponding measure value for the currently reached sample size. To obtain reliable results, the measured values for any considered sample size should be averaged over an appropriate number of self-contained experiments, for instance over a sufficiently large number of independent runs for the same input (if only a single sequence is considered), or over the corresponding values obtained for an adequate number of distinct inputs (if a particular benchmark set of RNA sequences is available).

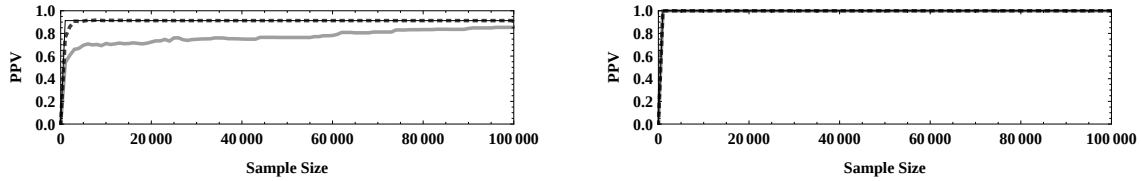
Figure 9.6 shows some averaged sensitivities and PPVs obtained from 50 independent runs for input sequence *E.coli* tRNA^{Ala}. However, more reliable results derived on the basis of our different benchmark sets, respectively, are displayed in Figures 9.7 to 9.12. Note that besides the most significant choices of $W_e \in \{-1, \infty\}$, we also consider a constant value of 30 for the W_e parameter, respectively, in order to facilitate comparison. However, it would also be reasonable to consider a specific constant value W_e for any benchmark set, chosen in accordance with the corresponding used value of $\max_{\texttt{hairpin}}$ for the dynamic sampling strategy. This means that the considered constant W_e value for any benchmark set alternatively could have been chosen conform to the $\max^*_{\texttt{hairpin}}$ parameter observed from training the respective database (as described in Section 9.4.1). This would imply $W_e = 15$ to be considered for tRNAs and 5S rRNAs, but $W_e = 30$ in connection with the S-151Rfam set (see Section 9.5.1), which we found less attractive with respect to comparing the results obtained on the basis of different data. Nevertheless, corresponding results have also been derived under the assumption of $W_e = 15$ for tRNAs and 5S rRNAs, but they are rather similar to those presented in Figures 9.7 to 9.10. Therefore, we decided not to present them in detail in this section, but they will indeed be additionally considered for the examinations to be performed in the preceding sections.

Anyway, let us first consider Figure 9.6 derived on the basis of a single exemplary tRNA sequence. We observe that under the assumption of the LSCFG model, the resulting sensitivity and PPV values are principally not affected by approximative preprocessing. Contrarily, when making use of approximate probabilities in connection with the traditional SCFG model, sample sizes about 40 to 50 times as large as for a precise preprocessing are needed to generate competitive predictions with the common sampling strategy. Notably, our alternative variant seems to require the consideration of even larger sample sizes in order to ensure comparable accuracies. This might be due to the heuristic character of this sampling strategy, as discussed in Section 9.4.3, which inevitably increases the number of inconsistencies in the considered sampling distributions.

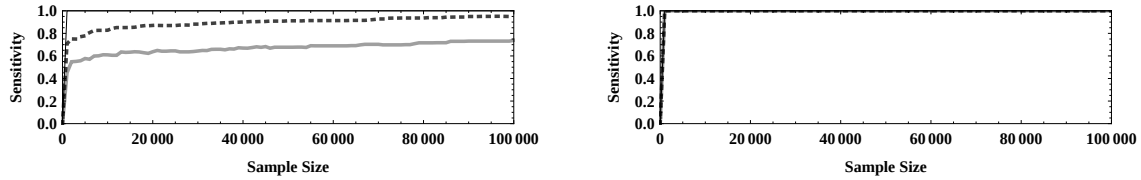
The results presented in Figures 9.7 and 9.8 show that using our heuristic preprocessing variants, the expected sensitivity and PPV values for our complete tRNA benchmark set are not significantly less accurate than those obtained on the basis of exact preprocessing. Notably, already for rather small sample sizes, the expected accuracies are competitive and the corresponding variances are extremely low. This suggests that when considering larger samples, at least for short low invariant types of RNAs, no accuracy is sacrificed in connection with approximated sampling probabilities.



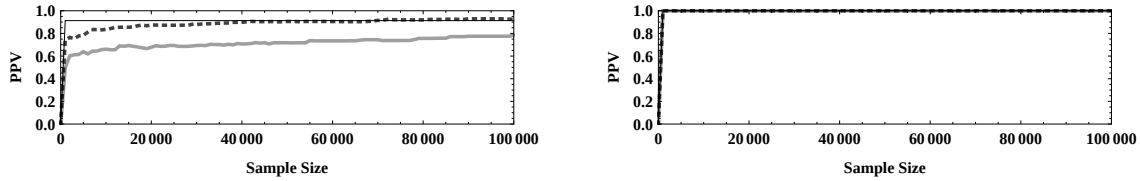
(a) Sensitivity results obtained with common sampling strategy.



(b) PPV results obtained with common sampling strategy.

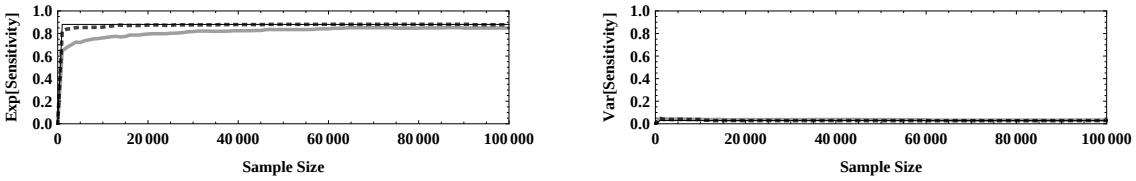


(c) Sensitivity results obtained with alternative sampling strategy.

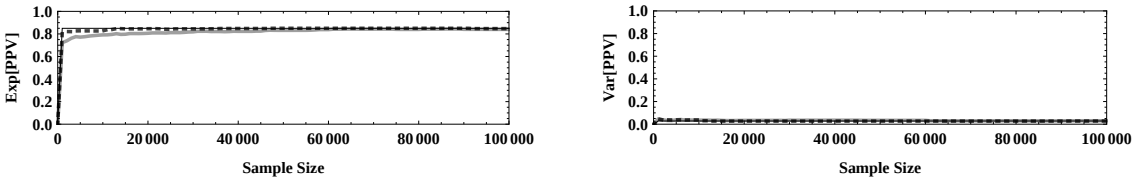


(d) PPV results obtained with alternative sampling strategy.

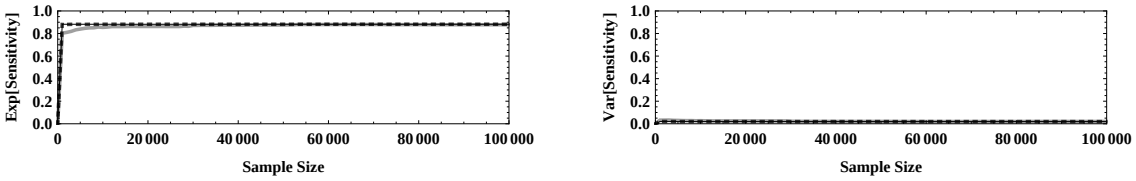
Figure 9.6.: **Accuracies obtained with different sample sizes for a single tRNA sequence.** Figures show averaged accuracy measures for MP predictions as functions of sample size, derived on the basis of 50 independent runs for *E.coli* tRNA^{Ala} by considering $W_e = -1$ (no window, thick gray lines), $W_e = 30$ (moderate window, thick dotted darker gray lines) and $W_e = +\infty$ (complete window, thin black lines), respectively, under the assumption of the traditional SCFG model (figures on the left) and the LSCFG model (figures on the right), with structural constraints $\min_{\text{HL}} = 3$ and $\min_{\text{hel}} = 2$. For our alternative strategy, we used restrictions $\max_{\texttt{hairpin}} = 15$, $\max_{\texttt{bulge}} = 26$ and $\max_{\texttt{strand}} = 23$, as observed from training.



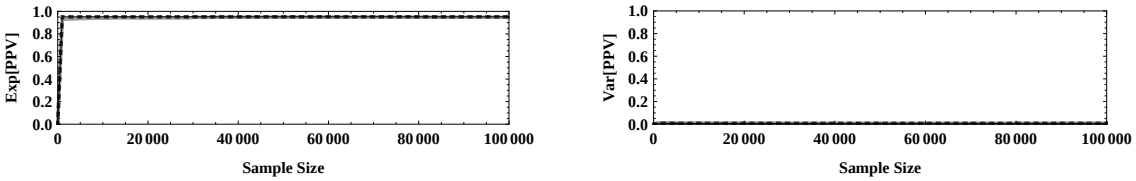
(a) Sensitivity results for traditional SCFG model.



(b) PPV results for traditional SCFG model.

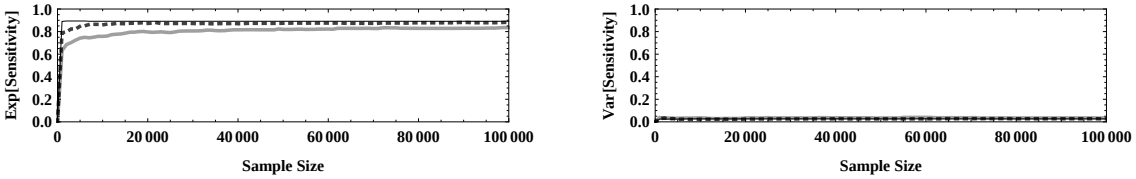


(c) Sensitivity results for LSCFG model.

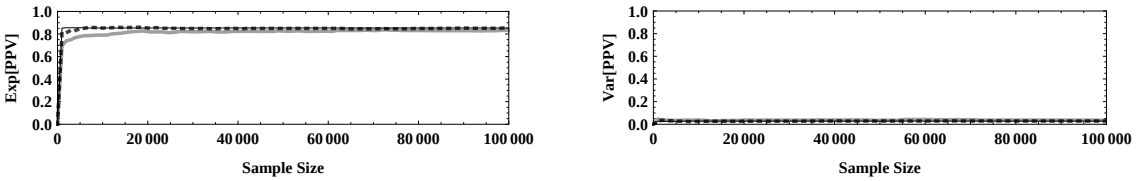


(d) PPV results for LSCFG model.

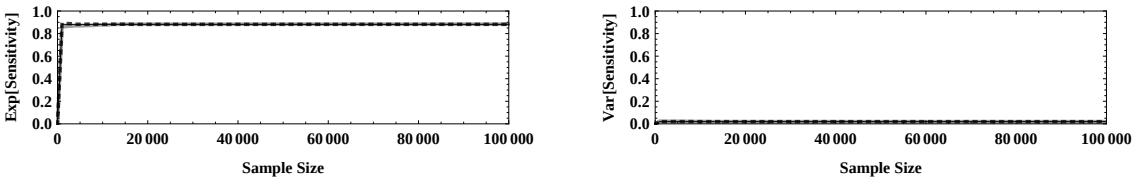
Figure 9.7.: **Prediction results for tRNA benchmark, obtained with the common strategy.** Figures show averaged accuracy measures for MP predictions as functions of sample size (left) and corresponding variances (right), derived on the basis of our tRNA benchmark set by employing the common sampling strategy for $W_e = -1$ (no window, thick gray lines), $W_e = 30$ (moderate window, thick dotted darker gray lines) and $W_e = +\infty$ (complete window, thin black lines), respectively, under the assumption of $\min_{HL} = 3$ and $\min_{hel} = 2$.



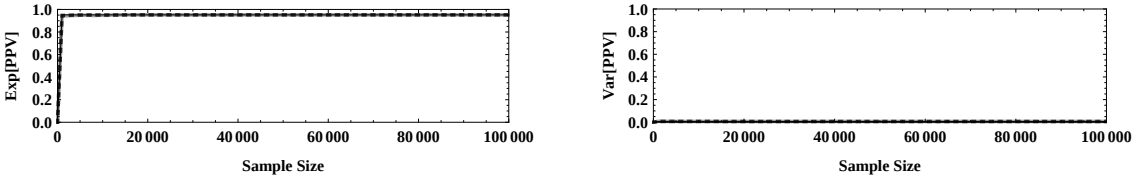
(a) Sensitivity results for traditional SCFG model.



(b) PPV results for traditional SCFG model.

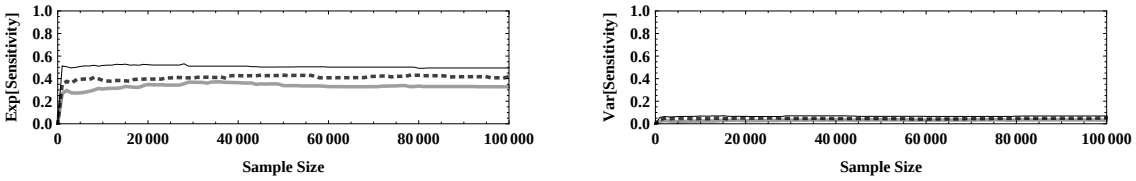


(c) Sensitivity results for LSCFG model.

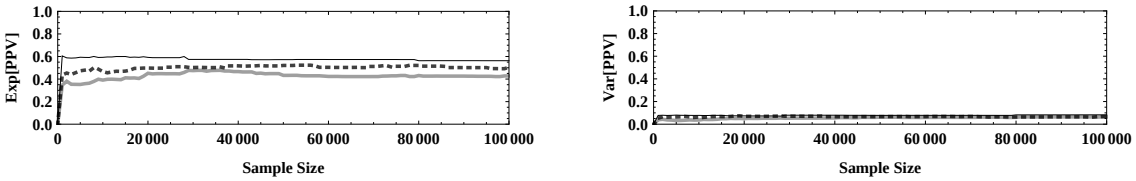


(d) PPV results for LSCFG model.

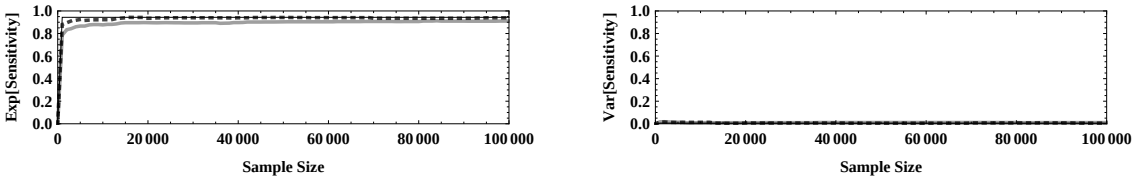
Figure 9.8.: **Prediction results for tRNA benchmark, obtained with the dynamic strategy.** Figures show averaged accuracy measures for MP predictions as functions of sample size (left) and corresponding variances (right) corresponding to those of Figure 9.7, obtained from our tRNA benchmark set by employing the dynamic sampling strategy (with restrictions $\max_{\mathtt{hairpin}} = 15$, $\max_{\mathtt{bulge}} = 26$ and $\max_{\mathtt{strand}} = 23$ observed from training data).



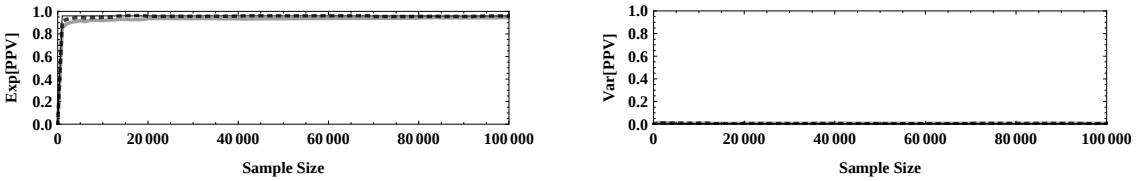
(a) Sensitivity results for traditional SCFG model.



(b) PPV results for traditional SCFG model.

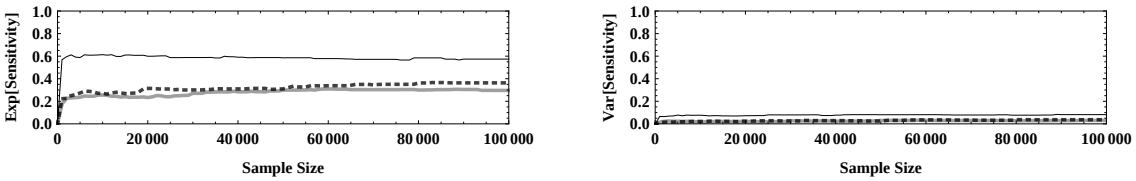


(c) Sensitivity results for LSCFG model.

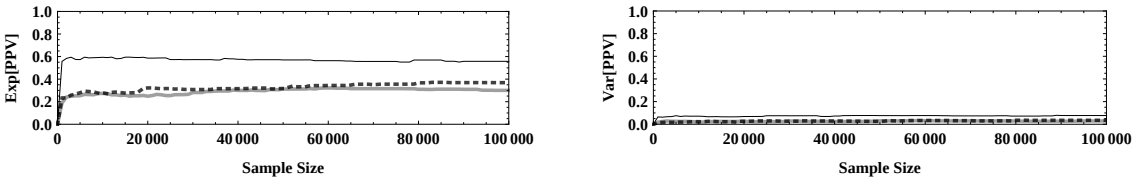


(d) PPV results for LSCFG model.

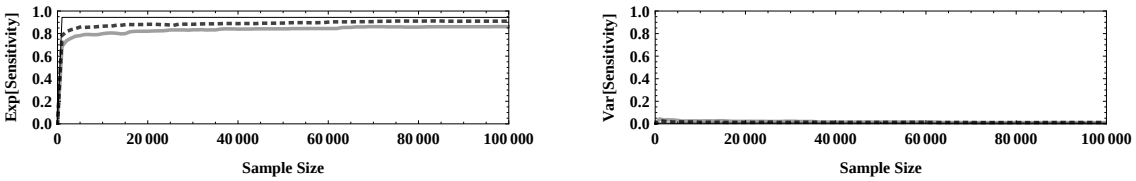
Figure 9.9.: **Prediction results for 5S rRNA benchmark, obtained with the common strategy.** Figures show averaged accuracy measures for MP predictions as functions of sample size (left) and corresponding variances (right), derived on the basis of our 5S rRNA benchmark set by employing the common sampling strategy for $W_e = -1$ (no window, thick gray lines), $W_e = 30$ (moderate window, thick dotted darker gray lines) and $W_e = +\infty$ (complete window, thin black lines), respectively, under the assumption of $\min_{HL} = 3$ and $\min_{hel} = 2$.



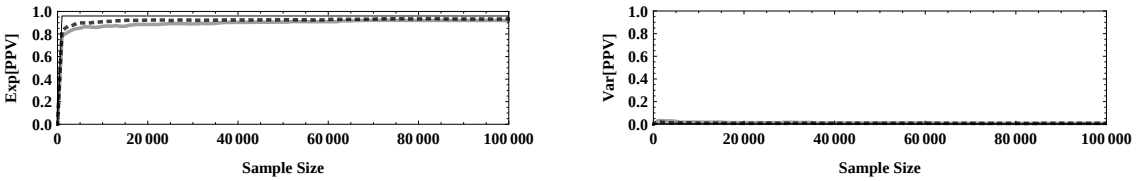
(a) Sensitivity results for traditional SCFG model.



(b) PPV results for traditional SCFG model.

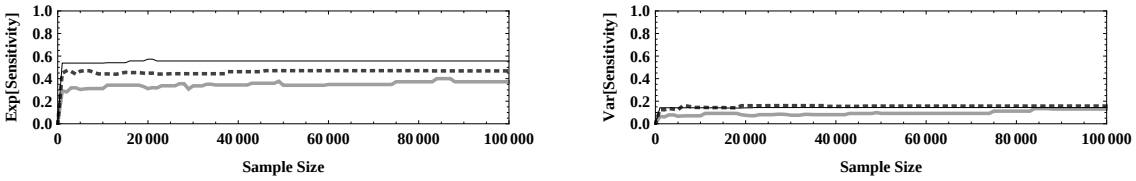


(c) Sensitivity results for LSCFG model.

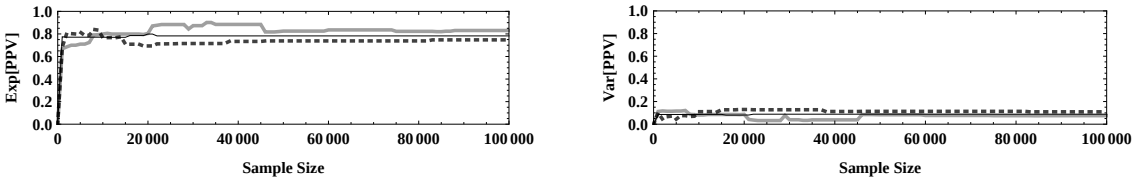


(d) PPV results for LSCFG model.

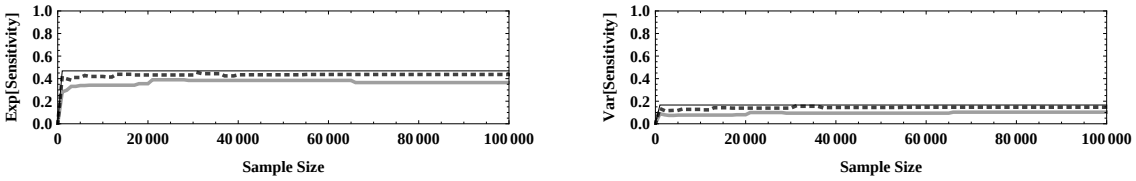
Figure 9.10.: **Prediction results for 5S rRNA benchmark, obtained with the dynamic strategy.** Figures show averaged accuracy measures for MP predictions as functions of sample size (left) and corresponding variances (right) corresponding to those of Figure 9.9, obtained from our 5S rRNA benchmark set by employing the dynamic sampling strategy (with restrictions $\max_{\mathtt{hairpin}} = 15$, $\max_{\mathtt{bulge}} = 7$ and $\max_{\mathtt{strand}} = 6$ observed from training data).



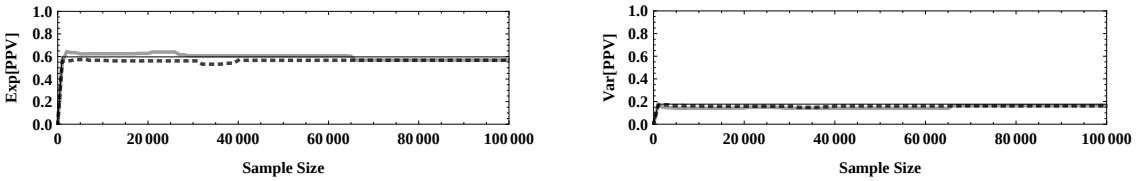
(a) Sensitivity results for traditional SCFG model.



(b) PPV results for traditional SCFG model.

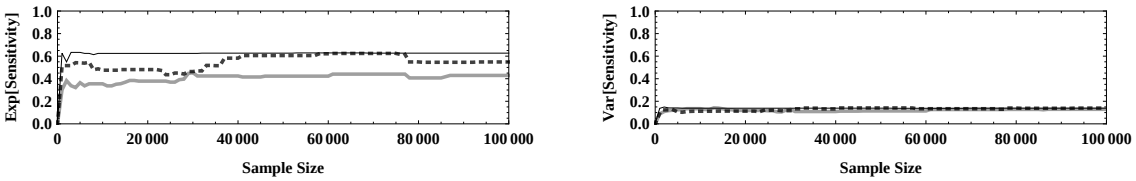


(c) Sensitivity results for LSCFG model.

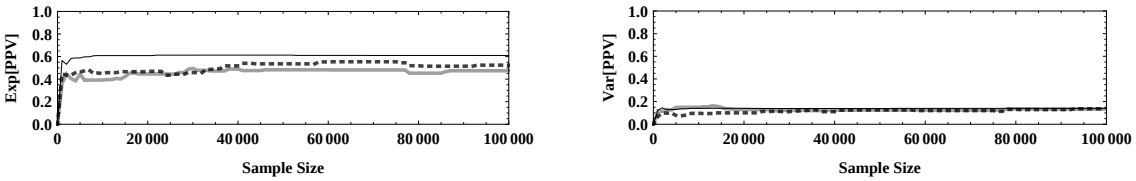


(d) PPV results for LSCFG model.

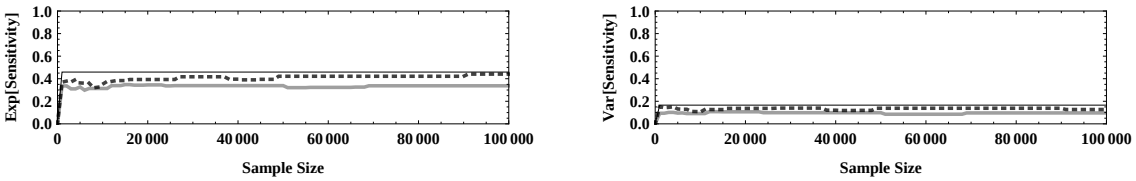
Figure 9.11.: **Prediction results for S-151Rfam benchmark, obtained with the common strategy.** Figures show averaged accuracy measures for MP predictions as functions of sample size (left) and corresponding variances (right), derived on the basis of our S-151Rfam benchmark set by employing the common sampling strategy for $W_e = -1$ (no window, thick gray lines), $W_e = 30$ (moderate window, thick dotted darker gray lines) and $W_e = +\infty$ (complete window, thin black lines), respectively, under the assumption of $\min_{HL} = 3$ and $\min_{hel} = 2$.



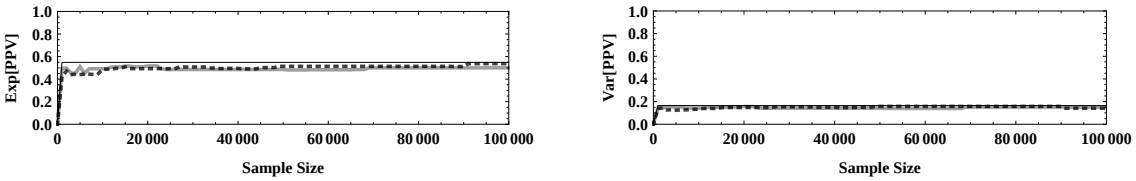
(a) Sensitivity results for traditional SCFG model.



(b) PPV results for traditional SCFG model.



(c) Sensitivity results for LSCFG model.



(d) PPV results for LSCFG model.

Figure 9.12.: **Prediction results for S-151Rfam benchmark, obtained with the dynamic strategy.** Figures show averaged accuracy measures for MP predictions as functions of sample size (left) and corresponding variances (right) corresponding to those of Figure 9.11, obtained from our S-151Rfam benchmark set by employing the dynamic sampling strategy (assuming the common restrictions $\max_{\mathtt{hairpin}} = \max_{\mathtt{bulge}} = \max_{\mathtt{strand}} = 30$).

Nevertheless, the corresponding plots for 5S rRNAs as collected in Figures 9.9 and 9.10 draw a different picture. We observe a decline in the averaged sensitivity and PPV values obtained on the basis of approximate probabilities compared to the corresponding results that can be reached with exact probabilities, especially when considering the traditional SCFG model.

Anyway, the results for our benchmark of the S-151Rfam data set as shown in Figures 9.11 and 9.12 are more interesting, and indeed partially quite surprising. First, we find that the averaged sensitivities get smaller with increasing level of approximation, or else get larger with growing window size W_e, which is totally expected. However, there is one interesting observation as regards these results, namely that when employing our dynamic sampling variant, the sensitivity reached with constant W_e is for a particular range of considered sample sizes almost equal to that obtained in the exact case (see Figure 9.12a). This perfectly matches our intention that this reverse sampling strategy might be more adequate in connection with mixed preprocessing variants. In principle, the expected PPVs obtained with approximated probabilities are closer to the corresponding values measured in cases of exact preprocessing than the averaged sensitivities. Moreover, the differences between mixed heuristic variants (for constant window sizes) and complete approximation are less in terms of measured PPVs than in terms of the corresponding sensitivity values. Anyway, the most surprising observation in this context is that the resulting PPVs are not for all considered sample sizes ordered as naturally expected, especially when using the common sampling strategy for obtaining the corresponding predictions. Since in that case, the most grave approximation variant (using $W_e = -1$ and thus no window at all) seems to yield the best results, primarily in the length-independent case (see Figure 9.11b). Last but not least, we observe that for the considered benchmark of the S-151Rfam set, the variances of both the sensitivities and PPVs are much greater than the corresponding variances for tRNAs and 5S rRNAs, which must be due to the greater structural diversity.

In summary, the presented results perfectly demonstrate that a stability in resulting predictions and a competitive prediction accuracy can only be reached by increasing the sample size. That is, more candidate structures ought to be generated for guaranteeing that the resulting MP predictions are reproducible (by independent runs for the same input sequence) and also of high quality. As expected, this negative effect might in most cases be considerably lowered by using (larger) constant values of $W_e \geqslant 0$. Anyway, a sample size of only 1000 structures as typically used for statistical sampling approaches [DL03, CLD05] might generally not suffice in connection with our heuristic methods, as indicated by the presented experiments.

Thus, for a naive implementation the speedup gained by approximation may partly be lost. However, unlike prediction algorithms using DP, sampling can easily be parallelized. Making use of a grid environment where today one may assume a processor to have about 8 cores, a grid of size 5 or 6 computers is sufficient to compensate the increased sample size. Furthermore, since we only make use of the most probable sampled structure for our prediction, sampling can be done in-place, storing in each core only the best structure seen so far. This reduces the memory requirements and keeps the communication costs rather moderate since it is finally only necessary to gather m structures from m cores and select the best. In fact, we performed a series of experiments, making use of Mathematica's parallel computation features, which proved that the overall process scales linearly in the number of cores used with a non-measurable communication overhead. This finally proves the applicability of our approach, providing a factor n speedup compared to established prediction tools but still maintaining the limits implied by a quadratic memory consumption (in our case used to store parameter values).

Finally, it should be noted that plots as considered above, obtained by comprehensive experimental validation procedures for a particular class of RNAs and showing expected accuracies of sampling based prediction approaches depending on considered sample size, are also very interesting for biologists, but from a different perspective. In fact, if they want a specific value of a particular accuracy measure to be reached when applying the corresponding method to

a given input sequence (of which the RNA type is known), then they can easily read off an appropriate estimate of the minimum sample size required for ensuring the desired accuracy from the respective plot (for that class of RNAs). In cases where rather small predictive accuracies are sufficient and the respective plots indicate that only comparatively few candidate structures need to be generated, this simple action might actually help saving a significant amount of computation time in practice, whereas the results still meet the demands preliminary imposed on the applied tool.

9.5.4. Accuracy of Predictions According to Common Measures

In order to quantify the decline in predictive accuracy that results from considering different dimensions of approximations in the preprocessing step, we decided to record the sensitivities and PPVs for appropriate sample sizes. However, since there seems to be no generally valid, statistically sufficient size that can reliably be considered for any particular combination of applied preprocessing variant and used RNA data, it is not obvious which ones of the measured values should be recorded.

Of course, we could simply collect the accuracies of predicted foldings as observed for the largest considered sample size of 100 000 structures. Alternatively, we could consider a fixed sample size for any employed preprocessing variant, adapted to the used value of W_e, as we did in Section 9.5.2 for probability profiling of a particular tRNA sequence. However, the identification of such adapted sample sizes might be quite difficult. In fact, it seems impossible to preliminary choose fixed sample sizes only in dependence of W_e that might be useful in connection with any underlying stochastic model. Therefore, we decided to tabulate the best accuracies as measured for one of the considered samples sizes from the respective plots discussed in Section 9.5.3.

For the sake of simplicity, we used the *F measure* (or *F1 measure*, that is the harmonic mean of the sensitivity and PPV) [vR79] as a proxy for prediction accuracy, which is formally defined as follows:

$$F = \frac{1}{2} \cdot \frac{1}{\frac{1}{\text{Sens.}} + \frac{1}{\text{PPV}}}.$$

The corresponding results are collected in Tables 9.2 to 9.4. Note that whenever the same best F measure was observed for more than one sample sizes, we recorded the smallest value, since that one might in some sense mark the point where a certain saturation is reached.

Let us first consider the results reported in Table 9.2 for tRNAs. We find that the predictive accuracy is principally not affected by the different dimensions of approximation, as only in cases where the traditional SCFG model is considered, a slight change for the worse can be observed with respect to the sensitivity of corresponding MP predictions.

Table 9.3 shows that for 5S rRNAs, much of the accuracy of predicted structures is sacrificed when using heuristic preprocessing rather than the exact variant, especially in connection with our novel dynamic sampling strategy, although it was originally designed to fit especially well with approximated probabilities. Nevertheless, the decline in both resulting sensitivity and PPV is still acceptable when considering the LSCFG model. Length-independent variants of our heuristic method, however, seem to produce predictions that are not competitive.

The results reported in Table 9.4 demonstrate that sensitivity is considerably decreased with increasing degree of approximation. Additionally, when employing our reverse sampling strategy, there results a noticeable decline in the resulting PPV, especially if the conventional SCFG model is considered. By using the traditional sampling variant, however, to our surprise the best PPVs are observed in case of complete approximation, independent on the underlying stochastic model. In fact, strikingly better PPVs of MP predictions result when using the common strategy, not only in cases of heuristic but also in case of exact preprocessing, in contrast to the results derived for tRNAs and 5S rRNAs. This might be caused by the greater structural diversity of the S-151Rfam set (compared to the pure tRNA and 5S rRNA data sets) that has thus been trained into the corresponding models. Actually, this should inevitably yield more possible foldings and hence increase many of the problems related to our alternative sampling strategy (see Section 9.4.3), such that in the end more structures might be sampled that are rather unlikely according to the exact distribution.

In summary, these observations (unsurprisingly) affirm the ones discussed in the last section and we might actually conclude that our heuristic sampling variants might still yield highly accurate predictions, but may require the consideration of larger sample sizes.

Approach	W_e	MP struct.			
		Sens.	PPV	F	Sample size
SCFG	∞	0.880745	0.849710	0.216237	1000
	30	0.881603	0.849924	0.216368	60000
	15	0.851689	0.841067	0.211586	94000
	$-\infty$	0.851936	0.843642	0.211942	86000
LSCFG	∞	0.882012	0.951869	0.228902	1000
	30	0.884676	0.952994	0.229391	1000
	15	0.889795	0.955357	0.230353	13000
	$-\infty$	0.879655	0.949725	0.228337	67000

(a) Results obtained with the common sampling strategy.

Approach	W_e	MP struct.			
		Sens.	PPV	F	Sample size
SCFG	∞	0.893868	0.855383	0.218551	2000
	30	0.878999	0.859859	0.217331	19000
	15	0.841728	0.846801	0.211064	25000
	$-\infty$	0.833860	0.836906	0.208845	73000
LSCFG	∞	0.882012	0.951869	0.228902	1000
	30	0.887822	0.949056	0.229355	3000
	15	0.888702	0.950612	0.229654	3000
	$-\infty$	0.882964	0.951869	0.229031	39000

(b) Results obtained with the alternative sampling strategy (using restrictions $\max_{hairpin} = 15$, $\max_{bulge} = 26$ and $\max_{strand} = 23$).

Table 9.2.: **Sensitivity and PPV values for our tRNA benchmark.** Tabulated are the best accuracies of predicted foldings (according to F measure) as observed for one of the considered sample sizes, obtained on the basis of our tRNA benchmark (under the assumption of $\min_{\text{hel}} = 2$ and $\min_{\text{HL}} = 3$).

Approach	W_e	MP struct.			
		Sens.	PPV	F	Sample size
SCFG	∞	0.531928	0.599580	0.140933	28000
	30	0.429657	0.522789	0.117917	78000
	15	0.342774	0.426523	0.095022	13000
	$-\infty$	0.371491	0.477511	0.104470	36000
LSCFG	∞	0.943581	0.959726	0.237896	1000
	30	0.943067	0.963752	0.238324	16000
	15	0.926506	0.952880	0.234877	92000
	$-\infty$	0.912614	0.946247	0.232282	84000

(a) Results obtained with the common sampling strategy.

Approach	W_e	MP struct.			
		Sens.	PPV	F	Sample size
SCFG	∞	0.612660	0.594854	0.150906	12000
	30	0.365905	0.371562	0.092178	83000
	15	0.369055	0.388958	0.094686	90000
	$-\infty$	0.309099	0.321670	0.078814	59000
LSCFG	∞	0.943581	0.959726	0.237896	1000
	30	0.911843	0.935873	0.230925	82000
	15	0.875834	0.923022	0.224702	98000
	$-\infty$	0.861265	0.921509	0.222592	69000

(b) Results obtained with the alternative sampling strategy (using restrictions $\max_{hairpin} = 15$, $\max_{bulge} = 7$ and $\max_{strand} = 6$).

Table 9.3.: **Sensitivity and PPV values for our 5S rRNA benchmark.** Tabulated are the best accuracies of predicted foldings (according to F measure) as observed for one of the considered sample sizes, obtained on the basis of our 5S rRNA benchmark (under the assumption of $\min_{\text{hel}} = 2$ and $\min_{\text{HL}} = 3$).

Approach	W_e	MP struct.			
		Sens.	PPV	F	Sample size
SCFG	∞	0.572697	0.798884	0.166785	20000
	30	0.470512	0.804554	0.148444	2000
	$-\infty$	0.400108	0.824078	0.134669	86000
LSCFG	∞	0.468801	0.596504	0.131250	1000
	30	0.453797	0.561029	0.125437	31000
	$-\infty$	0.390029	0.639490	0.121134	21000

(a) Results obtained with the common sampling strategy.

Approach	W_e	MP struct.			
		Sens.	PPV	F	Sample size
SCFG	∞	0.626702	0.612907	0.154932	32000
	30	0.624085	0.554011	0.146741	61000
	$-\infty$	0.453194	0.495573	0.118359	30000
LSCFG	∞	0.458924	0.548578	0.124941	1000
	30	0.440791	0.536965	0.121037	91000
	$-\infty$	0.347933	0.515808	0.103889	15000

(b) Results obtained with the alternative sampling strategy (using restrictions $\max_{\mathtt{hairpin}} = \max_{\mathtt{bulge}} = \max_{\mathtt{strand}} = 30$).

Table 9.4.: **Sensitivity and PPV values for our S-151Rfam benchmark.** Tabulated are the best accuracies of predicted foldings (according to F measure) as observed for one of the considered sample sizes, obtained on the basis of our S-151Rfam benchmark (under the assumption of $\min_{\mathrm{hel}} = 2$ and $\min_{\mathrm{HL}} = 3$).

9.5.5. Predictive Power by Means of Abstract Shapes

In this section, we will exclusively deal with the $\mathrm{CSP}_{\mathrm{freq}}$ measure, since it relates to the shapes of predictions rather than to the complete set of sampled candidate structures. To obtain the corresponding results for our different benchmark sets, respectively, we simply considered the predicted foldings that were derived for the calculation of the sensitivity and PPV values discussed just before, in Section 9.5.4. The complete collection of all these $\mathrm{CSP}_{\mathrm{freq}}$ values is given by Tables 9.5 to 9.7. Note that the respective plots of averaged $\mathrm{CSP}_{\mathrm{freq}}$ values for MP predictions as functions of sample size (and also of the corresponding variances) are collected in Figures E.9 to E.20. They actually have been derived along with those presented in Figures 9.7 to 9.12, meaning the same sample sets and predictions have indeed been considered in either case.

First, as regards tRNAs (see Table 9.5), we find that approximate probabilities do generally not have a negative impact on the frequency of correct predictions of secondary structure (level 0 shapes) when using a LSCFG model. However, the consideration of approximated sampling probabilities does actually cause a significant change for the worse if a corresponding conventional SCFG is used as the basis for statistical sampling. Nevertheless, on the most abstract shape level (level 5), our heuristic variants surprisingly even seem to yield a slight improvement with respect to frequency of correct structure predictions.

Comparing the $\mathrm{CSP}_{\mathrm{freq}}$ values presented in Table 9.6, we immediately observe that for our 5S rRNA benchmark, the correct shape (on abstraction levels 1 to 5, but especially on the highest ones) might still be predicted for most of the considered structures where this has been the case by using exact preprocessing, but only when using the corresponding LSCFG model as

basis for our heuristic methods. Otherwise, that is when preprocessing is performed under the assumption of the plain SCFG model, the correct shape is predicted in significantly less cases, especially when using our dynamic sampling strategy for generating the candidate structures.

However, the results for S-151Rfam sequences reported in Table 9.7 indicate that using a statistical sampling approach based on the corresponding probabilistic model for mixed structural data generally leads to a (more or less) significant decline in the corresponding CSP_{freq} value in connection with approximative preprocessing variants. Nevertheless, in some rare cases the contrary might be observed, that is a (more or less) higher frequency of correct shape predictions on a particular abstraction level and for a particular heuristic variant might result (for instance, on shape level 2 when using mixed preprocessing according to $W_e = 30$ on the basis of the conventional SCFG and by applying the common sampling strategy).

Summarizing the results observed for our different data sets, it seems reasonable to believe that at least on the highest shape abstraction level of type 5 shapes (where only nesting and adjacency of helical regions are accounted for), the correct structure may still be predicted in an acceptable number of times with our heuristic methods (compared to the number of times it is predicted on the basis of exact preprocessing). This might validate the application of the discussed accelerated sampling approach in practice.

Approach	W_e	Shape Level					
		0	1	2	3	4	5
SCFG	∞	0.26	0.48	0.66	0.66	0.66	0.73
	30	0.23	0.46	0.66	0.66	0.66	0.73
	15	0.16	0.39	0.66	0.66	0.66	0.74
	$-\infty$	0.18	0.40	0.66	0.66	0.66	0.75
LSCFG	∞	0.33	0.44	0.70	0.70	0.70	0.79
	30	0.37	0.52	0.75	0.75	0.75	0.84
	15	0.32	0.49	0.73	0.73	0.73	0.81
	$-\infty$	0.35	0.49	0.73	0.73	0.73	0.82

(a) Results obtained with the common sampling strategy.

Approach	W_e	Shape Level					
		0	1	2	3	4	5
SCFG	∞	0.25	0.47	0.66	0.66	0.66	0.73
	30	0.22	0.51	0.65	0.65	0.65	0.73
	15	0.13	0.51	0.65	0.65	0.65	0.74
	$-\infty$	0.15	0.42	0.68	0.68	0.68	0.78
LSCFG	∞	0.33	0.44	0.70	0.70	0.70	0.79
	30	0.35	0.47	0.72	0.72	0.72	0.81
	15	0.33	0.51	0.76	0.76	0.76	0.85
	$-\infty$	0.34	0.45	0.70	0.70	0.70	0.79

(b) Results obtained with the alternative sampling strategy (using restrictions $\max_{hairpin} = 15$, $\max_{bulge} = 26$ and $\max_{strand} = 23$).

Table 9.5.: **Frequencies of correct shape predictions for our tRNA benchmark.** Table records the CSP_{freq} values (for selection principle MP struct.) corresponding to the accuracies listed in Table 9.2, obtained on the basis of our tRNA benchmark (under the assumption of $\min_{hel} = 2$ and $\min_{HL} = 3$).

Approach	W_e	Shape Level					
		0	1	2	3	4	5
SCFG	∞	0.00	0.02	0.06	0.10	0.12	0.76
	30	0.00	0.00	0.00	0.02	0.06	0.62
	15	0.00	0.00	0.00	0.06	0.08	0.50
	$-\infty$	0.00	0.00	0.00	0.00	0.08	0.56
LSCFG	∞	0.30	0.52	0.60	0.60	0.96	1.00
	30	0.20	0.46	0.56	0.60	0.94	1.00
	15	0.18	0.48	0.56	0.56	0.94	0.98
	$-\infty$	0.14	0.52	0.58	0.58	0.94	1.00

(a) Results obtained with the common sampling strategy.

Approach	W_e	Shape Level					
		0	1	2	3	4	5
SCFG	∞	0.00	0.06	0.10	0.30	0.28	0.88
	30	0.00	0.00	0.02	0.08	0.08	0.56
	15	0.00	0.00	0.00	0.02	0.04	0.48
	$-\infty$	0.00	0.00	0.00	0.00	0.04	0.40
LSCFG	∞	0.30	0.52	0.60	0.60	0.96	1.00
	30	0.22	0.44	0.62	0.66	0.90	1.00
	15	0.08	0.52	0.64	0.64	0.94	1.00
	$-\infty$	0.04	0.36	0.48	0.58	0.80	1.00

(b) Results obtained with the alternative sampling strategy (using restrictions $\max_{hairpin} = 15$, $\max_{bulge} = 7$ and $\max_{strand} = 6$).

Table 9.6.: **Frequencies of correct shape predictions for our 5S rRNA benchmark.** Table records the CSP_{freq} values (for selection principle MP struct.) corresponding to the accuracies listed in Table 9.3, obtained on the basis of our 5S rRNA benchmark (under the assumption of $\min_{hel} = 2$ and $\min_{HL} = 3$).

Approach	W_e	Shape Level					
		0	1	2	3	4	5
SCFG	∞	0.07	0.13	0.13	0.20	0.13	0.40
	30	0.00	0.67	0.07	0.13	0.13	0.40
	$-\infty$	0.00	0.07	0.07	0.07	0.07	0.33
LSCFG	∞	0.20	0.20	0.27	0.27	0.27	0.53
	30	0.13	0.13	0.20	0.27	0.20	0.60
	$-\infty$	0.00	0.07	0.13	0.20	0.13	0.33

(a) Results obtained with the common sampling strategy.

Approach	W_e	Shape Level					
		0	1	2	3	4	5
SCFG	∞	0.13	0.20	0.20	0.27	0.20	0.47
	30	0.07	0.07	0.13	0.13	0.27	0.53
	$-\infty$	0.00	0.00	0.00	0.00	0.07	0.33
LSCFG	∞	0.20	0.20	0.20	0.20	0.20	0.53
	30	0.00	0.07	0.07	0.07	0.07	0.47
	$-\infty$	0.00	0.07	0.07	0.07	0.07	0.33

(b) Results obtained with the alternative sampling strategy (using restrictions $\max_{hairpin} = \max_{bulge} = \max_{strand} = 30$).

Table 9.7.: **Frequencies of correct shape predictions for our S-151Rfam benchmark.** Table records the CSP_{freq} values (for selection principle MP struct.) corresponding to the accuracies listed in Table 9.4, obtained on the basis of our S-151Rfam benchmark (under the assumption of $\min_{hel} = 2$ and $\min_{HL} = 3$).

9.6. Evaluation by Comparison to Leading Prediction Methods

To complete this chapter, we finally want to evaluate the predictive power of our heuristic (L)SCFG based sampling methods by a comparison to several leading tools implementing different probabilistic and physics-based approaches for RNA secondary structure prediction from a single sequence. Particularly, we used

- `CONTRAfold` 2.02 [DWB06],
- `PPfold` 2.0 [SKV+11] (a parallelized version of `Pfold` [KH99, KH03]),
- `RNAshapes` 2.1.6 [SVR+06b],
- `Sfold` 2.2 [DL03, DCL04],
- `UNAFold` 3.8 [MZ08] (which recently replaced `Mfold` [Zuk89b, Zuk03]) and
- `ViennaRNA` 1.8.5 [HFS+94, Hof03]

for deriving a single prediction for any of the RNA sequences in our three different benchmark sets, respectively. Afterwards, we calculated the corresponding accuracy measures sensitivity and PPV, as well as the corresponding frequencies of correct shape predictions, that is results equivalent to those considered in the preceding sections for any of our three benchmarks. The obtained values are collected in Tables 9.8 and 9.9.

First, comparing the results for tRNAs presented in Table 9.8a to those given in Table 9.2, we observe that the best accuracies (as measured for one of the considered sample sizes) obtained with our heuristic methods – even with complete approximation – are in any case superior to the corresponding accuracies obtained with the considered tools. In fact, only `PPfold` manages to reach a similar high PPV as our heuristic sampling methods based on the traditional SCFG model. Accordingly, the heuristic sampling methods devised in this chapter are all considerably more accurate with respect to correct shape predictions than the leading tools in connection with tRNA data, as can be observed from Tables 9.9a and 9.5

For 5S rRNAs, however, significantly more accurate prediction results can only be reached when considering the LSCFG model as basis for (exact or approximate) preprocessing. With respect to sensitivity and PPV, the (heuristic) statistical sampling approach based on the plain SCFG model is basically outperformed by all considered probabilistic and energy-based structure prediction methods. In fact, a competitive accuracy as measured by both sensitivity and PPV can only be reached by exact preprocessing and using the dynamic sampling strategy. This, together with

Tool	Choice	Sens.	PPV
`CONTRAfold`		0.804961	0.711828
`PPfold`		0.598735	0.844887
`RNAshapes`		0.664397	0.601762
`Sfold`	MP struct.	0.646401	0.578329
`Sfold`	Centroid	0.617218	0.641882
`UNAFold`		0.647860	0.590426
`ViennaRNA`		0.647374	0.577942

(a) Results obtained on the basis of our tRNA benchmark.

Tool	Choice	Sens.	PPV
`CONTRAfold`		0.671325	0.645352
`PPfold`		0.323644	0.681979
`RNAshapes`		0.635551	0.626101
`Sfold`	MP struct.	0.626048	0.628155
`Sfold`	Centroid	0.650084	0.717016
`UNAFold`		0.655115	0.658797
`ViennaRNA`		0.629402	0.624515

(b) Results obtained on the basis of our 5S rRNA benchmark.

Tool	Choice	Sens.	PPV
`CONTRAfold`		0.796034	0.648961
`PPfold`		0.532578	0.728682
`RNAshapes`		0.722380	0.578231
`Sfold`	MP struct.	0.725212	0.577878
`Sfold`	Centroid	0.742210	0.628297
`UNAFold`		0.742210	0.592760
`ViennaRNA`		0.756374	0.604072

(c) Results obtained on the basis of our S-151Rfam benchmark.

Table 9.8.: **Accuracies of leading RNA secondary structure prediction tools.**

Tool	Choice	Shape Level					
		0	1	2	3	4	5
CONTRAfold		0.00	0.21	0.25	0.25	0.35	0.54
PPfold		0.00	0.02	0.17	0.17	0.17	0.18
RNAshapes		0.00	0.18	0.35	0.35	0.38	0.44
Sfold	MP struct.	0.00	0.13	0.27	0.28	0.34	0.44
Sfold	Centroid	0.00	0.09	0.23	0.23	0.27	0.33
UNAFold		0.00	0.17	0.33	0.33	0.36	0.45
ViennaRNA		0.00	0.14	0.26	0.27	0.33	0.42

(a) Results obtained on the basis of our tRNA benchmark.

Tool	Choice	Shape Level					
		0	1	2	3	4	5
CONTRAfold		0.00	0.00	0.02	0.06	0.06	0.70
PPfold		0.00	0.00	0.00	0.00	0.02	0.48
RNAshapes		0.00	0.00	0.00	0.12	0.04	0.72
Sfold	MP struct.	0.00	0.00	0.00	0.08	0.02	0.66
Sfold	Centroid	0.00	0.00	0.00	0.08	0.12	0.82
UNAFold		0.00	0.00	0.00	0.10	0.04	0.72
ViennaRNA		0.00	0.00	0.00	0.10	0.02	0.76

(b) Results obtained on the basis of our 5S rRNA benchmark.

Tool	Choice	Shape Level					
		0	1	2	3	4	5
CONTRAfold		0.00	0.20	0.20	0.20	0.27	0.60
PPfold		0.00	0.20	0.20	0.20	0.20	0.40
RNAshapes		0.00	0.07	0.07	0.07	0.13	0.53
Sfold	MP struct.	0.00	0.07	0.07	0.07	0.13	0.60
Sfold	Centroid	0.00	0.13	0.13	0.13	0.27	0.73
UNAFold		0.00	0.07	0.07	0.07	0.13	0.67
ViennaRNA		0.00	0.07	0.07	0.07	0.13	0.67

(c) Results obtained on the basis of our S-151Rfam benchmark.

Table 9.9.: **Shape prediction accuracies of leading RNA secondary structure prediction tools.** Table contains CSP_{freq} values corresponding to the accuracies listed in Table 9.8.

the fact that the by far highest CSP_{freq} value on any shape level is obtained with this sampling variant in connection with exact probabilities, validates the consideration of our less restrictive dynamic sampling strategy as an attractive – albeit heuristic – alternative to the common one. Moreover, with respect to correct shape predictions, any of our heuristic sampling approaches is competitive (at least) to PPfold, especially when employing the common sampling strategy for the stochastic traceback step; with this strategy, PPfold is beaten even in the case of complete approximation for the preprocessing step. This indicates that our (heuristic) statistical sampling approaches are more accurate with respect to predicting the correct shape of foldings than to obtain high sensitivity and PPV, which from the perspective of biologists is more important in practice.

Finally, considering the results for our S-151Rfam benchmark, we find that the LSCFG based sampling approach is outperformed by all leading tools – even in case of exact inside-outside calculations – with respect to sensitivity and PPV of predicted foldings. However, as regards correct shape predictions, heuristic LSCFG based sampling variants with mixed preprocessing are competitive to some of the considered tools, especially in connection with the common sampling strategy. When relying on the plain SCFG model for the preprocessing step, then the mixed variant with constant $W_e = 30$, combined with either sampling strategy, seems to still yield an acceptable predictive accuracy compared to the well-established tools.

In summary, opposing the results obtained with our diverse heuristic statistical sampling variants for the different benchmark sets to those derived on the basis of several leading RNA tools indicates the validity of our approach with respect to particular applications. For instance, if correct shape prediction (on high abstraction levels) is of practical concern and a noticeable but still acceptable decrease in quality of proposed predictions might willingly be tolerated in favor of saving a significant amount of time, then a mixed preprocessing variant in connection with the common strategy may be employed.

9.7. Conclusions

Obviously, the major advantage of applying (most of) the heuristic sampling methods studied in this chapter for RNA secondary structure prediction form a single sequence is their increased efficiency compared to all other modern prediction algorithms (implemented in popular tools like the ones considered in Section 9.6). In fact, the worst-case time complexity can be reduced by a linear factor, such that the time and space requirements are both bounded by $\mathcal{O}(n^2)$ for an input sequence of length n.

However, a potential drawback lies in the observation that due to the approximated ensemble distribution, the overall quality of generated samples decreases (as indicated by probability profiling for specific loop types in Section 9.5.2). As a consequence, we usually need to use larger sample sizes for obtaining a competitive prediction accuracy and stable predictions. That is, more candidate structures for a given input sequence have to be generated to ensure that the algorithm outputs rather identical predictions in independent runs for that sequence (as discussed in Section 9.5.3). Nevertheless, according to our experiments, an efficient implementation that really takes advantage of the accelerated preprocessing but also handles large sample sizes can indeed be obtained by parallelization.

The practical applicability of the described heuristic statistical sampling methods to the problem of RNA folding has been indicated by the fact that the proposed prediction approach actually seems to be capable of yielding acceptable accuracies even for such types of RNAs whose molecules imply a great variety of structural features (see Sections 9.5.4, 9.5.5 and 9.6).

Note that all results presented in this chapter have been derived with a purposive proof-of-concept implementation of the described methods. A more sophisticated tool is planned to be realized in the near future. Furthermore, we have to clarify that the algorithms presented in this chapter are only heuristics rather than proper approximation algorithms, since they are not evidentially relative error bounded.

Chapter 10

Final Conclusions and Suggestions for Future Research

This chapter provides a summary of the results and main conclusions of this thesis, which were presented in Chapters 5 to 9. Furthermore, it suggests what further work could be interesting to pursue in connection with our scientific contributions.

10.1. Final Conclusions

First of all, the non-uniform random sampling method devised in Chapter 5 is capable of producing realistic secondary structures for any fixed size n, at least with respect to the appearance of the different structural motifs, and can be suited for any specific RNA type. Moreover, it is highly efficient. Particularly, a set of m random structures of size n can be sampled in $\mathcal{O}(m \cdot n \cdot \log(n))$ time after a preprocessing step in time $\mathcal{O}(n^2)$, which is not needed if the corresponding values are stored on the computer; other methods typically need $\mathcal{O}(m \cdot n^2)$ time in any case. For these reasons, it may be a handy tool for practical applications.

Chapter 6 shows that with respect to single sequence RNA secondary structure prediction, more complex SCFG designs can in some cases yield higher accuracy but may require larger and more reliable training sets than lightweight SCFGs or corresponding CLLMs. The presented SCFG based statistical sampling algorithm is capable of producing accurate (prediction) results, where its worst-case time and space requirements are equal to those of common RNA folding algorithms for single sequences. A greater structural diversity within generated samples is implied compared to the sampling with PFs. This constitutes a meaningful result, as that is actually a crucial aspect for preferring statistical sampling approaches over deterministic optimization algorithms for RNA folding.

The evaluations of Chapter 7 indicate that using length-dependent grammar parameters rather than their conventional counterparts can results in a more explicit and very powerful RNA model (for a specific class), such that the resulting sampling quality is more independent of (the complexity of) the specified RNA type. However, length-dependent probabilistic models seem to be more prone to overfitting (due to the required larger training sets) and lack of generalization (at least for low invariant RNA types obeying to a single typical shape, as that is then desperately trained into the model).

As demonstrated in Chapter 8, our (L)SCFG based statistical sampling approach behaves rather resistant to relative errors, but seems to be extremely sensitive as regards absolute errors. Notably, the discussion of potential problems that may be caused by disturbances in the conditional sampling distributions, the suggestions on how to prevent them as much as possible, and the reasons why they sometimes can hardly be avoided lead to meaningful conclusions that actually constituted the basis for designing our heuristic approximation methods. Therefore, the information gathered during our disturbance analysis might also be useful in other similar contexts.

Finally, Chapter 9 provided evidence that the idea of using the technique of (heuristic) approximation in order to reduce computation time may be successfully applied in the context of RNA secondary structure prediction, but at the cost of the optimality of the results. In fact, the time complexity of the preprocessing step and thus of our overall sampling algorithm has been reduced from cubic to quadratic in the worst-case, where the decline in accuracy that results from approximating the sampling probabilities seems to depend on the considered RNA type. Anyway, our heuristic methods can still yield predictions of respectable accuracy (given the fact that preprocessing is highly accelerated), but may require the consideration of larger sample sizes. This, however, is quite unproblematic in practice, since we can generate the candidate structures in-place (only the so far most probable structure needs to be stored), such that large sample sizes give no rise to memory consumption. Furthermore, generating samples can easily be parallelized or distributed on modern multi-core architectures or grids. In addition, our novel heuristic sampling strategy seems to produce more accurate results than the common one for particular settings, for instance in case of exact preprocessing for specific RNA types, although it was originally designed to fit especially well with our (mixed) heuristic preprocessing variants.

10.2. Suggestions for Future Research

The present thesis leaves a number of open questions that may be inspiration for further research in this area. For instance, recall that we used a sophisticated SCFG (representing a formal language counterpart to the thermodynamic model applied in the `Sfold` program) as probabilistic basis for the devised statistical sampling algorithms. However, it would also be possible to employ other SCFG designs, for example one of the commonly known lightweight grammars introduced in [DE04]. This might of course yield at least noticeable if not significant changes in the resulting sampling quality, which could be an interesting subject to be explored.

With respect to the improving the efficiency of RNA folding algorithms, it should be noticed that the critical point of the accelerated inside-outside calculations is the abstraction from the input sequence, which requires approximation of emission probability terms and results in fuzzy ensemble distributions. In this thesis, we only implemented a quite simple and straightforward idea for designing an appropriate heuristic that improved time performance on the basis of some of the results presented in Chapter 8. However, there might be other ways for speeding up SCFG based statistical sampling, for instance by developing a corresponding proper approximation algorithm.

In this context, it should also be noted that a similar heuristic approximation approach could potentially be considered in an attempt to reduce the worst-case time complexity of the sampling extension of the PF approach. In fact, sequence information is incorporated into the used (equilibrium) PFs and corresponding sampling probabilities only in the form of particular sequence-dependent free energy contributions (see Sections 3.2.3.2 and 3.2.4.1). Hence, it seems reasonable to believe that the time complexity for the preprocessing step (that is, for PF calculations) could possibly be reduced by a linear factor to $\mathcal{O}(n^2)$, for n the length of the input sequence, by using some sort of approximated (averaged) free energy contributions that do not depend on the actual sequence (but should rather contain as much sequence information as possible) – in analogy to the approximated preprocessing step (that is, inside-outside calculations) described in Section 9.3. Since there, we basically only had to use averaged emission terms instead of the exact emission probabilities in order to save time.

As regards the overall sampling algorithm, there is of course also room for improvement. For example, the results presented in Section 9.5.2 indicated that a gain in sampling quality could potentially be reached by considering increasing (but still reasonable, that is still small enough) values of the minimum allowed number $\min_{\text{hel}}$ of base pairs in helices for our heuristic sampling methods. However, although greater values of $\min_{\text{hel}}$ seem more realistic (due to the fact that longer helices typically imply higher structure stability), they unfortunately limit the number of possible foldings for a given input sequence, such that the native structure might not be formed if $\min_{\text{hel}}$ is chosen too large. Besides, greater values of $\min_{\text{hel}}$ also limit the corresponding amount of available training data, since only those trusted structures can be considered for parameter estimation that can be generated by the underlying parameterized SCFG according to the particular choice of $\min_{\text{hel}}$. A solution to this problem might lie in replacing this static restriction by considerations of the actual sizes of generated helices during the sampling process, such that a particular probability distribution for sampling the next substructure depends stronger on the length of the helix to be constructed. This could be realized, for example, by rewarding the inclusion of additional base pairs until a certain reasonable helix length is reached (proportional to its current length), whereas once a particular number of base pairs has been formed, helix extensions are penalized (also in proportion to the length of the already constructed stem). In order to obtain appropriate rewards, penalties, etc., one could for instance use some sort of expected values and variances derived from specific databases of RNAs, perhaps combined with observations from standard thermodynamic models (for example, in a similar fashion as done in [NS11a]).

Additionally, as already indicated by the end of Section 6.7, this work may inspire the development of new (heuristic) sampling algorithms for RNA structures with pseudoknots or RNA-RNA interactions.

Part III

Appendix

Chapter A

Derivation of the Weighted Unranking Algorithm

In this section, we give a complete and detailed description of the derivation of our weighted unranking algorithm for RNA secondary structures. The different steps are made according to the approach described in [WN10] to get an unranking algorithm that generates random RNA secondary structures of a given size n according to the distribution on all these structures.

A.1. Considered (unambiguous, ϵ-free and loop-free) SCFG

First, note that in [NS11a], to obtain the stochastic model for RNA secondary structures derived from real-world RNA data, the following unambiguous SCFG which unambiguously generates exactly the language $\mathcal{L}$ given in Definition 3.3.2 has been used:

Definition A.1.1 *The unambiguous SCFG $\mathcal{G}_u$ generating exactly the language $\mathcal{L}$ is given by $\mathcal{G}_u = (I_{\mathcal{G}_u}, \Sigma_{\mathcal{G}_u}, R_{\mathcal{G}_u}, S)$, where*

$$I_{\mathcal{G}_u} = \{S, T, C, A, L, G, B, F, H, P, Q, R, J, K, M, N, U\},$$

$\Sigma_{\mathcal{G}_u} = \{(,), \circ\}$ *and $R_{\mathcal{G}_u}$ contains exactly the following rules:*

$$\begin{array}{llll}
p_1 : S \to TAC, & & & \\
p_2 : T \to TAC, & p_3 : T \to C, & & \\
p_4 : C \to C\circ, & p_5 : C \to \epsilon, & & \\
p_6 : A \to (L), & & & \\
p_7 : L \to (L), & p_8 : L \to M, & p_9 : L \to P, & p_{10} : L \to Q, \\
p_{11} : L \to R, & p_{12} : L \to F, & p_{13} : L \to G, & \\
p_{14} : G \to (L)\circ, & p_{15} : G \to (L)B\circ\circ, & p_{16} : G \to \circ(L), & p_{17} : G \to \circ\circ B(L), \\
p_{18} : B \to B\circ, & p_{19} : B \to \epsilon, & & \\
p_{20} : F \to \circ\circ\circ, & p_{21} : F \to \circ\circ\circ\circ, & p_{22} : F \to \circ\circ\circ\circ\circ H, & \\
p_{23} : H \to H\circ, & p_{24} : H \to \epsilon, & & \\
p_{25} : P \to \circ(L)\circ, & p_{26} : P \to \circ(L)\circ\circ, & p_{27} : P \to \circ\circ(L)\circ, & p_{28} : P \to \circ\circ(L)\circ\circ, \\
p_{29} : Q \to \circ\circ(L)K\circ\circ\circ, & p_{30} : Q \to \circ\circ\circ J(L)K\circ\circ, & & \\
p_{31} : R \to \circ(L)K\circ\circ\circ, & p_{32} : R \to \circ\circ\circ J(L)\circ, & & \\
p_{33} : J \to J\circ, & p_{34} : J \to \epsilon, & & \\
p_{35} : K \to K\circ, & p_{36} : K \to \epsilon, & & \\
p_{37} : M \to U(L)U(L)N, & & & \\
p_{38} : N \to U(L)N, & p_{39} : N \to U, & & \\
p_{40} : U \to U\circ, & p_{41} : U \to \epsilon. & &
\end{array}$$

In this grammar, different intermediate symbols have been used to distinguish between different substructures. In fact, the reason why this grammar has so many production rules is that the grammar must be able to distinguish between all the different classes of substructures for which there are different free energy rules according to Turner's thermodynamic model considered in [NS11a].

However, as ϵ-freeness and loop-freeness are required preliminarily, we have to consider another unambiguous SCFG generating the same language $\mathcal{L}$, where we have to guarantee that the same substructures are distinguished as are distinguished in $\mathcal{G}_u$. Using the usual way of transforming a non-ϵ-free grammar into an ϵ-free one, the following definition can immediately be obtained from the previous one:

Definition A.1.2 *The unambiguous and ϵ-free SCFG $\mathcal{G}'_u$ generating exactly the language $\mathcal{L}$ is given by $\mathcal{G}'_u = (I_{\mathcal{G}'_u}, \Sigma_{\mathcal{G}'_u}, R_{\mathcal{G}'_u}, S')$, where*

$$I_{\mathcal{G}'_u} = \{S', S, T, C, A, L, G, B, F, H, P, Q, R, J, K, M, N, U\},$$

$\Sigma_{\mathcal{G}'_u} = \{(,), \circ\}$ *and $R_{\mathcal{G}'_u}$ contains exactly the following rules:*

$p'_0 : S' \to S,$
$p'_1 : S \to A,$ $p'_2 : S \to AC,$ $p'_3 : S \to TA,$ $p'_4 : S \to TAC,$
$p'_5 : T \to A,$ $p'_6 : T \to AC,$ $p'_7 : T \to TA,$ $p'_8 : T \to TAC,$
$p'_9 : T \to C,$
$p'_{10} : C \to \circ,$ $p'_{11} : C \to C\circ,$
$p'_{12} : A \to (L),$
$p'_{13} : L \to (L),$ $p'_{14} : L \to M,$ $p'_{15} : L \to P,$ $p'_{16} : L \to Q,$
$p'_{17} : L \to R,$ $p'_{18} : L \to F,$ $p'_{19} : L \to G,$
$p'_{20} : G \to (L)\circ,$ $p'_{21} : G \to (L)\circ\circ,$ $p'_{22} : G \to (L)B\circ\circ,$
$p'_{23} : G \to \circ(L),$ $p'_{24} : G \to \circ\circ(L),$ $p'_{25} : G \to \circ\circ B(L),$
$p'_{26} : B \to \circ,$ $p'_{27} : B \to B\circ,$
$p'_{28} : F \to \circ\circ\circ,$ $p'_{29} : F \to \circ\circ\circ\circ,$ $p'_{30} : F \to \circ\circ\circ\circ\circ,$ $p'_{31} : F \to \circ\circ\circ\circ\circ H,$
$p'_{32} : H \to \circ,$ $p'_{33} : H \to H\circ,$
$p'_{34} : P \to \circ(L)\circ,$ $p'_{35} : P \to \circ(L)\circ\circ,$ $p'_{36} : P \to \circ\circ(L)\circ,$ $p'_{37} : P \to \circ\circ(L)\circ\circ,$
$p'_{38} : Q \to \circ\circ(L)\circ\circ\circ,$ $p'_{39} : Q \to \circ\circ(L)K\circ\circ\circ,$ $p'_{40} : Q \to \circ\circ\circ(L)\circ\circ,$ $p'_{41} : Q \to \circ\circ\circ J(L)\circ\circ,$
$p'_{42} : Q \to \circ\circ\circ(L)K\circ\circ,$ $p'_{43} : Q \to \circ\circ\circ J(L)K\circ\circ,$
$p'_{44} : R \to \circ(L)\circ\circ\circ,$ $p'_{45} : R \to \circ(L)K\circ\circ\circ,$ $p'_{46} : R \to \circ\circ\circ(L)\circ,$ $p'_{47} : R \to \circ\circ\circ J(L)\circ,$
$p'_{48} : J \to \circ,$ $p'_{49} : J \to J\circ,$
$p'_{50} : K \to \circ,$ $p'_{51} : K \to K\circ,$
$p'_{52} : M \to (L)(L),$ $p'_{53} : M \to U(L)(L),$ $p'_{54} : M \to (L)U(L),$ $p'_{55} : M \to (L)(L)N,$
$p'_{56} : M \to U(L)U(L),$ $p'_{57} : M \to U(L)(L)N,$ $p'_{58} : M \to (L)U(L)N,$ $p'_{59} : M \to U(L)U(L)N,$
$p'_{60} : N \to (L),$ $p'_{61} : N \to U(L),$ $p'_{62} : N \to (L)N,$ $p'_{63} : N \to U(L)N,$
$p'_{64} : N \to U,$
$p'_{65} : U \to \circ,$ $p'_{66} : U \to U\circ.$

Unfortunately, the set of productions of $\mathcal{G}'_u$ contains productions with up to five nonterminal symbols in the conclusion. This is not acceptable for our purpose, for the following reason: the desired unranking algorithm makes use of the size of combinatorial classes whose representations somehow are derived from CFGs with particular integer weights on their productions. If we constructed this WCFG by starting with the grammar $\mathcal{G}'_u$, then this would yield a huge number of production rules. Consequently, the translation would imply a huge specification of the combinatorial classes and the corresponding function to compute their sizes and thus the corresponding unranking algorithm would have to distinguish between an unnecessarily and most importantly unacceptably large number of cases.

Nevertheless, the size of the production set of the weighted grammar underlying the desired unranking algorithm can be significantly reduced by starting with a modification of grammar $\mathcal{G}'_u$ which has only production rules with minimum possible numbers of nonterminal symbols in the conclusion. In fact, by transforming $\mathcal{G}'_u$ appropriately considering this observation, we obtained the SCFG $\widehat{\mathcal{G}}_u$:

Definition A.1.3 *The unambiguous ϵ-free SCFG $\widehat{\mathcal{G}}_u$ generating exactly the language $\mathcal{L}$ is given by $\widehat{\mathcal{G}}_u = (I_{\widehat{\mathcal{G}}_u}, \Sigma_{\widehat{\mathcal{G}}_u}, R_{\widehat{\mathcal{G}}_u}, S')$, where*

$$I_{\widehat{\mathcal{G}}_u} = \{S', E, S, T, C, A, L, G, D, B, F, H, P, Q, R, V, W, O, J, K, M, X, Y, Z, N, U\},$$

$\Sigma_{\widehat{\mathcal{G}}_u} = \{(,), \circ\}$ *and* $R_{\widehat{\mathcal{G}}_u}$ *contains exactly the following rules:*

$\widehat{p}_1 : S' \to E,$
$\widehat{p}_2 : E \to S,$ $\widehat{p}_3 : E \to SC,$

$\widehat{p}_4 : S \to A$, $\widehat{p}_5 : S \to TA$,
$\widehat{p}_6 : T \to E$, $\widehat{p}_7 : T \to C$,
$\widehat{p}_8 : C \to \circ$, $\widehat{p}_9 : C \to C\circ$,
$\widehat{p}_{10} : A \to (L)$,
$\widehat{p}_{11} : L \to A$, $\widehat{p}_{12} : L \to M$, $\widehat{p}_{13} : L \to P$, $\widehat{p}_{14} : L \to Q$,
$\widehat{p}_{15} : L \to R$, $\widehat{p}_{16} : L \to F$, $\widehat{p}_{17} : L \to G$,
$\widehat{p}_{18} : G \to A\circ$, $\widehat{p}_{19} : G \to AD$, $\widehat{p}_{20} : G \to \circ A$, $\widehat{p}_{21} : G \to DA$,
$\widehat{p}_{22} : D \to B\circ$,
$\widehat{p}_{23} : B \to \circ$, $\widehat{p}_{24} : B \to B\circ$,
$\widehat{p}_{25} : F \to \circ\circ\circ$, $\widehat{p}_{26} : F \to \circ\circ\circ\circ$, $\widehat{p}_{27} : F \to \circ\circ\circ\circ H$,
$\widehat{p}_{28} : H \to \circ$, $\widehat{p}_{29} : H \to H\circ$,
$\widehat{p}_{30} : P \to \circ A\circ$, $\widehat{p}_{31} : P \to \circ A\circ\circ$, $\widehat{p}_{32} : P \to \circ\circ A\circ$, $\widehat{p}_{33} : P \to \circ\circ A\circ\circ$,
$\widehat{p}_{34} : Q \to \circ\circ O\circ\circ$, $\widehat{p}_{35} : Q \to \circ\circ V\circ$,
$\widehat{p}_{36} : R \to \circ O\circ\circ$, $\widehat{p}_{37} : R \to \circ\circ W\circ$,
$\widehat{p}_{38} : V \to JO$,
$\widehat{p}_{39} : W \to JA$,
$\widehat{p}_{40} : O \to AK$,
$\widehat{p}_{41} : J \to \circ$, $\widehat{p}_{42} : J \to J\circ$,
$\widehat{p}_{43} : K \to \circ$, $\widehat{p}_{44} : K \to K\circ$,
$\widehat{p}_{45} : M \to XY$,
$\widehat{p}_{46} : X \to A$, $\widehat{p}_{47} : X \to UA$,
$\widehat{p}_{48} : Y \to Z$,
$\widehat{p}_{49} : Z \to X$, $\widehat{p}_{50} : Z \to XN$,
$\widehat{p}_{51} : N \to Z$, $\widehat{p}_{52} : N \to U$,
$\widehat{p}_{53} : U \to \circ$, $\widehat{p}_{54} : U \to U\circ$.

A.2. Transforming our SCFG into RNF

Now, we can construct the desired weighted grammar that will be underlying our unranking algorithm: In the first step, we gather all possible chains of productions that do not lengthen the sentential form. In fact, we have to consider all rules $A \to \alpha$, $A \neq S'$, with $|\alpha| = 1$, to obtain all such chains (note that these rules will be removed after step 1). Hence, we have to consider the following set R^1_{rnf} of 22 production rules:

$\widehat{p}_2 : E \to S$,
$\widehat{p}_4 : S \to A$,
$\widehat{p}_6 : T \to E$, $\widehat{p}_7 : T \to C$,
$\widehat{p}_8 : C \to \circ$,
$\widehat{p}_{11} : L \to A$, $\widehat{p}_{12} : L \to M$, $\widehat{p}_{13} : L \to P$, $\widehat{p}_{14} : L \to Q$,
$\widehat{p}_{15} : L \to R$, $\widehat{p}_{16} : L \to F$, $\widehat{p}_{17} : L \to G$,
$\widehat{p}_{23} : B \to \circ$,
$\widehat{p}_{28} : H \to \circ$,
$\widehat{p}_{41} : J \to \circ$,
$\widehat{p}_{43} : K \to \circ$,
$\widehat{p}_{46} : X \to A$,
$\widehat{p}_{48} : Y \to Z$,
$\widehat{p}_{49} : Z \to X$,

$$\widehat{p}_{51} : N \to Z, \quad \widehat{p}_{52} : N \to U,$$
$$\widehat{p}_{53} : U \to \circ.$$

Thus, the following 32 chains are gathered in step 1:

$$\begin{aligned}
&E \Rightarrow S, && targets[E] = \{(S, \lambda_{E,S} := \widehat{p}_2, \epsilon),\\
&E \Rightarrow S \Rightarrow A, && \qquad\qquad (A, \lambda_{E,A} := \widehat{p}_2 \cdot \widehat{p}_4, S)\},\\
&S \Rightarrow A, && targets[S] = \{(A, \lambda_{S,A} := \widehat{p}_4, \epsilon)\},\\
&T \Rightarrow E, && targets[T] = \{(E, \lambda_{T,E} := \widehat{p}_6, \epsilon),\\
&T \Rightarrow C, && \qquad\qquad (C, \lambda_{T,C} := \widehat{p}_7, \epsilon),\\
&T \Rightarrow C \Rightarrow \circ, && \qquad\qquad (\circ, \lambda_{T,\circ} := \widehat{p}_7 \cdot \widehat{p}_8, C),\\
&T \Rightarrow E \Rightarrow S, && \qquad\qquad (S, \lambda_{T,S} := \widehat{p}_6 \cdot \widehat{p}_2, E)\},\\
&T \Rightarrow E \Rightarrow S \Rightarrow A, && \qquad\qquad (A, \lambda_{T,A} := \widehat{p}_6 \cdot \widehat{p}_2 \cdot \widehat{p}_4, ES)\},\\
&C \Rightarrow \circ, && targets[C] = \{(\circ, \lambda_{C,\circ} := \widehat{p}_8, \epsilon)\},\\
&L \Rightarrow A, && targets[L] = \{(A, \lambda_{L,A} := \widehat{p}_{11}, \epsilon),\\
&L \Rightarrow M, && \qquad\qquad (M, \lambda_{L,M} := \widehat{p}_{12}, \epsilon),\\
&L \Rightarrow P, && \qquad\qquad (P, \lambda_{L,P} := \widehat{p}_{13}, \epsilon),\\
&L \Rightarrow Q, && \qquad\qquad (Q, \lambda_{L,Q} := \widehat{p}_{14}, \epsilon),\\
&L \Rightarrow R, && \qquad\qquad (R, \lambda_{L,R} := \widehat{p}_{15}, \epsilon),\\
&L \Rightarrow F, && \qquad\qquad (F, \lambda_{L,F} := \widehat{p}_{16}, \epsilon),\\
&L \Rightarrow G, && \qquad\qquad (G, \lambda_{L,G} := \widehat{p}_{17}, \epsilon)\},\\
&B \Rightarrow \circ, && targets[B] = \{(\circ, \lambda_{B,\circ} := \widehat{p}_{23}, \epsilon)\},\\
&H \Rightarrow \circ, && targets[H] = \{(\circ, \lambda_{H,\circ} := \widehat{p}_{28}, \epsilon)\},\\
&J \Rightarrow \circ, && targets[J] = \{(\circ, \lambda_{J,\circ} := \widehat{p}_{41}, \epsilon)\},\\
&K \Rightarrow \circ, && targets[K] = \{(\circ, \lambda_{K,\circ} := \widehat{p}_{43}, \epsilon)\},\\
&X \Rightarrow A, && targets[X] = \{(A, \lambda_{X,A} := \widehat{p}_{46}, \epsilon)\},\\
&Y \Rightarrow Z, && targets[Y] = \{(Z, \lambda_{Y,Z} := \widehat{p}_{48}, \epsilon),\\
&Y \Rightarrow Z \Rightarrow X, && \qquad\qquad (X, \lambda_{Y,X} := \widehat{p}_{48} \cdot \widehat{p}_{49}, Z),\\
&Y \Rightarrow Z \Rightarrow X \Rightarrow A, && \qquad\qquad (A, \lambda_{Y,A} := \widehat{p}_{48} \cdot \widehat{p}_{49} \cdot \widehat{p}_{46}, ZX)\},\\
&Z \Rightarrow X, && targets[Z] = \{(X, \lambda_{Z,X} := \widehat{p}_{49}, \epsilon),\\
&Z \Rightarrow X \Rightarrow A, && \qquad\qquad (A, \lambda_{Z,A} := \widehat{p}_{49} \cdot \widehat{p}_{46}, X)\},\\
&N \Rightarrow Z, && targets[N] = \{(Z, \lambda_{N,Z} := \widehat{p}_{51}, \epsilon),\\
&N \Rightarrow U, && \qquad\qquad (U, \lambda_{N,U} := \widehat{p}_{52}, \epsilon),\\
&N \Rightarrow U \Rightarrow \circ, && \qquad\qquad (\circ, \lambda_{N,\circ} := \widehat{p}_{52} \cdot \widehat{p}_{53}, U),\\
&N \Rightarrow Z \Rightarrow X, && \qquad\qquad (X, \lambda_{N,X} := \widehat{p}_{51} \cdot \widehat{p}_{49}, Z),
\end{aligned}$$

$$N \Rightarrow Z \Rightarrow X \Rightarrow A, \qquad (A, \lambda_{N,A} := \widehat{p}_{51} \cdot \widehat{p}_{49} \cdot \widehat{p}_{46}, ZX)\},$$

$$U \Rightarrow \circ, \qquad targets[U] = \{(\circ, \lambda_{U,\circ} := \widehat{p}_{53}, \epsilon)\}.$$

Furthermore, the 22 production rules contained in R^1_{rnf} are now removed. This results in the following set $R^1_{\widehat{\mathcal{G}}_u} := R_{\widehat{\mathcal{G}}_u} \setminus R^1_{rnf}$ of 32 rules:

$$\begin{array}{llll}
\widehat{p}_1 : S' \to E, & & & \\
\widehat{p}_3 : E \to SC, & & & \\
\widehat{p}_5 : S \to TA, & & & \\
\widehat{p}_9 : C \to C\circ, & & & \\
\widehat{p}_{10} : A \to (L), & & & \\
\widehat{p}_{18} : G \to A\circ, & \widehat{p}_{19} : G \to AD, & \widehat{p}_{20} : G \to \circ A, & \widehat{p}_{21} : G \to DA, \\
\widehat{p}_{22} : D \to B\circ, & & & \\
\widehat{p}_{24} : B \to B\circ, & & & \\
\widehat{p}_{25} : F \to \circ\circ\circ, & \widehat{p}_{26} : F \to \circ\circ\circ\circ, & \widehat{p}_{27} : F \to \circ\circ\circ\circ H, & \\
\widehat{p}_{29} : H \to H\circ, & & & \\
\widehat{p}_{30} : P \to \circ A\circ, & \widehat{p}_{31} : P \to \circ A\circ\circ, & \widehat{p}_{32} : P \to \circ\circ A\circ, & \widehat{p}_{33} : P \to \circ\circ A\circ\circ, \\
\widehat{p}_{34} : Q \to \circ\circ O\circ\circ, & \widehat{p}_{35} : Q \to \circ\circ V\circ, & & \\
\widehat{p}_{36} : R \to \circ O\circ\circ, & \widehat{p}_{37} : R \to \circ\circ W\circ, & & \\
\widehat{p}_{38} : V \to JO, & & & \\
\widehat{p}_{39} : W \to JA, & & & \\
\widehat{p}_{40} : O \to AK, & & & \\
\widehat{p}_{42} : J \to J\circ, & & & \\
\widehat{p}_{44} : K \to K\circ, & & & \\
\widehat{p}_{45} : M \to XY, & & & \\
\widehat{p}_{47} : X \to UA, & & & \\
\widehat{p}_{50} : Z \to XN, & & & \\
\widehat{p}_{54} : U \to U\circ. & & &
\end{array}$$

Additionally, in step 2, for each chain a new intermediate symbol and a new production are introduced. Thus, according to the 32 chains gathered in step 1, we here obtain the following set R^2_{rnf} of 32 new production rules:

$$\begin{array}{llll}
1 : E^{S,\epsilon} \to S, & 1 : E^{A,S} \to A, & & \\
1 : S^{A,\epsilon} \to A, & & & \\
1 : T^{E,\epsilon} \to E, & 1 : T^{C,\epsilon} \to C, & 1 : T^{\circ,C} \to \circ, & \\
1 : T^{S,E} \to S, & 1 : T^{A,ES} \to A, & & \\
1 : C^{\circ,\epsilon} \to \circ, & & & \\
1 : L^{A,\epsilon} \to A, & 1 : L^{M,\epsilon} \to M, & 1 : L^{P,\epsilon} \to P, & 1 : L^{Q,\epsilon} \to Q, \\
1 : L^{R,\epsilon} \to R, & 1 : L^{F,\epsilon} \to F, & 1 : L^{G,\epsilon} \to G, & \\
1 : B^{\circ,\epsilon} \to \circ, & & & \\
1 : H^{\circ,\epsilon} \to \circ, & & & \\
1 : J^{\circ,\epsilon} \to \circ, & & & \\
1 : K^{\circ,\epsilon} \to \circ, & & & \\
1 : X^{A,\epsilon} \to A, & & & \\
1 : Y^{Z,\epsilon} \to Z, & 1 : Y^{X,Z} \to X, & 1 : Y^{A,ZX} \to A, & \\
1 : Z^{X,\epsilon} \to X, & & &
\end{array}$$

$$1 : Z^{A,X} \to A,$$
$$1 : N^{Z,\epsilon} \to Z, \quad 1 : N^{U,\epsilon} \to U, \quad 1 : N^{\circ,U} \to \circ,$$
$$1 : N^{X,Z} \to X, \quad 1 : N^{A,ZX} \to A,$$
$$1 : U^{\circ,\epsilon} \to \circ.$$

In step 3, for each occurrence of a nonterminal symbol in the conclusion of a production and each chain starting with this nonterminal symbol, we have to add a new production with the corresponding new intermediate symbol instead of the considered one. Thus, in step 3, the remaining 32 production rules from $R^1_{\widehat{\mathcal{G}}_u} = R_{\widehat{\mathcal{G}}_u} \setminus R^1_{rnf}$ are transformed (according to R^2_{rnf}) into the following set $R^2_{\widehat{\mathcal{G}}_u}$ of 79 new rules:

$$\widehat{p}_1 : S' \to E, \qquad \widehat{p}_1 \cdot \lambda_{E,S} : S' \to E^{S,\epsilon},$$
$$\widehat{p}_1 \cdot \lambda_{E,A} : S' \to E^{A,S},$$
$$\widehat{p}_3 : E \to SC, \qquad \widehat{p}_3 \cdot \lambda_{S,A} : E \to S^{A,\epsilon}C,$$
$$\widehat{p}_3 \cdot \lambda_{C,\circ} : E \to SC^{\circ,\epsilon}, \qquad \widehat{p}_3 \cdot \lambda_{S,A} \cdot \lambda_{C,\circ} : E \to S^{A,\epsilon}C^{\circ,\epsilon},$$
$$\widehat{p}_5 : S \to TA, \qquad \widehat{p}_5 \cdot \lambda_{T,E} : S \to T^{E,\epsilon}A,$$
$$\widehat{p}_5 \cdot \lambda_{T,C} : S \to T^{C,\epsilon}A, \qquad \widehat{p}_5 \cdot \lambda_{T,\circ} : S \to T^{\circ,C}A,$$
$$\widehat{p}_5 \cdot \lambda_{T,S} : S \to T^{S,E}A, \qquad \widehat{p}_5 \cdot \lambda_{T,A} : S \to T^{A,ES}A,$$
$$\widehat{p}_9 : C \to C\circ, \qquad \widehat{p}_9 \cdot \lambda_{C,\circ} : C \to C^{\circ,\epsilon}\circ,$$
$$\widehat{p}_{10} : A \to (L), \qquad \widehat{p}_{10} \cdot \lambda_{L,A} : A \to (L^{A,\epsilon}),$$
$$\widehat{p}_{10} \cdot \lambda_{L,M} : A \to (L^{M,\epsilon}), \qquad \widehat{p}_{10} \cdot \lambda_{L,P} : A \to (L^{P,\epsilon}),$$
$$\widehat{p}_{10} \cdot \lambda_{L,Q} : A \to (L^{Q,\epsilon}), \qquad \widehat{p}_{10} \cdot \lambda_{L,R} : A \to (L^{R,\epsilon}),$$
$$\widehat{p}_{10} \cdot \lambda_{L,F} : A \to (L^{F,\epsilon}), \qquad \widehat{p}_{10} \cdot \lambda_{L,G} : A \to (L^{G,\epsilon}),$$
$$\widehat{p}_{18} : G \to A\circ, \qquad \widehat{p}_{19} : G \to AD,$$
$$\widehat{p}_{20} : G \to \circ A, \qquad \widehat{p}_{21} : G \to DA,$$
$$\widehat{p}_{22} : D \to B\circ, \qquad \widehat{p}_{22} \cdot \lambda_{B,\circ} : D \to B^{\circ,\epsilon}\circ,$$
$$\widehat{p}_{24} : B \to B\circ, \qquad \widehat{p}_{24} \cdot \lambda_{B,\circ} : B \to B^{\circ,\epsilon}\circ,$$
$$\widehat{p}_{25} : F \to \circ\circ\circ, \qquad \widehat{p}_{26} : F \to \circ\circ\circ\circ,$$
$$\widehat{p}_{27} : F \to \circ\circ\circ\circ H, \qquad \widehat{p}_{27} \cdot \lambda_{H,\circ} : F \to \circ\circ\circ\circ H^{\circ,\epsilon},$$
$$\widehat{p}_{29} : H \to H\circ, \qquad \widehat{p}_{29} \cdot \lambda_{H,\circ} : H \to H^{\circ,\epsilon}\circ,$$
$$\widehat{p}_{30} : P \to \circ A\circ, \qquad \widehat{p}_{31} : P \to \circ A\circ\circ,$$
$$\widehat{p}_{32} : P \to \circ\circ A\circ, \qquad \widehat{p}_{33} : P \to \circ\circ A\circ\circ,$$
$$\widehat{p}_{34} : Q \to \circ\circ O\circ\circ, \qquad \widehat{p}_{35} : Q \to \circ\circ V\circ,$$
$$\widehat{p}_{36} : R \to \circ O\circ\circ, \qquad \widehat{p}_{37} : R \to \circ\circ W\circ,$$
$$\widehat{p}_{38} : V \to JO, \qquad \widehat{p}_{38} \cdot \lambda_{J,\circ} : V \to J^{\circ,\epsilon}O,$$
$$\widehat{p}_{39} : W \to JA, \qquad \widehat{p}_{39} \cdot \lambda_{J,\circ} : W \to J^{\circ,\epsilon}A,$$
$$\widehat{p}_{40} : O \to AK, \qquad \widehat{p}_{40} \cdot \lambda_{K,\circ} : O \to AK^{\circ,\epsilon},$$
$$\widehat{p}_{42} : J \to J\circ, \qquad \widehat{p}_{42} \cdot \lambda_{J,\circ} : J \to J^{\circ,\epsilon}\circ,$$
$$\widehat{p}_{44} : K \to K\circ, \qquad \widehat{p}_{44} \cdot \lambda_{K,\circ} : K \to K^{\circ,\epsilon}\circ,$$
$$\widehat{p}_{45} : M \to XY, \qquad \widehat{p}_{45} \cdot \lambda_{Y,Z} : M \to XY^{Z,\epsilon},$$
$$\widehat{p}_{45} \cdot \lambda_{Y,X} : M \to XY^{X,Z}, \qquad \widehat{p}_{45} \cdot \lambda_{Y,A} : M \to XY^{A,ZX},$$
$$\widehat{p}_{45} \cdot \lambda_{X,A} : M \to X^{A,\epsilon}Y, \qquad \widehat{p}_{45} \cdot \lambda_{X,A} \cdot \lambda_{Y,Z} : M \to X^{A,\epsilon}Y^{Z,\epsilon},$$
$$\widehat{p}_{45} \cdot \lambda_{X,A} \cdot \lambda_{Y,X} : M \to X^{A,\epsilon}Y^{X,Z}, \qquad \widehat{p}_{45} \cdot \lambda_{X,A} \cdot \lambda_{Y,A} : M \to X^{A,\epsilon}Y^{A,ZX},$$
$$\widehat{p}_{47} : X \to UA, \qquad \widehat{p}_{47} \cdot \lambda_{U,\circ} : X \to U^{\circ,\epsilon}A,$$
$$\widehat{p}_{50} : Z \to XN, \qquad \widehat{p}_{50} \cdot \lambda_{N,Z} : Z \to XN^{Z,\epsilon},$$
$$\widehat{p}_{50} \cdot \lambda_{N,U} : Z \to XN^{U,\epsilon}, \qquad \widehat{p}_{50} \cdot \lambda_{N,\circ} : Z \to XN^{\circ,U},$$
$$\widehat{p}_{50} \cdot \lambda_{N,X} : Z \to XN^{X,Z}, \qquad \widehat{p}_{50} \cdot \lambda_{N,A} : Z \to XN^{A,ZX},$$

$$\begin{aligned}
\widehat{p}_{50} \cdot \lambda_{X,A} &: Z \to X^{A,\epsilon} N, & \widehat{p}_{50} \cdot \lambda_{X,A} \cdot \lambda_{N,Z} &: Z \to X^{A,\epsilon} N^{Z,\epsilon},\\
\widehat{p}_{50} \cdot \lambda_{X,A} \cdot \lambda_{N,U} &: Z \to X^{A,\epsilon} N^{U,\epsilon}, & \widehat{p}_{50} \cdot \lambda_{X,A} \cdot \lambda_{N,\circ} &: Z \to X^{A,\epsilon} N^{\circ,U},\\
\widehat{p}_{50} \cdot \lambda_{X,A} \cdot \lambda_{N,X} &: Z \to X^{A,\epsilon} N^{X,Z}, & \widehat{p}_{50} \cdot \lambda_{X,A} \cdot \lambda_{N,A} &: Z \to X^{A,\epsilon} N^{A,ZX},\\
\widehat{p}_{54} &: U \to U\circ, & \widehat{p}_{54} \cdot \lambda_{U,\circ} &: U \to U^{\circ,\epsilon}\circ.
\end{aligned}$$

In step 4, we must delete all intermediate symbols that no longer occur as premise. Obviously, intermediate symbols no longer occurring as premise of a production are

$$T, L, N, Y.$$

We easily observe that the productions that contain at least one of these four intermediate symbols in the conclusion and thus have to be removed are exactly the following ones:

$$\begin{aligned}
\widehat{p}_5 &: S \to TA,\\
\widehat{p}_{10} &: A \to (L),\\
\widehat{p}_{45} &: M \to XY, & \widehat{p}_{45} \cdot \lambda_{X,A} &: M \to X^{A,\epsilon} Y,\\
\widehat{p}_{50} &: Z \to XN, & \widehat{p}_{50} \cdot \lambda_{X,A} &: Z \to X^{A,\epsilon} N.
\end{aligned}$$

Consequently, after the removal of these six rules from $R^2_{\widehat{\mathcal{G}}_u}$, there still remain 73 new production rules.

Finally, in step 5, we must make sure that the conclusion of all productions with premise S' (axiom of $\widehat{\mathcal{G}}_u$ that we started with) does not have a length greater than 1. However, since there is only one production with premise S' in our start grammar $\widehat{\mathcal{G}}_u$ and the conclusion of this production has size 1, there is nothing to do. Thus, the resulting new grammar is given by:

Definition A.2.1 *The WCFG $\widehat{\mathcal{G}}^*_u$ generating exactly the language $\mathcal{L}$ is given by*

$$\widehat{\mathcal{G}}^*_u = (I_{\widehat{\mathcal{G}}^*_u} \cup I'_{\widehat{\mathcal{G}}^*_u}, \Sigma_{\widehat{\mathcal{G}}^*_u}, R_{\widehat{\mathcal{G}}^*_u} \cup R'_{\widehat{\mathcal{G}}^*_u}, S'),$$

where

$$\begin{aligned}
I_{\widehat{\mathcal{G}}^*_u} &= \{S', E, S, C, A, G, D, B, F, H, P, Q, R, V, W, O, J, K, M, X, Z, U\},\\
I'_{\widehat{\mathcal{G}}^*_u} &= \{E^{S,\epsilon}, E^{A,S}, S^{A,\epsilon},\\
&\quad T^{E,\epsilon}, T^{C,\epsilon}, T^{\circ,C}, T^{S,E}, T^{A,ES}, C^{\circ,\epsilon},\\
&\quad L^{A,\epsilon}, L^{M,\epsilon}, L^{P,\epsilon}, L^{Q,\epsilon}, L^{R,\epsilon}, L^{F,\epsilon}, L^{G,\epsilon},\\
&\quad B^{\circ,\epsilon}, H^{\circ,\epsilon}, J^{\circ,\epsilon}, K^{\circ,\epsilon},\\
&\quad X^{A,\epsilon}, Y^{Z,\epsilon}, Y^{X,Z}, Y^{A,ZX}, Z^{X,\epsilon}, Z^{A,X},\\
&\quad N^{Z,\epsilon}, N^{U,\epsilon}, N^{\circ,U}, N^{X,Z}, N^{A,ZX}, U^{\circ,\epsilon}\},
\end{aligned}$$

$\Sigma_{\widehat{\mathcal{G}}^*_u} = \{(,), \circ\}$ *and* $R_{\widehat{\mathcal{G}}^*_u}$ *contains exactly the following rules:*

$$\begin{aligned}
&\lambda_1 : S' \to E, && \lambda_2 : S' \to E^{S,\epsilon}, && \lambda_3 : S' \to E^{A,S},\\
&\lambda_4 : E \to SC, && \lambda_5 : E \to S^{A,\epsilon} C, && \lambda_6 : E \to SC^{\circ,\epsilon}, && \lambda_7 : E \to S^{A,\epsilon} C^{\circ,\epsilon},\\
&\lambda_8 : S \to T^{E,\epsilon} A, && \lambda_9 : S \to T^{C,\epsilon} A, && \lambda_{10} : S \to T^{\circ,C} A,\\
&\lambda_{11} : S \to T^{S,E} A, && \lambda_{12} : S \to T^{A,ES} A,\\
&\lambda_{13} : C \to C\circ, && \lambda_{14} : C \to C^{\circ,\epsilon}\circ,\\
&\lambda_{15} : A \to (L^{A,\epsilon}), && \lambda_{16} : A \to (L^{M,\epsilon}), && \lambda_{17} : A \to (L^{P,\epsilon}), && \lambda_{18} : A \to (L^{Q,\epsilon}),\\
&\lambda_{19} : A \to (L^{R,\epsilon}), && \lambda_{20} : A \to (L^{F,\epsilon}), && \lambda_{21} : A \to (L^{G,\epsilon}),\\
&\lambda_{22} : G \to A\circ, && \lambda_{23} : G \to AD, && \lambda_{24} : G \to \circ A, && \lambda_{25} : G \to DA,
\end{aligned}$$

$\lambda_{26} : D \to B\circ$, $\lambda_{27} : D \to B^{\circ,\epsilon}\circ$,
$\lambda_{28} : B \to B\circ$, $\lambda_{29} : B \to B^{\circ,\epsilon}\circ$,
$\lambda_{30} : F \to \circ\circ\circ$, $\lambda_{31} : F \to \circ\circ\circ\circ$, $\lambda_{32} : F \to \circ\circ\circ\circ H$, $\lambda_{33} : F \to \circ\circ\circ\circ H^{\circ,\epsilon}$,
$\lambda_{34} : H \to H\circ$, $\lambda_{35} : H \to H^{\circ,\epsilon}\circ$,
$\lambda_{36} : P \to \circ A\circ$, $\lambda_{37} : P \to \circ A\circ\circ$, $\lambda_{38} : P \to \circ\circ A\circ$, $\lambda_{39} : P \to \circ\circ A\circ\circ$,
$\lambda_{40} : Q \to \circ\circ O\circ\circ$, $\lambda_{41} : Q \to \circ\circ V\circ$,
$\lambda_{42} : R \to \circ O\circ\circ$, $\lambda_{43} : R \to \circ\circ W\circ$,
$\lambda_{44} : V \to JO$, $\lambda_{45} : V \to J^{\circ,\epsilon}O$,
$\lambda_{46} : W \to JA$, $\lambda_{47} : W \to J^{\circ,\epsilon}A$,
$\lambda_{48} : O \to AK$, $\lambda_{49} : O \to AK^{\circ,\epsilon}$,
$\lambda_{50} : J \to J\circ$, $\lambda_{51} : J \to J^{\circ,\epsilon}\circ$,
$\lambda_{52} : K \to K\circ$, $\lambda_{53} : K \to K^{\circ,\epsilon}\circ$,
$\lambda_{54} : M \to XY^{Z,\epsilon}$, $\lambda_{55} : M \to XY^{X,Z}$, $\lambda_{56} : M \to XY^{A,ZX}$,
$\lambda_{57} : M \to X^{A,\epsilon}Y^{Z,\epsilon}$, $\lambda_{58} : M \to X^{A,\epsilon}Y^{X,Z}$, $\lambda_{59} : M \to X^{A,\epsilon}Y^{A,ZX}$,
$\lambda_{60} : X \to UA$, $\lambda_{61} : X \to U^{\circ,\epsilon}A$,
$\lambda_{62} : Z \to XN^{Z,\epsilon}$, $\lambda_{63} : Z \to XN^{U,\epsilon}$, $\lambda_{64} : Z \to XN^{\circ,U}$,
$\lambda_{65} : Z \to XN^{X,Z}$, $\lambda_{66} : Z \to XN^{A,ZX}$,
$\lambda_{67} : Z \to X^{A,\epsilon}N^{Z,\epsilon}$, $\lambda_{68} : Z \to X^{A,\epsilon}N^{U,\epsilon}$, $\lambda_{69} : Z \to X^{A,\epsilon}N^{\circ,U}$,
$\lambda_{70} : Z \to X^{A,\epsilon}N^{X,Z}$, $\lambda_{71} : Z \to X^{A,\epsilon}N^{A,ZX}$,
$\lambda_{72} : U \to U\circ$, $\lambda_{73} : U \to U^{\circ,\epsilon}\circ$,

whereas $R'_{\widehat{\mathcal{G}}^*_u}$ *contains exactly the following rules:*

$\lambda_{74} : E^{S,\epsilon} \to S$, $\lambda_{75} : E^{A,S} \to A$,
$\lambda_{76} : S^{A,\epsilon} \to A$,
$\lambda_{77} : T^{E,\epsilon} \to E$, $\lambda_{78} : T^{C,\epsilon} \to C$, $\lambda_{79} : T^{\circ,C} \to \circ$,
$\lambda_{80} : T^{S,E} \to S$, $\lambda_{81} : T^{A,ES} \to A$,
$\lambda_{82} : C^{\circ,\epsilon} \to \circ$,
$\lambda_{83} : L^{A,\epsilon} \to A$, $\lambda_{84} : L^{M,\epsilon} \to M$, $\lambda_{85} : L^{P,\epsilon} \to P$, $\lambda_{86} : L^{Q,\epsilon} \to Q$,
$\lambda_{87} : L^{R,\epsilon} \to R$, $\lambda_{88} : L^{F,\epsilon} \to F$, $\lambda_{89} : L^{G,\epsilon} \to G$,
$\lambda_{90} : B^{\circ,\epsilon} \to \circ$,
$\lambda_{91} : H^{\circ,\epsilon} \to \circ$,
$\lambda_{92} : J^{\circ,\epsilon} \to \circ$,
$\lambda_{93} : K^{\circ,\epsilon} \to \circ$,
$\lambda_{94} : X^{A,\epsilon} \to A$,
$\lambda_{95} : Y^{Z,\epsilon} \to Z$, $\lambda_{96} : Y^{X,Z} \to X$, $\lambda_{97} : Y^{A,ZX} \to A$,
$\lambda_{98} : Z^{X,\epsilon} \to X$, $\lambda_{99} : Z^{A,X} \to A$,
$\lambda_{100} : N^{Z,\epsilon} \to Z$, $\lambda_{101} : N^{U,\epsilon} \to U$, $\lambda_{102} : N^{\circ,U} \to \circ$,
$\lambda_{103} : N^{X,Z} \to X$, $\lambda_{104} : N^{A,ZX} \to A$,
$\lambda_{105} : U^{\circ,\epsilon} \to \circ$.

A.3. Reweighting the Production Rules

Now, the weights of the 73 production rules given in the subset of productions $R_{\widehat{\mathcal{G}}^*_u}$ have to be reweighted. In order to achieve this goal, we first have to compute the two common denominators s and c, where s is the common denominator of the weights of productions with

Nonterminal Nt	Weights of Rules with Premise Nt			
S′	$\lambda_1 := 1.0000$,	$\lambda_2 := 0.0212$,	$\lambda_3 := 0.0003$,	
E	$\lambda_4 := 0.9788$,	$\lambda_5 := 0.0134$,	$\lambda_6 := 0.0944$,	$\lambda_7 := 0.0013$,
S	$\lambda_8 := 0.8559$,	$\lambda_9 := 0.1304$,	$\lambda_{10} := 0.0126$,	$\lambda_{11} := 0.0181$,
	$\lambda_{12} := 0.0002$,			
C	$\lambda_{13} := 0.9036$,	$\lambda_{14} := 0.0871$,		
A	$\lambda_{15} := 0.7630$,	$\lambda_{16} := 0.0402$,	$\lambda_{17} := 0.0186$,	$\lambda_{18} := 0.0367$,
	$\lambda_{19} := 0.0072$,	$\lambda_{20} := 0.0858$,	$\lambda_{21} := 0.0484$,	
G	$\lambda_{22} := 0.3038$,	$\lambda_{23} := 0.1884$,	$\lambda_{24} := 0.3081$,	$\lambda_{25} := 0.1996$,
D	$\lambda_{26} := 1.0000$,	$\lambda_{27} := 0.3896$,		
B	$\lambda_{28} := 0.6104$,	$\lambda_{29} := 0.2378$,		
F	$\lambda_{30} := 0.0575$,	$\lambda_{31} := 0.3409$,	$\lambda_{32} := 0.6016$,	$\lambda_{33} := 0.1211$,
H	$\lambda_{34} := 0.7987$,	$\lambda_{35} := 0.1608$,		
P	$\lambda_{36} := 0.1085$,	$\lambda_{37} := 0.2144$,	$\lambda_{38} := 0.2011$,	$\lambda_{39} := 0.4760$,
Q	$\lambda_{40} := 0.1713$,	$\lambda_{41} := 0.8287$,		
R	$\lambda_{42} := 0.4150$,	$\lambda_{43} := 0.5850$,		
V	$\lambda_{44} := 1.0000$,	$\lambda_{45} := 0.3243$,		
W	$\lambda_{46} := 1.0000$,	$\lambda_{47} := 0.3243$,		
O	$\lambda_{48} := 1.0000$,	$\lambda_{49} := 0.2928$,		
J	$\lambda_{50} := 0.6757$,	$\lambda_{51} := 0.2191$,		
K	$\lambda_{52} := 0.7072$,	$\lambda_{53} := 0.2071$,		
M	$\lambda_{54} := 1.0000$,	$\lambda_{55} := 0.0510$,	$\lambda_{56} := 0.0036$,	$\lambda_{57} := 0.0712$,
	$\lambda_{58} := 0.0036$,	$\lambda_{59} := 0.0003$,		
X	$\lambda_{60} := 0.9288$,	$\lambda_{61} := 0.1968$,		
Z	$\lambda_{62} := 0.4211$,	$\lambda_{63} := 0.5279$,	$\lambda_{64} := 0.1119$,	$\lambda_{65} := 0.0215$,
	$\lambda_{66} := 0.0015$,	$\lambda_{67} := 0.0300$,	$\lambda_{68} := 0.0376$,	$\lambda_{69} := 0.0080$,
	$\lambda_{70} := 0.0015$,	$\lambda_{71} := 0.0001$,		
U	$\lambda_{72} := 0.7881$,	$\lambda_{73} := 0.1670$.		

Table A.1.: **Rounded weights for $\widehat{\mathcal{G}}_u^*$.** Table reports floating point approximations of the probabilities (weights) λ_i, $1 \leqslant i \leqslant 73$, for the production rules of the grammar $\widehat{\mathcal{G}}_u^*$ (rounded to four decimal places). Note that for $i \in \{74, \ldots, 105\}$, $\lambda_i := 1$ holds.

premise S' (that is, of productions number 1 to 3), and c is the common denominator of the weights of the remaining productions (that is, of productions number 4 to 73) of $R_{\widehat{\mathcal{G}}_u^*}$. Using the rounded probabilities (weights) for the production rules of $\widehat{\mathcal{G}}_u^*$ as given in Table A.1, we immediately find the smallest common denominators to be $s = 10,000$ and $c = 10,000$.

The desired new weights for the considered set of productions $R_{\widehat{\mathcal{G}}_u^*}$ are then computed by multiplying the old weights of productions with source S' by s, and by multiplying the old weights of productions $A \to \alpha$, $A \neq S'$ (and $A \in I_{\widehat{\mathcal{G}}_u^*}$), by $c^{|\alpha|-1}$. Formally, for the reweighted set of productions $R_{\widehat{\mathcal{G}}_u^*}$, we get the following weights:

$$\mu_i := \lambda_i \cdot s, \text{ for } i \in \{1, 2, 3\}$$

and

$$\mu_i := \lambda_i \cdot c^{|\alpha_i|-1}, \text{ where } \lambda_i : A_i \to \alpha_i, \text{ for } i \in \{4, \ldots, 73\}.$$

The resulting integer weights can be found in Table A.2.

Nonterminal Nt	Integer weights of Rules with Premise Nt	
S′	$\mu_1 := 10000$,	$\mu_2 := 212$,
	$\mu_3 := 3$,	
E	$\mu_4 := 9788$,	$\mu_5 := 134$,
	$\mu_6 := 944$,	$\mu_7 := 13$,
S	$\mu_8 := 8559$,	$\mu_9 := 1304$,
	$\mu_{10} := 126$,	$\mu_{11} := 181$,
	$\mu_{12} := 2$,	
C	$\mu_{13} := 9036$,	$\mu_{14} := 871$,
A	$\mu_{15} := 76300000$,	$\mu_{16} := 4020000$,
	$\mu_{17} := 1860000$,	$\mu_{18} := 3670000$,
	$\mu_{19} := 720000$,	$\mu_{20} := 8580000$,
	$\mu_{21} := 4840000$,	
G	$\mu_{22} := 3038$,	$\mu_{23} := 1884$,
	$\mu_{24} := 3081$,	$\mu_{25} := 1996$,
D	$\mu_{26} := 10000$,	$\mu_{27} := 3896$,
B	$\mu_{28} := 6104$,	$\mu_{29} := 2378$,
F	$\mu_{30} := 5750000$,	$\mu_{31} := 340900000000$,
	$\mu_{32} := 6016000000000000$,	$\mu_{33} := 1211000000000000$,
H	$\mu_{34} := 7987$,	$\mu_{35} := 1608$,
P	$\mu_{36} := 10850000$,	$\mu_{37} := 214400000000$,
	$\mu_{38} := 201100000000$,	$\mu_{39} := 4760000000000000$,
Q	$\mu_{40} := 1713000000000000$,	$\mu_{41} := 828700000000$,
R	$\mu_{42} := 415000000000$,	$\mu_{43} := 585000000000$,
V	$\mu_{44} := 10000$,	$\mu_{45} := 3243$,
W	$\mu_{46} := 10000$,	$\mu_{47} := 3243$,
O	$\mu_{48} := 10000$,	$\mu_{49} := 2928$,
J	$\mu_{50} := 6757$,	$\mu_{51} := 2191$,
K	$\mu_{52} := 7072$,	$\mu_{53} := 2071$,
M	$\mu_{54} := 10000$,	$\mu_{55} := 510$,
	$\mu_{56} := 36$,	$\mu_{57} := 712$,
	$\mu_{58} := 36$,	$\mu_{59} := 3$,
X	$\mu_{60} := 9288$,	$\mu_{61} := 1968$,
Z	$\mu_{62} := 4211$,	$\mu_{63} := 5279$,
	$\mu_{64} := 1119$,	$\mu_{65} := 215$,
	$\mu_{66} := 15$,	$\mu_{67} := 300$,
	$\mu_{68} := 376$,	$\mu_{69} := 80$,
	$\mu_{70} := 15$,	$\mu_{71} := 1$,
U	$\mu_{72} := 7881$,	$\mu_{73} := 1670$.

Table A.2.: **Integer weights for $\widehat{\mathcal{G}}_u^*$.** Table reports the integer weights μ_i, $1 \leqslant i \leqslant 73$, for the production rules of the grammar $\widehat{\mathcal{G}}_u^*$. Note that for $i \in \{74, \ldots, 105\}$, $\mu_i := 1$ holds.

A.4. Transforming Reweighted Grammar into Admissible Specification

Given the reweighted grammar $\widehat{\mathcal{G}}_u^*$, we immediately obtain the following admissible specification of the corresponding combinatorial classes (note that this specification has already been simplified by removing classes that are only duplicates of others):

$$
\begin{array}{lll}
\mathcal{E}_1 = \mathcal{S} \times \mathcal{C}, & \mathcal{E}_2 = \mathcal{A} \times \mathcal{C}, & \\
\mathcal{E}_3 = \mathcal{S} \times \alpha_\circ, & \mathcal{E}_4 = \mathcal{A} \times \alpha_\circ, & \\
\mathcal{S}_1 = \mathcal{E} \times \mathcal{A}, & \mathcal{S}_2 = \mathcal{C} \times \mathcal{A}, & \mathcal{S}_3 = \alpha_\circ \times \mathcal{A}, \\
\mathcal{S}_4 = \mathcal{S} \times \mathcal{A}, & \mathcal{S}_5 = \mathcal{A} \times \mathcal{A}, & \\
\mathcal{C}_1 = \mathcal{C} \times \alpha_\circ, & \mathcal{C}_2 = \alpha_\circ \times \alpha_\circ, & \\
\mathcal{A}_1 = \alpha_(\times \mathcal{A} \times \alpha_), & \mathcal{A}_2 = \alpha_(\times \mathcal{M} \times \alpha_), & \mathcal{A}_3 = \alpha_(\times \mathcal{P} \times \alpha_), \\
\mathcal{A}_4 = \alpha_(\times \mathcal{Q} \times \alpha_), & \mathcal{A}_5 = \alpha_(\times \mathcal{R} \times \alpha_), & \mathcal{A}_6 = \alpha_(\times \mathcal{F} \times \alpha_), \\
\mathcal{A}_7 = \alpha_(\times \mathcal{G} \times \alpha_), & & \\
\mathcal{G}_1 = \mathcal{A} \times \alpha_\circ, & \mathcal{G}_2 = \mathcal{A} \times \mathcal{D}, & \\
\mathcal{G}_3 = \alpha_\circ \times \mathcal{A}, & \mathcal{G}_4 = \mathcal{D} \times \mathcal{A}, & \\
\mathcal{D}_1 = \mathcal{B} \times \alpha_\circ, & \mathcal{D}_2 = \alpha_\circ \times \alpha_\circ, & \\
\mathcal{B}_1 = \mathcal{B} \times \alpha_\circ, & \mathcal{B}_2 = \alpha_\circ \times \alpha_\circ, & \\
\mathcal{F}_1 = \alpha_\circ \times \alpha_\circ \times \alpha_\circ, & \mathcal{F}_2 = \alpha_\circ \times \alpha_\circ \times \alpha_\circ \times \alpha_\circ, & \mathcal{F}_3 = \alpha_\circ \times \alpha_\circ \times \alpha_\circ \times \alpha_\circ \times \mathcal{H}, \\
\mathcal{F}_4 = \alpha_\circ \times \alpha_\circ \times \alpha_\circ \times \alpha_\circ \times \alpha_\circ, & & \\
\mathcal{H}_1 = \mathcal{H} \times \alpha_\circ, & \mathcal{H}_2 = \alpha_\circ \times \alpha_\circ, & \\
\mathcal{P}_1 = \alpha_\circ \times \mathcal{A} \times \alpha_\circ, & \mathcal{P}_2 = \alpha_\circ \times \mathcal{A} \times \alpha_\circ \times \alpha_\circ, & \mathcal{P}_3 = \alpha_\circ \times \alpha_\circ \times \mathcal{A} \times \alpha_\circ, \\
\mathcal{P}_4 = \alpha_\circ \times \alpha_\circ \times \mathcal{A} \times \alpha_\circ \times \alpha_\circ, & & \\
\mathcal{Q}_1 = \alpha_\circ \times \alpha_\circ \times \mathcal{O} \times \alpha_\circ \times \alpha_\circ, & \mathcal{Q}_2 = \alpha_\circ \times \alpha_\circ \times \mathcal{V} \times \alpha_\circ, & \\
\mathcal{R}_1 = \alpha_\circ \times \mathcal{O} \times \alpha_\circ \times \alpha_\circ, & \mathcal{R}_2 = \alpha_\circ \times \alpha_\circ \times \mathcal{W} \times \alpha_\circ, & \\
\mathcal{V}_1 = \mathcal{J} \times \mathcal{O}, & \mathcal{V}_2 = \alpha_\circ \times \mathcal{O}, & \\
\mathcal{W}_1 = \mathcal{J} \times \mathcal{A}, & \mathcal{W}_2 = \alpha_\circ \times \mathcal{A}, & \\
\mathcal{O}_1 = \mathcal{A} \times \mathcal{K}, & \mathcal{O}_2 = \mathcal{A} \times \alpha_\circ, & \\
\mathcal{J}_1 = \mathcal{J} \times \alpha_\circ, & \mathcal{J}_2 = \alpha_\circ \times \alpha_\circ, & \\
\mathcal{K}_1 = \mathcal{K} \times \alpha_\circ, & \mathcal{K}_2 = \alpha_\circ \times \alpha_\circ, & \\
\mathcal{M}_1 = \mathcal{X} \times \mathcal{Z}, & \mathcal{M}_2 = \mathcal{X} \times \mathcal{X}, & \mathcal{M}_3 = \mathcal{X} \times \mathcal{A}, \\
\mathcal{M}_4 = \mathcal{A} \times \mathcal{Z}, & \mathcal{M}_5 = \mathcal{A} \times \mathcal{X}, & \mathcal{M}_6 = \mathcal{A} \times \mathcal{A}, \\
\mathcal{X}_1 = \mathcal{U} \times \mathcal{A}, & \mathcal{X}_2 = \alpha_\circ \times \mathcal{A}, & \\
\mathcal{Z}_1 = \mathcal{X} \times \mathcal{Z}, & \mathcal{Z}_2 = \mathcal{X} \times \mathcal{U}, & \mathcal{Z}_3 = \mathcal{X} \times \alpha_\circ, \\
\mathcal{Z}_4 = \mathcal{X} \times \mathcal{X}, & \mathcal{Z}_5 = \mathcal{X} \times \mathcal{A}, & \\
\mathcal{Z}_6 = \mathcal{A} \times \mathcal{Z}, & \mathcal{Z}_7 = \mathcal{A} \times \mathcal{U}, & \mathcal{Z}_8 = \mathcal{A} \times \alpha_\circ, \\
\mathcal{Z}_9 = \mathcal{A} \times \mathcal{X}, & \mathcal{Z}_{10} = \mathcal{A} \times \mathcal{A}, & \\
\mathcal{U}_1 = \mathcal{U} \times \alpha_\circ, & \mathcal{U}_2 = \alpha_\circ \times \alpha_\circ, &
\end{array}
$$

$$
\begin{aligned}
\mathcal{S}' &= \mu_1 \cdot \mathcal{E} + \mu_2 \cdot \mathcal{S} + \mu_3 \cdot \mathcal{A}, \\
\mathcal{E} &= \mu_4 \cdot \mathcal{E}_1 + \mu_5 \cdot \mathcal{E}_2 + \mu_6 \cdot \mathcal{E}_3 + \mu_7 \cdot \mathcal{E}_4, \\
\mathcal{S} &= \mu_8 \cdot \mathcal{S}_1 + \mu_9 \cdot \mathcal{S}_2 + \mu_{10} \cdot \mathcal{S}_3 + \mu_{11} \cdot \mathcal{S}_4 + \mu_{12} \cdot \mathcal{S}_5, \\
\mathcal{C} &= \mu_{13} \cdot \mathcal{C}_1 + \mu_{14} \cdot \mathcal{C}_2, \\
\mathcal{A} &= \mu_{15} \cdot \mathcal{A}_1 + \mu_{16} \cdot \mathcal{A}_2 + \mu_{17} \cdot \mathcal{A}_3 + \mu_{18} \cdot \mathcal{A}_4 + \mu_{19} \cdot \mathcal{A}_5 + \mu_{20} \cdot \mathcal{A}_6 + \mu_{21} \cdot \mathcal{A}_7, \\
\mathcal{G} &= \mu_{22} \cdot \mathcal{G}_1 + \mu_{23} \cdot \mathcal{G}_2 + \mu_{24} \cdot \mathcal{G}_3 + \mu_{25} \cdot \mathcal{G}_4, \\
\mathcal{D} &= \mu_{26} \cdot \mathcal{D}_1 + \mu_{27} \cdot \mathcal{D}_2,
\end{aligned}
$$

$$
\begin{aligned}
\mathcal{B} &= \mu_{28} \cdot \mathcal{B}_1 + \mu_{29} \cdot \mathcal{B}_2, \\
\mathcal{F} &= \mu_{30} \cdot \mathcal{F}_1 + \mu_{31} \cdot \mathcal{F}_2 + \mu_{32} \cdot \mathcal{F}_3 + \mu_{33} \cdot \mathcal{F}_4, \\
\mathcal{H} &= \mu_{34} \cdot \mathcal{H}_1 + \mu_{35} \cdot \mathcal{H}_2, \\
\mathcal{P} &= \mu_{36} \cdot \mathcal{P}_1 + \mu_{37} \cdot \mathcal{P}_2 + \mu_{38} \cdot \mathcal{P}_3 + \mu_{39} \cdot \mathcal{P}_4, \\
\mathcal{Q} &= \mu_{40} \cdot \mathcal{Q}_1 + \mu_{41} \cdot \mathcal{Q}_2, \\
\mathcal{R} &= \mu_{42} \cdot \mathcal{R}_1 + \mu_{43} \cdot \mathcal{R}_2, \\
\mathcal{V} &= \mu_{44} \cdot \mathcal{V}_1 + \mu_{45} \cdot \mathcal{V}_2, \\
\mathcal{W} &= \mu_{46} \cdot \mathcal{W}_1 + \mu_{47} \cdot \mathcal{W}_2, \\
\mathcal{O} &= \mu_{48} \cdot \mathcal{O}_1 + \mu_{49} \cdot \mathcal{O}_2, \\
\mathcal{J} &= \mu_{50} \cdot \mathcal{J}_1 + \mu_{51} \cdot \mathcal{J}_2, \\
\mathcal{K} &= \mu_{52} \cdot \mathcal{K}_1 + \mu_{53} \cdot \mathcal{K}_2, \\
\mathcal{M} &= \mu_{54} \cdot \mathcal{M}_1 + \mu_{55} \cdot \mathcal{M}_2 + \mu_{56} \cdot \mathcal{M}_3 + \mu_{57} \cdot \mathcal{M}_4 + \mu_{58} \cdot \mathcal{M}_5 + \mu_{59} \cdot \mathcal{M}_6, \\
\mathcal{X} &= \mu_{60} \cdot \mathcal{X}_1 + \mu_{61} \cdot \mathcal{X}_2, \\
\mathcal{Z} &= \mu_{62} \cdot \mathcal{Z}_1 + \mu_{63} \cdot \mathcal{Z}_2 + \mu_{64} \cdot \mathcal{Z}_3 + \mu_{65} \cdot \mathcal{Z}_4 + \mu_{66} \cdot \mathcal{Z}_5 + \\
&\qquad \mu_{67} \cdot \mathcal{Z}_6 + \mu_{68} \cdot \mathcal{Z}_7 + \mu_{69} \cdot \mathcal{Z}_8 + \mu_{70} \cdot \mathcal{Z}_9 + \mu_{71} \cdot \mathcal{Z}_{10}, \\
\mathcal{U} &= \mu_{72} \cdot \mathcal{U}_1 + \mu_{73} \cdot \mathcal{U}_2.
\end{aligned}
$$

Now, this (simplified) specification can easily be transformed into the following recursive form for the function `size`:

$$\texttt{size}(\mathcal{I},n) := \begin{cases} \mu_1 \cdot \texttt{size}(\mathcal{E},n) + \mu_2 \cdot \texttt{size}(\mathcal{S},n) & \\ \quad +\mu_3 \cdot \texttt{size}(\mathcal{A},n) & \mathcal{I} = \mathcal{S}', \\ \texttt{size}_E(\mathcal{I},n) & \mathcal{I} \in \{\mathcal{E}_i \mid 1 \leqslant i \leqslant 4\} \text{ or } \mathcal{I} = \mathcal{E}, \\ \texttt{size}_S(\mathcal{I},n) & \mathcal{I} \in \{\mathcal{S}_i \mid 1 \leqslant i \leqslant 5\} \text{ or } \mathcal{I} = \mathcal{S}, \\ \texttt{size}_C(\mathcal{I},n) & \mathcal{I} \in \{\mathcal{C}_i \mid 1 \leqslant i \leqslant 2\} \text{ or } \mathcal{I} = \mathcal{C}, \\ \texttt{size}_A(\mathcal{I},n) & \mathcal{I} \in \{\mathcal{A}_i \mid 1 \leqslant i \leqslant 7\} \text{ or } \mathcal{I} = \mathcal{A}, \\ \texttt{size}_G(\mathcal{I},n) & \mathcal{I} \in \{\mathcal{G}_i \mid 1 \leqslant i \leqslant 4\} \text{ or } \mathcal{I} = \mathcal{G}, \\ \texttt{size}_D(\mathcal{I},n) & \mathcal{I} \in \{\mathcal{D}_i \mid 1 \leqslant i \leqslant 2\} \text{ or } \mathcal{I} = \mathcal{D}, \\ \texttt{size}_B(\mathcal{I},n) & \mathcal{I} \in \{\mathcal{B}_i \mid 1 \leqslant i \leqslant 2\} \text{ or } \mathcal{I} = \mathcal{B}, \\ \texttt{size}_F(\mathcal{I},n) & \mathcal{I} \in \{\mathcal{F}_i \mid 1 \leqslant i \leqslant 4\} \text{ or } \mathcal{I} = \mathcal{F}, \\ \texttt{size}_H(\mathcal{I},n) & \mathcal{I} \in \{\mathcal{H}_i \mid 1 \leqslant i \leqslant 2\} \text{ or } \mathcal{I} = \mathcal{H}, \\ \texttt{size}_P(\mathcal{I},n) & \mathcal{I} \in \{\mathcal{P}_i \mid 1 \leqslant i \leqslant 4\} \text{ or } \mathcal{I} = \mathcal{P}, \\ \texttt{size}_Q(\mathcal{I},n) & \mathcal{I} \in \{\mathcal{Q}_i \mid 1 \leqslant i \leqslant 2\} \text{ or } \mathcal{I} = \mathcal{Q}, \\ \texttt{size}_R(\mathcal{I},n) & \mathcal{I} \in \{\mathcal{R}_i \mid 1 \leqslant i \leqslant 2\} \text{ or } \mathcal{I} = \mathcal{R}, \\ \texttt{size}_V(\mathcal{I},n) & \mathcal{I} \in \{\mathcal{V}_i \mid 1 \leqslant i \leqslant 2\} \text{ or } \mathcal{I} = \mathcal{V}, \\ \texttt{size}_W(\mathcal{I},n) & \mathcal{I} \in \{\mathcal{W}_i \mid 1 \leqslant i \leqslant 2\} \text{ or } \mathcal{I} = \mathcal{W}, \\ \texttt{size}_O(\mathcal{I},n) & \mathcal{I} \in \{\mathcal{O}_i \mid 1 \leqslant i \leqslant 2\} \text{ or } \mathcal{I} = \mathcal{O}, \\ \texttt{size}_J(\mathcal{I},n) & \mathcal{I} \in \{\mathcal{J}_i \mid 1 \leqslant i \leqslant 2\} \text{ or } \mathcal{I} = \mathcal{J}, \\ \texttt{size}_K(\mathcal{I},n) & \mathcal{I} \in \{\mathcal{K}_i \mid 1 \leqslant i \leqslant 2\} \text{ or } \mathcal{I} = \mathcal{K}, \\ \texttt{size}_M(\mathcal{I},n) & \mathcal{I} \in \{\mathcal{M}_i \mid 1 \leqslant i \leqslant 6\} \text{ or } \mathcal{I} = \mathcal{M}, \\ \texttt{size}_X(\mathcal{I},n) & \mathcal{I} \in \{\mathcal{X}_i \mid 1 \leqslant i \leqslant 2\} \text{ or } \mathcal{I} = \mathcal{X}, \\ \texttt{size}_Z(\mathcal{I},n) & \mathcal{I} \in \{\mathcal{Z}_i \mid 1 \leqslant i \leqslant 10\} \text{ or } \mathcal{I} = \mathcal{Z}, \\ \texttt{size}_U(\mathcal{I},n) & \mathcal{I} \in \{\mathcal{U}_i \mid 1 \leqslant i \leqslant 2\} \text{ or } \mathcal{I} = \mathcal{U}, \\ 0 & \text{else}, \end{cases}$$

where

$$\texttt{size}_E(\mathcal{I},n) := \begin{cases} \sum_{j=1}^{n-1} \texttt{size}(\mathcal{S},j) \cdot \texttt{size}(\mathcal{C},n-j) & \mathcal{I} = \mathcal{E}_1, \\ \sum_{j=1}^{n-1} \texttt{size}(\mathcal{A},j) \cdot \texttt{size}(\mathcal{C},n-j) & \mathcal{I} = \mathcal{E}_2, \\ \texttt{size}(\mathcal{S},n-1) & \mathcal{I} = \mathcal{E}_3, \\ \texttt{size}(\mathcal{A},n-1) & \mathcal{I} = \mathcal{E}_4, \\ \mu_4 \cdot \texttt{size}(\mathcal{E}_1,n) + \mu_5 \cdot \texttt{size}(\mathcal{E}_2,n) + \mu_6 \cdot \texttt{size}(\mathcal{E}_3,n) & \\ \quad +\mu_7 \cdot \texttt{size}(\mathcal{E}_4,n) & \mathcal{I} = \mathcal{E}, \\ 0 & \text{else}, \end{cases}$$

$$\texttt{size}_{\mathcal{S}}(\mathcal{I}, n) := \begin{cases} \sum_{j=1}^{n-1} \texttt{size}(\mathcal{E}, j) \cdot \texttt{size}(\mathcal{A}, n-j) & \mathcal{I} = \mathcal{S}_1, \\ \sum_{j=1}^{n-1} \texttt{size}(\mathcal{C}, j) \cdot \texttt{size}(\mathcal{A}, n-j) & \mathcal{I} = \mathcal{S}_2, \\ \texttt{size}(\mathcal{A}, n-1) & \mathcal{I} = \mathcal{S}_3, \\ \sum_{j=1}^{n-1} \texttt{size}(\mathcal{S}, j) \cdot \texttt{size}(\mathcal{A}, n-j) & \mathcal{I} = \mathcal{S}_4, \\ \sum_{j=1}^{n-1} \texttt{size}(\mathcal{A}, j) \cdot \texttt{size}(\mathcal{A}, n-j) & \mathcal{I} = \mathcal{S}_5, \\ \mu_8 \cdot \texttt{size}(\mathcal{S}_1, n) + \mu_9 \cdot \texttt{size}(\mathcal{S}_2, n) + \mu_{10} \cdot \texttt{size}(\mathcal{S}_3, n) & \\ +\mu_{11} \cdot \texttt{size}(\mathcal{S}_4, n) + \mu_{12} \cdot \texttt{size}(\mathcal{S}_5, n) & \mathcal{I} = \mathcal{S}, \\ 0 & \text{else}, \end{cases}$$

$$\texttt{size}_{\mathcal{C}}(\mathcal{I}, n) := \begin{cases} \texttt{size}(\mathcal{C}, n-1) & \mathcal{I} = \mathcal{C}_1, \\ 1 & \mathcal{I} = \mathcal{C}_2 \text{ and } n = 2, \\ \mu_{13} \cdot \texttt{size}(\mathcal{C}_1, n) + \mu_{14} \cdot \texttt{size}(\mathcal{C}_2, n) & \mathcal{I} = \mathcal{C}, \\ 0 & \text{else}, \end{cases}$$

$$\texttt{size}_{\mathcal{A}}(\mathcal{I}, n) := \begin{cases} \texttt{size}(\mathcal{A}, n-2) & \mathcal{I} = \mathcal{A}_1, \\ \texttt{size}(\mathcal{M}, n-2) & \mathcal{I} = \mathcal{A}_2, \\ \texttt{size}(\mathcal{P}, n-2) & \mathcal{I} = \mathcal{A}_3, \\ \texttt{size}(\mathcal{Q}, n-2) & \mathcal{I} = \mathcal{A}_4, \\ \texttt{size}(\mathcal{R}, n-2) & \mathcal{I} = \mathcal{A}_5, \\ \texttt{size}(\mathcal{F}, n-2) & \mathcal{I} = \mathcal{A}_6, \\ \texttt{size}(\mathcal{G}, n-2) & \mathcal{I} = \mathcal{A}_7, \\ \mu_{15} \cdot \texttt{size}(\mathcal{A}_1, n) + \mu_{16} \cdot \texttt{size}(\mathcal{A}_2, n) + \mu_{17} \cdot \texttt{size}(\mathcal{A}_3, n) & \\ +\mu_{18} \cdot \texttt{size}(\mathcal{A}_4, n) + \mu_{19} \cdot \texttt{size}(\mathcal{A}_5, n) + \mu_{20} \cdot \texttt{size}(\mathcal{A}_6, n) & \\ +\mu_{21} \cdot \texttt{size}(\mathcal{A}_7, n) & \mathcal{I} = \mathcal{A}, \\ 0 & \text{else}, \end{cases}$$

$$\texttt{size}_{\mathcal{G}}(\mathcal{I}, n) := \begin{cases} \texttt{size}(\mathcal{A}, n-1) & \mathcal{I} = \mathcal{G}_1, \\ \sum_{j=1}^{n-1} \texttt{size}(\mathcal{A}, j) \cdot \texttt{size}(\mathcal{D}, n-j) & \mathcal{I} = \mathcal{G}_2, \\ \texttt{size}(\mathcal{A}, n-1) & \mathcal{I} = \mathcal{G}_3, \\ \sum_{j=1}^{n-1} \texttt{size}(\mathcal{D}, j) \cdot \texttt{size}(\mathcal{A}, n-j) & \mathcal{I} = \mathcal{G}_4, \\ \mu_{22} \cdot \texttt{size}(\mathcal{G}_1, n) + \mu_{23} \cdot \texttt{size}(\mathcal{G}_2, n) + \mu_{24} \cdot \texttt{size}(\mathcal{G}_3, n) & \\ +\mu_{25} \cdot \texttt{size}(\mathcal{G}_4, n) & \mathcal{I} = \mathcal{G}, \\ 0 & \text{else}, \end{cases}$$

$$\texttt{size}_{\mathcal{D}}(\mathcal{I}, n) := \begin{cases} \texttt{size}(\mathcal{B}, n-1) & \mathcal{I} = \mathcal{D}_1, \\ 1 & \mathcal{I} = \mathcal{D}_2 \text{ and } n = 2, \\ \mu_{26} \cdot \texttt{size}(\mathcal{D}_1, n) + \mu_{27} \cdot \texttt{size}(\mathcal{D}_2, n) & \mathcal{I} = \mathcal{D}, \\ 0 & \text{else}, \end{cases}$$

$$\texttt{size}_{\mathrm{B}}(\mathcal{I}, n) := \begin{cases} \texttt{size}(\mathcal{B}, n-1) & \mathcal{I} = \mathcal{B}_1, \\ 1 & \mathcal{I} = \mathcal{B}_2 \text{ and } n = 2, \\ \mu_{28} \cdot \texttt{size}(\mathcal{B}_1, n) + \mu_{29} \cdot \texttt{size}(\mathcal{B}_2, n) & \mathcal{I} = \mathcal{B}, \\ 0 & \text{else}, \end{cases}$$

$$\texttt{size}_{\mathrm{F}}(\mathcal{I}, n) := \begin{cases} 1 & \mathcal{I} = \mathcal{F}_1 \text{ and } n = 3, \\ 1 & \mathcal{I} = \mathcal{F}_2 \text{ and } n = 4, \\ \texttt{size}(\mathcal{H}, n-4) & \mathcal{I} = \mathcal{F}_3, \\ 1 & \mathcal{I} = \mathcal{F}_4 \text{ and } n = 5, \\ \mu_{30} \cdot \texttt{size}(\mathcal{F}_1, n) + \mu_{31} \cdot \texttt{size}(\mathcal{F}_2, n) + \mu_{32} \cdot \texttt{size}(\mathcal{F}_3, n) & \\ +\mu_{33} \cdot \texttt{size}(\mathcal{F}_4, n) & \mathcal{I} = \mathcal{F}, \\ 0 & \text{else}, \end{cases}$$

$$\texttt{size}_{\mathrm{H}}(\mathcal{I}, n) := \begin{cases} \texttt{size}(\mathcal{H}, n-1) & \mathcal{I} = \mathcal{H}_1, \\ 1 & \mathcal{I} = \mathcal{H}_2 \text{ and } n = 2, \\ \mu_{34} \cdot \texttt{size}(\mathcal{H}_1, n) + \mu_{35} \cdot \texttt{size}(\mathcal{H}_2, n) & \mathcal{I} = \mathcal{H}, \\ 0 & \text{else}, \end{cases}$$

$$\texttt{size}_{\mathrm{P}}(\mathcal{I}, n) := \begin{cases} \texttt{size}(\mathcal{A}, n-2) & \mathcal{I} = \mathcal{P}_1, \\ \texttt{size}(\mathcal{A}, n-3) & \mathcal{I} = \mathcal{P}_2, \\ \texttt{size}(\mathcal{A}, n-3) & \mathcal{I} = \mathcal{P}_3, \\ \texttt{size}(\mathcal{A}, n-4) & \mathcal{I} = \mathcal{P}_4, \\ \mu_{36} \cdot \texttt{size}(\mathcal{P}_1, n) + \mu_{37} \cdot \texttt{size}(\mathcal{P}_2, n) + \mu_{38} \cdot \texttt{size}(\mathcal{P}_3, n) & \\ +\mu_{39} \cdot \texttt{size}(\mathcal{P}_4, n) & \mathcal{I} = \mathcal{P}, \\ 0 & \text{else}, \end{cases}$$

$$\texttt{size}_{\mathrm{Q}}(\mathcal{I}, n) := \begin{cases} \texttt{size}(\mathcal{O}, n-4) & \mathcal{I} = \mathcal{Q}_1, \\ \texttt{size}(\mathcal{V}, n-3) & \mathcal{I} = \mathcal{Q}_2, \\ \mu_{40} \cdot \texttt{size}(\mathcal{Q}_1, n) + \mu_{41} \cdot \texttt{size}(\mathcal{Q}_2, n) & \mathcal{I} = \mathcal{Q}, \\ 0 & \text{else}, \end{cases}$$

$$\texttt{size}_{\mathrm{R}}(\mathcal{I}, n) := \begin{cases} \texttt{size}(\mathcal{O}, n-3) & \mathcal{I} = \mathcal{R}_1, \\ \texttt{size}(\mathcal{W}, n-3) & \mathcal{I} = \mathcal{R}_2, \\ \mu_{42} \cdot \texttt{size}(\mathcal{R}_1, n) + \mu_{43} \cdot \texttt{size}(\mathcal{R}_2, n) & \mathcal{I} = \mathcal{R}, \\ 0 & \text{else}, \end{cases}$$

$$\texttt{size}_{\mathcal{V}}(\mathcal{I}, n) := \begin{cases} \sum_{j=1}^{n-1} \texttt{size}(\mathcal{J}, j) \cdot \texttt{size}(\mathcal{O}, n-j) & \mathcal{I} = \mathcal{V}_1, \\ \texttt{size}(\mathcal{O}, n-1) & \mathcal{I} = \mathcal{V}_2, \\ \mu_{44} \cdot \texttt{size}(\mathcal{V}_1, n) + \mu_{45} \cdot \texttt{size}(\mathcal{V}_2, n) & \mathcal{I} = \mathcal{V}, \\ 0 & \text{else}, \end{cases}$$

$$\texttt{size}_{\mathcal{W}}(\mathcal{I}, n) := \begin{cases} \sum_{j=1}^{n-1} \texttt{size}(\mathcal{J}, j) \cdot \texttt{size}(\mathcal{A}, n-j) & \mathcal{I} = \mathcal{W}_1, \\ \texttt{size}(\mathcal{A}, n-1) & \mathcal{I} = \mathcal{W}_2, \\ \mu_{46} \cdot \texttt{size}(\mathcal{W}_1, n) + \mu_{47} \cdot \texttt{size}(\mathcal{W}_2, n) & \mathcal{I} = \mathcal{W}, \\ 0 & \text{else}, \end{cases}$$

$$\texttt{size}_{\mathcal{O}}(\mathcal{I}, n) := \begin{cases} \sum_{j=1}^{n-1} \texttt{size}(\mathcal{A}, j) \cdot \texttt{size}(\mathcal{K}, n-j) & \mathcal{I} = \mathcal{O}_1, \\ \texttt{size}(\mathcal{A}, n-1) & \mathcal{I} = \mathcal{O}_2, \\ \mu_{48} \cdot \texttt{size}(\mathcal{O}_1, n) + \mu_{49} \cdot \texttt{size}(\mathcal{O}_2, n) & \mathcal{I} = \mathcal{O}, \\ 0 & \text{else}, \end{cases}$$

$$\texttt{size}_{\mathcal{J}}(\mathcal{I}, n) := \begin{cases} \texttt{size}(\mathcal{J}, n-1) & \mathcal{I} = \mathcal{J}_1, \\ 1 & \mathcal{I} = \mathcal{J}_2 \text{ and } n = 2, \\ \mu_{50} \cdot \texttt{size}(\mathcal{J}_1, n) + \mu_{51} \cdot \texttt{size}(\mathcal{J}_2, n) & \mathcal{I} = \mathcal{J}, \\ 0 & \text{else}, \end{cases}$$

$$\texttt{size}_{\mathcal{K}}(\mathcal{I}, n) := \begin{cases} \texttt{size}(\mathcal{K}, n-1) & \mathcal{I} = \mathcal{K}_1, \\ 1 & \mathcal{I} = \mathcal{K}_2 \text{ and } n = 2, \\ \mu_{52} \cdot \texttt{size}(\mathcal{K}_1, n) + \mu_{53} \cdot \texttt{size}(\mathcal{K}_2, n) & \mathcal{I} = \mathcal{K}, \\ 0 & \text{else}, \end{cases}$$

$$\texttt{size}_{\mathcal{M}}(\mathcal{I}, n) := \begin{cases} \sum_{j=1}^{n-1} \texttt{size}(\mathcal{X}, j) \cdot \texttt{size}(\mathcal{Z}, n-j) & \mathcal{I} = \mathcal{M}_1, \\ \sum_{j=1}^{n-1} \texttt{size}(\mathcal{X}, j) \cdot \texttt{size}(\mathcal{X}, n-j) & \mathcal{I} = \mathcal{M}_2, \\ \sum_{j=1}^{n-1} \texttt{size}(\mathcal{X}, j) \cdot \texttt{size}(\mathcal{A}, n-j) & \mathcal{I} = \mathcal{M}_3, \\ \sum_{j=1}^{n-1} \texttt{size}(\mathcal{A}, j) \cdot \texttt{size}(\mathcal{Z}, n-j) & \mathcal{I} = \mathcal{M}_4, \\ \sum_{j=1}^{n-1} \texttt{size}(\mathcal{A}, j) \cdot \texttt{size}(\mathcal{X}, n-j) & \mathcal{I} = \mathcal{M}_5, \\ \sum_{j=1}^{n-1} \texttt{size}(\mathcal{A}, j) \cdot \texttt{size}(\mathcal{A}, n-j) & \mathcal{I} = \mathcal{M}_6, \\ \mu_{54} \cdot \texttt{size}(\mathcal{M}_1, n) + \mu_{55} \cdot \texttt{size}(\mathcal{M}_2, n) + \mu_{56} \cdot \texttt{size}(\mathcal{M}_3, n) & \\ + \mu_{57} \cdot \texttt{size}(\mathcal{M}_4, n) + \mu_{58} \cdot \texttt{size}(\mathcal{M}_5, n) + \mu_{59} \cdot \texttt{size}(\mathcal{M}_6, n) & \mathcal{I} = \mathcal{M}, \\ 0 & \text{else}, \end{cases}$$

$$\texttt{size}_{\mathcal{X}}(\mathcal{I}, n) := \begin{cases} \sum_{j=1}^{n-1} \texttt{size}(\mathcal{U}, j) \cdot \texttt{size}(\mathcal{A}, n-j) & \mathcal{I} = \mathcal{X}_1, \\ \texttt{size}(\mathcal{A}, n-1) & \mathcal{I} = \mathcal{X}_2, \\ \mu_{60} \cdot \texttt{size}(\mathcal{X}_1, n) + \mu_{61} \cdot \texttt{size}(\mathcal{X}_2, n) & \mathcal{I} = \mathcal{X}, \\ 0 & \text{else}, \end{cases}$$

$$\texttt{size}_Z(\mathcal{I}, n) := \begin{cases} \sum_{j=1}^{n-1} \texttt{size}(\mathcal{X}, j) \cdot \texttt{size}(\mathcal{Z}, n-j) & \mathcal{I} = \mathcal{Z}_1, \\ \sum_{j=1}^{n-1} \texttt{size}(\mathcal{X}, j) \cdot \texttt{size}(\mathcal{U}, n-j) & \mathcal{I} = \mathcal{Z}_2, \\ \texttt{size}(\mathcal{X}, n-1) & \mathcal{I} = \mathcal{Z}_3, \\ \sum_{j=1}^{n-1} \texttt{size}(\mathcal{X}, j) \cdot \texttt{size}(\mathcal{X}, n-j) & \mathcal{I} = \mathcal{Z}_4, \\ \sum_{j=1}^{n-1} \texttt{size}(\mathcal{X}, j) \cdot \texttt{size}(\mathcal{A}, n-j) & \mathcal{I} = \mathcal{Z}_5, \\ \sum_{j=1}^{n-1} \texttt{size}(\mathcal{A}, j) \cdot \texttt{size}(\mathcal{Z}, n-j) & \mathcal{I} = \mathcal{Z}_6, \\ \sum_{j=1}^{n-1} \texttt{size}(\mathcal{A}, j) \cdot \texttt{size}(\mathcal{U}, n-j) & \mathcal{I} = \mathcal{Z}_7, \\ \texttt{size}(\mathcal{A}, n-1) & \mathcal{I} = \mathcal{Z}_8, \\ \sum_{j=1}^{n-1} \texttt{size}(\mathcal{A}, j) \cdot \texttt{size}(\mathcal{X}, n-j) & \mathcal{I} = \mathcal{Z}_9, \\ \sum_{j=1}^{n-1} \texttt{size}(\mathcal{A}, j) \cdot \texttt{size}(\mathcal{A}, n-j) & \mathcal{I} = \mathcal{Z}_{10}, \\ \mu_{62} \cdot \texttt{size}(\mathcal{Z}_1, n) + \mu_{63} \cdot \texttt{size}(\mathcal{Z}_2, n) + \mu_{64} \cdot \texttt{size}(\mathcal{Z}_3, n) & \\ +\mu_{65} \cdot \texttt{size}(\mathcal{Z}_4, n) + \mu_{66} \cdot \texttt{size}(\mathcal{Z}_5, n) + \mu_{67} \cdot \texttt{size}(\mathcal{Z}_6, n) & \\ +\mu_{68} \cdot \texttt{size}(\mathcal{Z}_7, n) + \mu_{69} \cdot \texttt{size}(\mathcal{Z}_8, n) + \mu_{70} \cdot \texttt{size}(\mathcal{Z}_9, n) & \\ +\mu_{71} \cdot \texttt{size}(\mathcal{Z}_{10}, n) & \mathcal{I} = \mathcal{Z}, \\ 0 & \text{else}, \end{cases}$$

$$\texttt{size}_U(\mathcal{I}, n) := \begin{cases} \texttt{size}(\mathcal{U}, n-1) & \mathcal{I} = \mathcal{U}_1, \\ 1 & \mathcal{I} = \mathcal{U}_2 \text{ and } n = 2, \\ \mu_{72} \cdot \texttt{size}(\mathcal{U}_1, n) + \mu_{73} \cdot \texttt{size}(\mathcal{U}_2, n) & \mathcal{I} = \mathcal{U}, \\ 0 & \text{else}. \end{cases}$$

From those recurrences, the desired algorithm can easily be constructed. As the complete presentation of this algorithm would be too comprehensive, we decided to omit it and instead refer to Algorithms 1 to 4 and 6 given in [WN10], since for the construction of our unranking algorithm, we had to use exactly these Algorithms as subroutines.

Chapter B

Tables and Figures Relating to Chapter 6

Approach	Parameters	Shape Level					
		0	1	2	3	4	5
PF	$\max_{bulge} = 30$	0.0633	0.1216	0.2071	0.2117	0.2639	0.3694
SCFG	$\min_{HL} = 1, \min_{hel} = 1$	0.2099	0.3699	0.5594	0.5594	0.5599	0.6302
	$\min_{HL} = 1, \min_{hel} = 2$	0.2187	0.3833	0.5830	0.5830	0.5835	0.6607
	$\min_{HL} = 3, \min_{hel} = 1$	0.2450	0.4448	0.6417	0.6417	0.6422	0.7356
	$\min_{HL} = 3, \min_{hel} = 2$	0.2409	0.4364	0.6399	0.6399	0.6403	0.7379

(a) CSP_{freq} (selection principle MF struct.).

Approach	Parameters	Shape Level					
		0	1	2	3	4	5
PF	$\max_{bulge} = 30$	0.0416	0.1049	0.1923	0.1960	0.2496	0.3559
SCFG	$\min_{HL} = 1, \min_{hel} = 1$	0.0555	0.2094	0.4193	0.4193	0.4207	0.4679
	$\min_{HL} = 1, \min_{hel} = 2$	0.0656	0.2446	0.4961	0.4961	0.4984	0.5613
	$\min_{HL} = 3, \min_{hel} = 1$	0.0772	0.2510	0.4928	0.4928	0.4942	0.5497
	$\min_{HL} = 3, \min_{hel} = 2$	0.1008	0.2917	0.5525	0.5525	0.5543	0.6241

(b) CSP_{freq} (selection principle MEA struct.).

Approach	Parameters	Shape Level					
		0	1	2	3	4	5
PF	$\max_{bulge} = 30$	0.0264	0.0800	0.1595	0.1627	0.1932	0.2677
SCFG	$\min_{HL} = 1, \min_{hel} = 1$	0.0374	0.1276	0.2973	0.2973	0.2978	0.3130
	$\min_{HL} = 1, \min_{hel} = 2$	0.0485	0.1623	0.3791	0.3791	0.3800	0.4097
	$\min_{HL} = 3, \min_{hel} = 1$	0.0536	0.1665	0.3773	0.3773	0.3778	0.4060
	$\min_{HL} = 3, \min_{hel} = 2$	0.0758	0.2150	0.4563	0.4563	0.4568	0.5003

(c) CSP_{freq} (selection principle Centroid).

Table B.1.: **Results related to shapes of selected predictions, obtained for our tRNA data.** They were computed by 10-fold cross-validation procedures, using sample size 1000.

Approach	Parameters	Shape Level					
		0	1	2	3	4	5
PF	$\max_{bulge} = 30$	0.5196	0.6740	0.8160	0.8239	0.8798	0.9556
SCFG	$\min_{HL} = 1, \min_{hel} = 1$	0.6838	0.9459	0.9903	0.9903	0.9908	0.9995
	$\min_{HL} = 1, \min_{hel} = 2$	0.6806	0.9006	0.9630	0.9635	0.9640	0.9991
	$\min_{HL} = 3, \min_{hel} = 1$	0.7148	0.9459	0.9875	0.9880	0.9885	0.9991
	$\min_{HL} = 3, \min_{hel} = 2$	0.7111	0.8997	0.9677	0.9681	0.9686	0.9995

(a) CSO_{freq}.

Approach	Parameters	Shape Level					
		0	1	2	3	4	5
PF	$\max_{bulge} = 30$	21.073	58.200	136.67	140.63	205.54	328.56
SCFG	$\min_{HL} = 1, \min_{hel} = 1$	16.202	98.357	327.26	327.27	327.51	418.80
	$\min_{HL} = 1, \min_{hel} = 2$	25.205	142.50	453.03	453.03	453.10	527.04
	$\min_{HL} = 3, \min_{hel} = 1$	24.883	130.04	392.78	392.79	393.05	494.79
	$\min_{HL} = 3, \min_{hel} = 2$	34.898	173.73	513.05	513.06	513.08	595.26

(b) CS_{num}.

Approach	Parameters	Shape Level					
		0	1	2	3	4	5
PF	$\max_{bulge} = 30$	355.32	130.22	81.796	33.125	22.585	4.8848
SCFG	$\min_{HL} = 1, \min_{hel} = 1$	802.27	244.52	60.504	60.030	59.916	28.764
	$\min_{HL} = 1, \min_{hel} = 2$	652.75	125.69	24.687	24.687	24.687	16.019
	$\min_{HL} = 3, \min_{hel} = 1$	752.71	208.65	48.257	47.797	47.691	21.838
	$\min_{HL} = 3, \min_{hel} = 2$	592.84	103.04	18.921	18.921	18.921	12.053

(c) DS_{num}.

Table B.2.: **Results related to shapes of sampled structures, obtained for our tRNA data.** They were computed by 10-fold cross-validation procedures, using sample size 1000.

Approach	Parameters	Shape Level					
		0	1	2	3	4	5
PF	$\max_{bulge} = 30$	0.0000	0.0009	0.0078	0.0513	0.0261	0.6353
SCFG	$\min_{HL} = 1, \min_{hel} = 1$	0.0000	0.0026	0.0052	0.0131	0.0357	0.7128
	$\min_{HL} = 1, \min_{hel} = 2$	0.0000	0.0052	0.0139	0.0331	0.0522	0.7502
	$\min_{HL} = 3, \min_{hel} = 1$	0.0000	0.0044	0.0113	0.0314	0.0766	0.7781
	$\min_{HL} = 3, \min_{hel} = 2$	0.0009	0.0096	0.0244	0.0609	0.1027	0.8207

(a) CSP_{freq} (selection principle MF struct.).

Approach	Parameters	Shape Level					
		0	1	2	3	4	5
PF	$\max_{bulge} = 30$	0.0000	0.0052	0.0139	0.0835	0.0696	0.6640
SCFG	$\min_{HL} = 1, \min_{hel} = 1$	0.0000	0.0000	0.0000	0.0000	0.0261	0.3820
	$\min_{HL} = 1, \min_{hel} = 2$	0.0000	0.0009	0.0009	0.0035	0.0566	0.4769
	$\min_{HL} = 3, \min_{hel} = 1$	0.0000	0.0000	0.0000	0.0009	0.0261	0.3977
	$\min_{HL} = 3, \min_{hel} = 2$	0.0000	0.0009	0.0009	0.0035	0.0557	0.5387

(b) CSP_{freq} (selection principle MEA struct.).

Approach	Parameters	Shape Level					
		0	1	2	3	4	5
PF	$\max_{bulge} = 30$	0.0000	0.0026	0.0104	0.0775	0.0731	0.7214
SCFG	$\min_{HL} = 1, \min_{hel} = 1$	0.0000	0.0000	0.0000	0.0000	0.0104	0.1097
	$\min_{HL} = 1, \min_{hel} = 2$	0.0000	0.0000	0.0000	0.0000	0.0148	0.1279
	$\min_{HL} = 3, \min_{hel} = 1$	0.0000	0.0000	0.0000	0.0000	0.0078	0.1236
	$\min_{HL} = 3, \min_{hel} = 2$	0.0000	0.0000	0.0000	0.0009	0.0139	0.1549

(c) CSP_{freq} (selection principle Centroid).

Table B.3.: **Results related to shapes of selected predictions, obtained for our 5S rRNA data.** They were computed by 10-fold cross-validation procedures, using sample size 1000.

Approach	Parameters	Shape Level					
		0	1	2	3	4	5
PF	$\max_{bulge} = 30$	0.0009	0.1662	0.3063	0.7580	0.6883	0.9817
SCFG	$\min_{HL} = 1, \min_{hel} = 1$	0.0000	0.2855	0.4526	0.9852	0.9974	1.0000
	$\min_{HL} = 1, \min_{hel} = 2$	0.0017	0.4135	0.5754	0.9861	0.9983	0.9991
	$\min_{HL} = 3, \min_{hel} = 1$	0.0000	0.3308	0.4883	0.9904	0.9974	1.0000
	$\min_{HL} = 3, \min_{hel} = 2$	0.0026	0.4509	0.6372	0.9904	0.9974	0.9991

(a) CSO_{freq}.

Approach	Parameters	Shape Level					
		0	1	2	3	4	5
PF	$\max_{bulge} = 30$	0.0009	0.7571	3.4207	36.641	30.288	600.35
SCFG	$\min_{HL} = 1, \min_{hel} = 1$	0.0000	0.5432	1.1811	20.640	51.834	573.72
	$\min_{HL} = 1, \min_{hel} = 2$	0.0017	1.1428	2.6615	32.051	64.332	608.06
	$\min_{HL} = 3, \min_{hel} = 1$	0.0000	0.6651	1.4309	22.983	54.635	569.80
	$\min_{HL} = 3, \min_{hel} = 2$	0.0026	1.3795	3.1949	36.673	71.080	609.58

(b) CS_{num}.

Approach	Parameters	Shape Level					
		0	1	2	3	4	5
PF	$\max_{bulge} = 30$	710.75	333.72	237.71	93.335	63.661	7.0951
SCFG	$\min_{HL} = 1, \min_{hel} = 1$	999.67	941.77	866.98	336.69	167.10	16.476
	$\min_{HL} = 1, \min_{hel} = 2$	999.18	884.49	764.79	249.02	129.35	14.198
	$\min_{HL} = 3, \min_{hel} = 1$	999.93	947.19	874.03	331.75	163.09	15.620
	$\min_{HL} = 3, \min_{hel} = 2$	999.68	885.81	762.67	239.28	123.91	13.558

(c) DS_{num}.

Table B.4.: **Results related to shapes of sampled structures, obtained for our 5S rRNA data.** They were computed by 10-fold cross-validation procedures, using sample size 1000.

Approach	Parameters	Shape Level					
		0	1	2	3	4	5
PF	$\max_{bulge} = 30$	0.0661	0.1255	0.1586	0.2050	0.2183	0.4834
SCFG	$\min_{HL} = 1, \min_{hel} = 1$	0.0530	0.0993	0.1191	0.1324	0.1589	0.3776
	$\min_{HL} = 1, \min_{hel} = 2$	0.0398	0.1193	0.1457	0.1656	0.1856	0.4106
	$\min_{HL} = 3, \min_{hel} = 1$	0.0530	0.1259	0.1390	0.1590	0.1789	0.4107
	$\min_{HL} = 3, \min_{hel} = 2$	0.0530	0.1258	0.1522	0.1788	0.1985	0.4240

(a) CSP_{freq} (selection principle MF struct.).

Approach	Parameters	Shape Level					
		0	1	2	3	4	5
PF	$\max_{bulge} = 30$	0.0660	0.1123	0.1453	0.1984	0.2051	0.4902
SCFG	$\min_{HL} = 1, \min_{hel} = 1$	0.0264	0.0927	0.0993	0.1125	0.1325	0.3778
	$\min_{HL} = 1, \min_{hel} = 2$	0.0264	0.1193	0.1391	0.1523	0.1789	0.4239
	$\min_{HL} = 3, \min_{hel} = 1$	0.0264	0.0927	0.0993	0.1125	0.1325	0.3777
	$\min_{HL} = 3, \min_{hel} = 2$	0.0197	0.1127	0.1391	0.1656	0.2055	0.4109

(b) CSP_{freq} (selection principle MEA struct.).

Approach	Parameters	Shape Level					
		0	1	2	3	4	5
PF	$\max_{bulge} = 30$	0.0793	0.1321	0.1653	0.1917	0.2449	0.5100
SCFG	$\min_{HL} = 1, \min_{hel} = 1$	0.0197	0.0861	0.1059	0.1190	0.1258	0.3181
	$\min_{HL} = 1, \min_{hel} = 2$	0.0197	0.0795	0.0926	0.1191	0.1192	0.3578
	$\min_{HL} = 3, \min_{hel} = 1$	0.0197	0.0795	0.0926	0.1125	0.1125	0.3181
	$\min_{HL} = 3, \min_{hel} = 2$	0.0197	0.0927	0.1125	0.1390	0.1391	0.3577

(c) CSP_{freq} (selection principle Centroid).

Table B.5.: **Results related to shapes of selected predictions, obtained for S-151Rfam database.** They were computed by two-fold cross-validation procedures, using sample size 1000.

Approach	Parameters	Shape Level					
		0	1	2	3	4	5
PF	$\max_{bulge} = 30$	0.3638	0.4433	0.4766	0.5231	0.6488	0.7947
SCFG	$\min_{HL} = 1, \min_{hel} = 1$	0.2520	0.5497	0.6095	0.6888	0.7683	0.9604
	$\min_{HL} = 1, \min_{hel} = 2$	0.2717	0.5630	0.6158	0.7284	0.8079	0.9605
	$\min_{HL} = 3, \min_{hel} = 1$	0.2518	0.5429	0.6093	0.7218	0.7815	0.9472
	$\min_{HL} = 3, \min_{hel} = 2$	0.2715	0.5564	0.6027	0.7087	0.7484	0.9604

(a) CSO_{freq}.

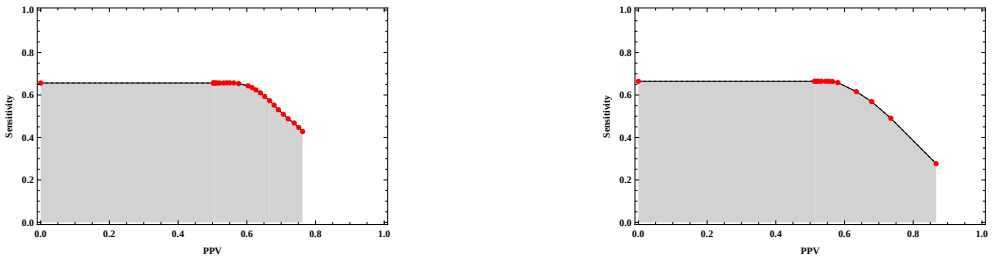
Approach	Parameters	Shape Level					
		0	1	2	3	4	5
PF	$\max_{bulge} = 30$	40.390	88.886	121.55	158.32	195.83	453.58
SCFG	$\min_{HL} = 1, \min_{hel} = 1$	10.743	47.281	63.587	97.088	121.64	362.44
	$\min_{HL} = 1, \min_{hel} = 2$	12.968	58.796	78.776	115.96	139.09	387.16
	$\min_{HL} = 3, \min_{hel} = 1$	12.468	51.569	67.603	104.67	125.50	365.84
	$\min_{HL} = 3, \min_{hel} = 2$	15.059	63.707	83.965	125.82	142.99	391.39

(b) CS_{num}.

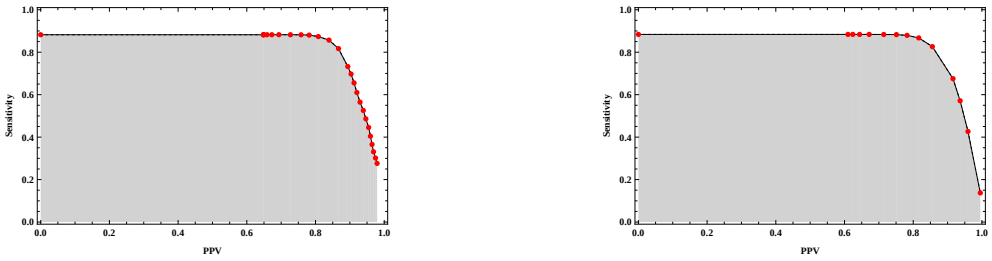
Approach	Parameters	Shape Level					
		0	1	2	3	4	5
PF	$\max_{bulge} = 30$	540.74	304.36	255.40	150.89	117.24	18.795
SCFG	$\min_{HL} = 1, \min_{hel} = 1$	892.14	600.39	526.36	368.49	322.88	99.601
	$\min_{HL} = 1, \min_{hel} = 2$	849.32	538.56	466.17	322.99	286.12	84.480
	$\min_{HL} = 3, \min_{hel} = 1$	888.89	588.97	516.66	358.72	315.25	94.603
	$\min_{HL} = 3, \min_{hel} = 2$	840.03	522.53	452.04	307.61	273.92	77.536

(c) DS_{num}.

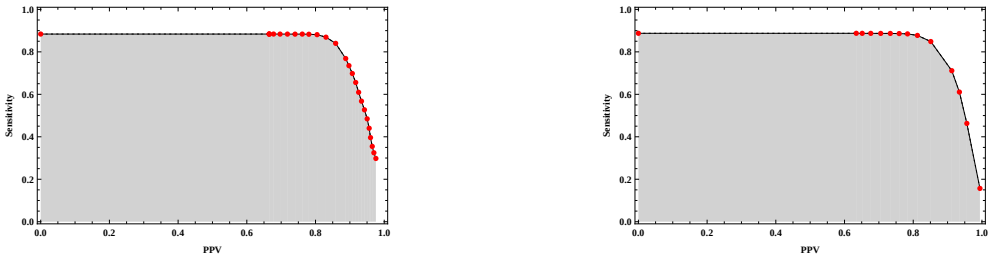
Table B.6.: **Results related to shapes of sampled structures, obtained for S-151Rfam database.** They were computed by two-fold cross-validation procedures, using sample size 1000.



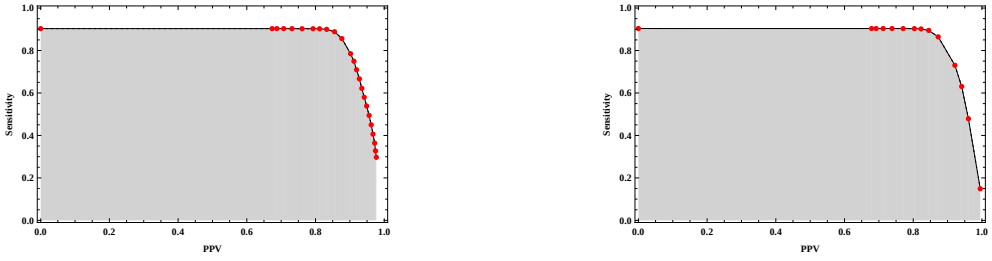
(a) PF approach (with parameter $\max_{\texttt{bulge}} = 30$).



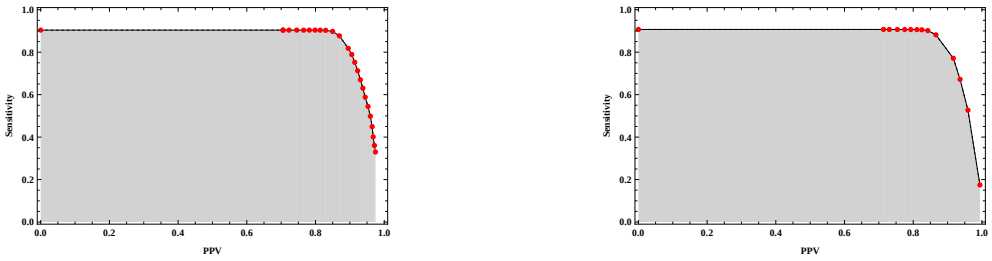
(b) SCFG approach (with least realistic parameter combination $\min_{\mathrm{HL}} = 1$ and $\min_{\mathrm{hel}} = 1$).



(c) SCFG approach (with parameter combination $\min_{\mathrm{HL}} = 1$ and $\min_{\mathrm{hel}} = 2$).

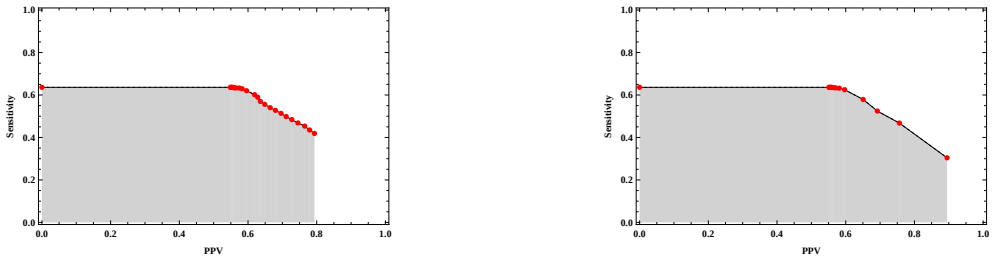


(d) SCFG approach (with parameter combination $\min_{\mathrm{HL}} = 3$ and $\min_{\mathrm{hel}} = 1$).

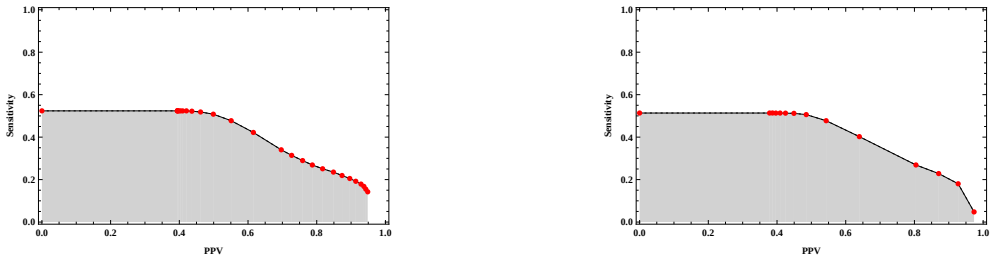


(e) SCFG approach (with most realistic parameter combination $\min_{\mathrm{HL}} = 3$ and $\min_{\mathrm{hel}} = 2$).

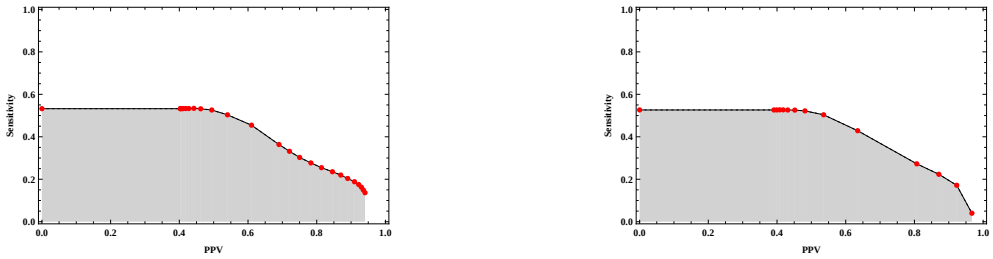
Figure B.1.: **Comparison of the (areas under) ROC curves obtained for our tRNA database.** Results were computed by 10-fold cross-validation procedures, using sample size 1000. For each considered sampling variant, the corresponding ROC curves are shown for prediction principle MEA structure (figure on the left) and centroid (figure on the right), respectively.



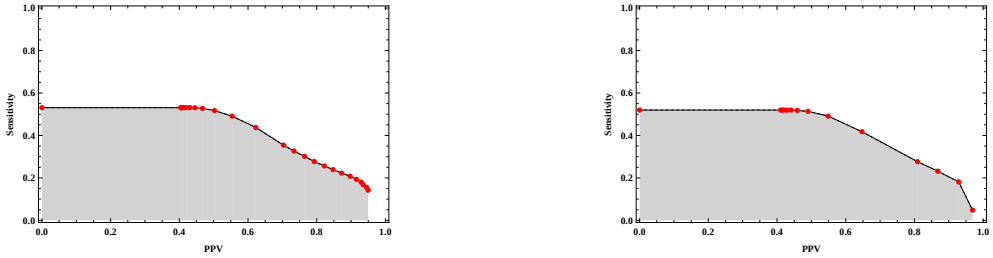
(a) PF approach (with parameter $\max_{\texttt{bulge}} = 30$).



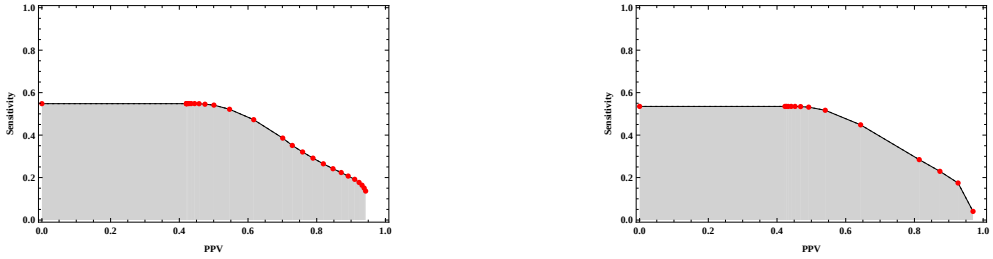
(b) SCFG approach (with least realistic parameter combination $\min_{\mathrm{HL}} = 1$ and $\min_{\mathrm{hel}} = 1$).



(c) SCFG approach (with parameter combination $\min_{\mathrm{HL}} = 1$ and $\min_{\mathrm{hel}} = 2$).

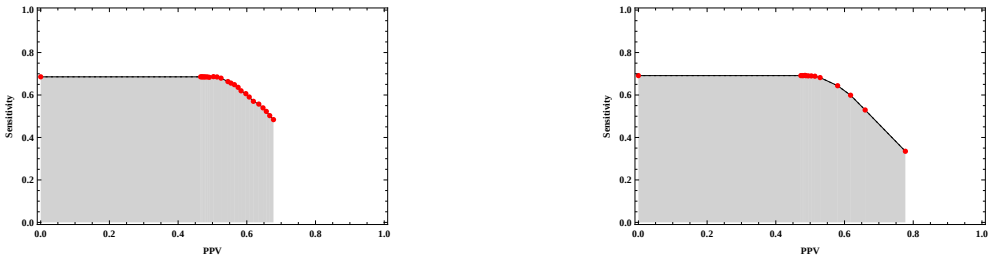


(d) SCFG approach (with parameter combination $\min_{\mathrm{HL}} = 3$ and $\min_{\mathrm{hel}} = 1$).

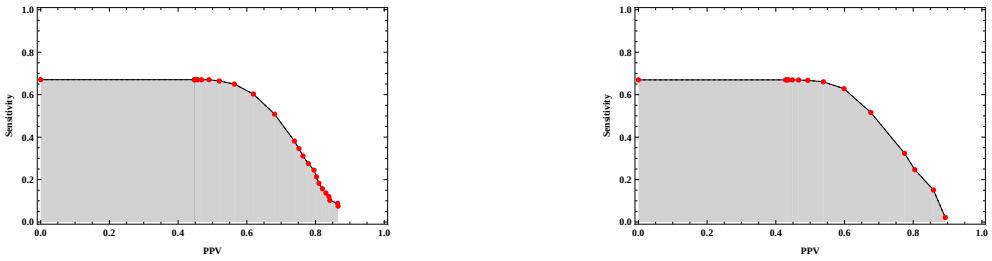


(e) SCFG approach (with most realistic parameter combination $\min_{\mathrm{HL}} = 3$ and $\min_{\mathrm{hel}} = 2$).

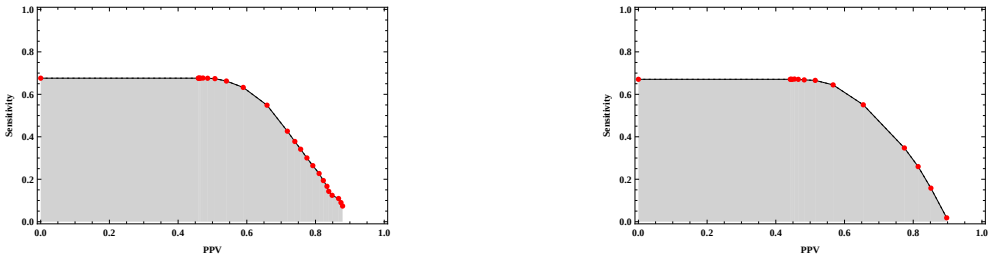
Figure B.2.: **Comparison of the (areas under) ROC curves obtained for our 5S rRNA database.** Results were computed by 10-fold cross-validation procedures, using sample size 1000. For each considered sampling variant, the corresponding ROC curves are shown for prediction principle MEA structure (figure on the left) and centroid (figure on the right), respectively.



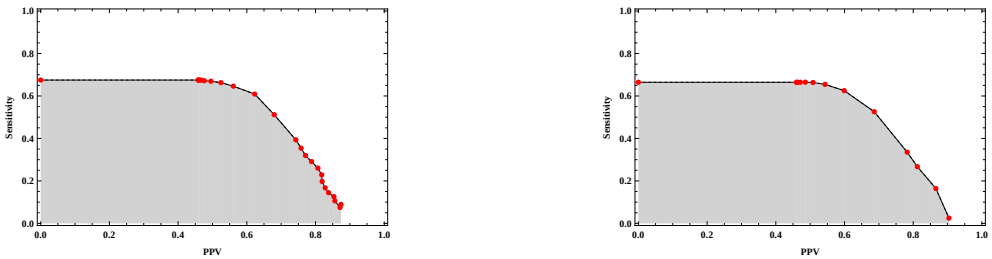
(a) PF approach (with parameter $\max_{\texttt{bulge}} = 30$).



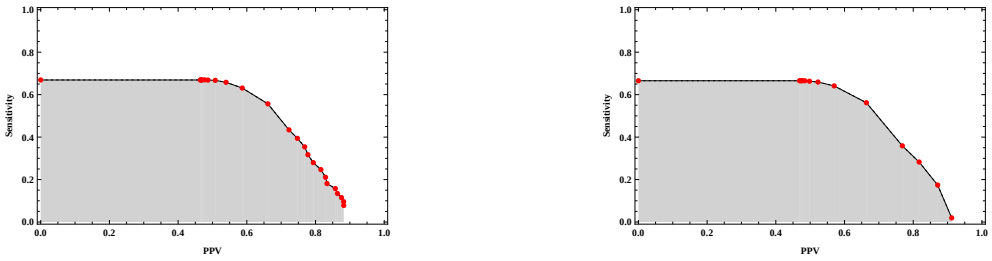
(b) SCFG approach (with least realistic parameter combination $\min_{\mathrm{HL}} = 1$ and $\min_{\mathrm{hel}} = 1$).



(c) SCFG approach (with parameter combination $\min_{\mathrm{HL}} = 1$ and $\min_{\mathrm{hel}} = 2$).



(d) SCFG approach (with parameter combination $\min_{\mathrm{HL}} = 3$ and $\min_{\mathrm{hel}} = 1$).



(e) SCFG approach (with most realistic parameter combination $\min_{\mathrm{HL}} = 3$ and $\min_{\mathrm{hel}} = 2$).

Figure B.3.: **Comparison of the (areas under) ROC curves obtained for the mixed S-151Rfam database.** Results were computed by two-fold cross-validation procedures, using the same folds as in [DWB06] and sample size 1000. For each considered sampling variant, the corresponding ROC curves are shown for prediction principle MEA structure (figure on the left) and centroid (figure on the right), respectively.

Chapter C

Tables and Figures Relating to Chapter 7

Approach	Parameters	Shape Level					
		0	1	2	3	4	5
PF	$\max_{bulge} = 30$	0.0633	0.1216	0.2071	0.2117	0.2639	0.3694
SCFG	$\min_{HL} = 1, \min_{hel} = 1$	0.2099	0.3699	0.5594	0.5594	0.5599	0.6302
	$\min_{HL} = 1, \min_{hel} = 2$	0.2187	0.3833	0.5830	0.5830	0.5835	0.6607
	$\min_{HL} = 3, \min_{hel} = 1$	0.2450	0.4448	0.6417	0.6417	0.6422	0.7356
	$\min_{HL} = 3, \min_{hel} = 2$	0.2409	0.4364	0.6399	0.6399	0.6403	0.7379
LSCFG	$\min_{HL} = 1, \min_{hel} = 1$	0.3278	0.4896	0.6565	0.6570	0.6570	0.7341
	$\min_{HL} = 1, \min_{hel} = 2$	0.2936	0.4018	0.6792	0.6792	0.6796	0.7642
	$\min_{HL} = 3, \min_{hel} = 1$	0.3440	0.5137	0.6805	0.6805	0.6810	0.7628
	$\min_{HL} = 3, \min_{hel} = 2$	0.2982	0.4124	0.6963	0.6963	0.6967	0.7873

(a) CSP_{freq} (selection principle MF struct.).

Approach	Parameters	Shape Level					
		0	1	2	3	4	5
PF	$\max_{bulge} = 30$	0.0416	0.1049	0.1923	0.1960	0.2496	0.3559
SCFG	$\min_{HL} = 1, \min_{hel} = 1$	0.0555	0.2094	0.4193	0.4193	0.4207	0.4679
	$\min_{HL} = 1, \min_{hel} = 2$	0.0656	0.2446	0.4961	0.4961	0.4984	0.5613
	$\min_{HL} = 3, \min_{hel} = 1$	0.0772	0.2510	0.4928	0.4928	0.4942	0.5497
	$\min_{HL} = 3, \min_{hel} = 2$	0.1008	0.2917	0.5525	0.5525	0.5543	0.6241
LSCFG	$\min_{HL} = 1, \min_{hel} = 1$	0.1854	0.3574	0.4919	0.4919	0.4919	0.5465
	$\min_{HL} = 1, \min_{hel} = 2$	0.1956	0.4013	0.6824	0.6824	0.6829	0.7712
	$\min_{HL} = 3, \min_{hel} = 1$	0.1951	0.3676	0.4979	0.4984	0.4979	0.5552
	$\min_{HL} = 3, \min_{hel} = 2$	0.2053	0.4115	0.6958	0.6958	0.6963	0.7869

(b) CSP_{freq} (selection principle MEA struct.).

Approach	Parameters	Shape Level					
		0	1	2	3	4	5
PF	$\max_{bulge} = 30$	0.0264	0.0800	0.1595	0.1627	0.1932	0.2677
SCFG	$\min_{HL} = 1, \min_{hel} = 1$	0.0374	0.1276	0.2973	0.2973	0.2978	0.3130
	$\min_{HL} = 1, \min_{hel} = 2$	0.0485	0.1623	0.3791	0.3791	0.3800	0.4097
	$\min_{HL} = 3, \min_{hel} = 1$	0.0536	0.1665	0.3773	0.3773	0.3778	0.4060
	$\min_{HL} = 3, \min_{hel} = 2$	0.0758	0.2150	0.4563	0.4563	0.4568	0.5003
LSCFG	$\min_{HL} = 1, \min_{hel} = 1$	0.1729	0.3158	0.4300	0.4300	0.4300	0.4762
	$\min_{HL} = 1, \min_{hel} = 2$	0.1877	0.3768	0.6362	0.6362	0.6366	0.7157
	$\min_{HL} = 3, \min_{hel} = 1$	0.1812	0.3199	0.4304	0.4304	0.4304	0.4780
	$\min_{HL} = 3, \min_{hel} = 2$	0.1956	0.3824	0.6426	0.6426	0.6431	0.7240

(c) CSP_{freq} (selection principle Centroid).

Table C.1.: **Results related to shapes of selected predictions, obtained for our tRNA data.** They were computed by 10-fold cross-validation procedures, using sample size 1000.

Approach	Parameters	Shape Level					
		0	1	2	3	4	5
PF	$\max_{bulge} = 30$	0.5196	0.6740	0.8160	0.8239	0.8798	0.9556
SCFG	$\min_{HL} = 1, \min_{hel} = 1$	0.6838	0.9459	0.9903	0.9903	0.9908	0.9995
	$\min_{HL} = 1, \min_{hel} = 2$	0.6806	0.9006	0.9630	0.9635	0.9640	0.9991
	$\min_{HL} = 3, \min_{hel} = 1$	0.7148	0.9459	0.9875	0.9880	0.9885	0.9991
	$\min_{HL} = 3, \min_{hel} = 2$	0.7111	0.8997	0.9677	0.9681	0.9686	0.9995
LSCFG	$\min_{HL} = 1, \min_{hel} = 1$	0.8234	0.9288	0.9723	0.9750	0.9727	0.9986
	$\min_{HL} = 1, \min_{hel} = 2$	0.5479	0.8100	0.9006	0.9011	0.9011	0.9963
	$\min_{HL} = 3, \min_{hel} = 1$	0.8391	0.9441	0.9778	0.9783	0.9783	0.9986
	$\min_{HL} = 3, \min_{hel} = 2$	0.5479	0.8160	0.9015	0.9015	0.9020	0.9963

(a) CSO_{freq}.

Approach	Parameters	Shape Level					
		0	1	2	3	4	5
PF	$\max_{bulge} = 30$	21.073	58.200	136.67	140.63	205.54	328.56
SCFG	$\min_{HL} = 1, \min_{hel} = 1$	16.202	98.357	327.26	327.27	327.51	418.80
	$\min_{HL} = 1, \min_{hel} = 2$	25.205	142.50	453.03	453.03	453.10	527.04
	$\min_{HL} = 3, \min_{hel} = 1$	24.883	130.04	392.78	392.79	393.05	494.79
	$\min_{HL} = 3, \min_{hel} = 2$	34.898	173.73	513.05	513.06	513.08	595.26
LSCFG	$\min_{HL} = 1, \min_{hel} = 1$	101.69	326.26	708.52	708.94	709.42	805.87
	$\min_{HL} = 1, \min_{hel} = 2$	101.77	294.14	717.92	717.92	718.37	811.29
	$\min_{HL} = 3, \min_{hel} = 1$	102.65	331.18	717.08	717.54	718.10	818.76
	$\min_{HL} = 3, \min_{hel} = 2$	104.09	300.04	730.09	730.09	730.54	826.21

(b) CS_{num}.

Approach	Parameters	Shape Level					
		0	1	2	3	4	5
PF	$\max_{bulge} = 30$	355.32	130.22	81.796	33.125	22.585	4.8848
SCFG	$\min_{HL} = 1, \min_{hel} = 1$	802.27	244.52	60.504	60.030	59.916	28.764
	$\min_{HL} = 1, \min_{hel} = 2$	652.75	125.69	24.687	24.687	24.687	16.019
	$\min_{HL} = 3, \min_{hel} = 1$	752.71	208.65	48.257	47.797	47.691	21.838
	$\min_{HL} = 3, \min_{hel} = 2$	592.84	103.04	18.921	18.921	18.921	12.053
LSCFG	$\min_{HL} = 1, \min_{hel} = 1$	238.30	15.045	5.6854	5.4122	5.1806	3.2274
	$\min_{HL} = 1, \min_{hel} = 2$	125.37	8.2070	2.6736	2.6736	2.6736	2.4123
	$\min_{HL} = 3, \min_{hel} = 1$	244.62	16.121	6.1883	5.8268	5.6244	3.1974
	$\min_{HL} = 3, \min_{hel} = 2$	126.84	8.2815	2.7296	2.7296	2.7296	2.3869

(c) DS_{num}.

Table C.2.: **Results related to shapes of sampled structures, obtained for our tRNA data.** They were computed by 10-fold cross-validation procedures, using sample size 1000.

Approach	Parameters	Shape Level					
		0	1	2	3	4	5
PF	$\max_{bulge} = 30$	0.0000	0.0009	0.0078	0.0513	0.0261	0.6353
SCFG	$\min_{HL} = 1, \min_{hel} = 1$	0.0000	0.0026	0.0052	0.0131	0.0357	0.7128
	$\min_{HL} = 1, \min_{hel} = 2$	0.0000	0.0052	0.0139	0.0331	0.0522	0.7502
	$\min_{HL} = 3, \min_{hel} = 1$	0.0000	0.0044	0.0113	0.0314	0.0766	0.7781
	$\min_{HL} = 3, \min_{hel} = 2$	0.0009	0.0096	0.0244	0.0609	0.1027	0.8207
LSCFG	$\min_{HL} = 1, \min_{hel} = 1$	0.2002	0.4239	0.4700	0.4857	0.9426	0.9861
	$\min_{HL} = 1, \min_{hel} = 2$	0.0000	0.0087	0.0087	0.0522	0.9321	0.9948
	$\min_{HL} = 3, \min_{hel} = 1$	0.1984	0.4221	0.4710	0.4857	0.9391	0.9835
	$\min_{HL} = 3, \min_{hel} = 2$	0.0000	0.0087	0.0087	0.0522	0.9313	0.9948

(a) CSP_{freq} (selection principle MF struct.).

Approach	Parameters	Shape Level					
		0	1	2	3	4	5
PF	$\max_{bulge} = 30$	0.0000	0.0052	0.0139	0.0835	0.0696	0.6640
SCFG	$\min_{HL} = 1, \min_{hel} = 1$	0.0000	0.0000	0.0000	0.0000	0.0261	0.3820
	$\min_{HL} = 1, \min_{hel} = 2$	0.0000	0.0009	0.0009	0.0035	0.0566	0.4769
	$\min_{HL} = 3, \min_{hel} = 1$	0.0000	0.0000	0.0000	0.0009	0.0261	0.3977
	$\min_{HL} = 3, \min_{hel} = 2$	0.0000	0.0009	0.0009	0.0035	0.0557	0.5387
LSCFG	$\min_{HL} = 1, \min_{hel} = 1$	0.1062	0.3891	0.4290	0.4378	0.9051	0.9835
	$\min_{HL} = 1, \min_{hel} = 2$	0.0000	0.0078	0.0078	0.0514	0.9078	0.9957
	$\min_{HL} = 3, \min_{hel} = 1$	0.1062	0.4065	0.4456	0.4535	0.8990	0.9835
	$\min_{HL} = 3, \min_{hel} = 2$	0.0000	0.0078	0.0078	0.0540	0.9078	0.9948

(b) CSP_{freq} (selection principle MEA struct.).

Approach	Parameters	Shape Level					
		0	1	2	3	4	5
PF	$\max_{bulge} = 30$	0.0000	0.0026	0.0104	0.0775	0.0731	0.7214
SCFG	$\min_{HL} = 1, \min_{hel} = 1$	0.0000	0.0000	0.0000	0.0000	0.0104	0.1097
	$\min_{HL} = 1, \min_{hel} = 2$	0.0000	0.0000	0.0000	0.0000	0.0148	0.1279
	$\min_{HL} = 3, \min_{hel} = 1$	0.0000	0.0000	0.0000	0.0000	0.0078	0.1236
	$\min_{HL} = 3, \min_{hel} = 2$	0.0000	0.0000	0.0000	0.0009	0.0139	0.1549
LSCFG	$\min_{HL} = 1, \min_{hel} = 1$	0.0966	0.2916	0.3238	0.3316	0.8703	0.9686
	$\min_{HL} = 1, \min_{hel} = 2$	0.0000	0.0061	0.0061	0.0426	0.8982	0.9887
	$\min_{HL} = 3, \min_{hel} = 1$	0.0949	0.2951	0.3281	0.3386	0.8660	0.9712
	$\min_{HL} = 3, \min_{hel} = 2$	0.0000	0.0070	0.0070	0.0453	0.8982	0.9861

(c) CSP_{freq} (selection principle Centroid).

Table C.3.: **Results related to shapes of selected predictions, obtained for our 5S rRNA data.** They were computed by 10-fold cross-validation procedures, using sample size 1000.

Approach	Parameters	Shape Level					
		0	1	2	3	4	5
PF	$\max_{bulge} = 30$	0.0009	0.1662	0.3063	0.7580	0.6883	0.9817
SCFG	$\min_{HL} = 1, \min_{hel} = 1$	0.0000	0.2855	0.4526	0.9852	0.9974	1.0000
	$\min_{HL} = 1, \min_{hel} = 2$	0.0017	0.4135	0.5754	0.9861	0.9983	0.9991
	$\min_{HL} = 3, \min_{hel} = 1$	0.0000	0.3308	0.4883	0.9904	0.9974	1.0000
	$\min_{HL} = 3, \min_{hel} = 2$	0.0026	0.4509	0.6372	0.9904	0.9974	0.9991
LSCFG	$\min_{HL} = 1, \min_{hel} = 1$	0.6258	0.8912	0.9295	0.9504	0.9948	1.0000
	$\min_{HL} = 1, \min_{hel} = 2$	0.0000	0.0374	0.0392	0.5588	0.9957	1.0000
	$\min_{HL} = 3, \min_{hel} = 1$	0.6197	0.8938	0.9286	0.9547	0.9948	1.0000
	$\min_{HL} = 3, \min_{hel} = 2$	0.0000	0.0435	0.0453	0.5822	0.9948	1.0000

(a) CSO_{freq}.

Approach	Parameters	Shape Level					
		0	1	2	3	4	5
PF	$\max_{bulge} = 30$	0.0009	0.7571	3.4207	36.641	30.288	600.35
SCFG	$\min_{HL} = 1, \min_{hel} = 1$	0.0000	0.5432	1.1811	20.640	51.834	573.72
	$\min_{HL} = 1, \min_{hel} = 2$	0.0017	1.1428	2.6615	32.051	64.332	608.06
	$\min_{HL} = 3, \min_{hel} = 1$	0.0000	0.6651	1.4309	22.983	54.635	569.80
	$\min_{HL} = 3, \min_{hel} = 2$	0.0026	1.3795	3.1949	36.673	71.080	609.58
LSCFG	$\min_{HL} = 1, \min_{hel} = 1$	42.599	347.33	421.29	455.78	881.11	983.88
	$\min_{HL} = 1, \min_{hel} = 2$	0.0000	8.2238	8.3039	51.288	890.71	993.23
	$\min_{HL} = 3, \min_{hel} = 1$	42.962	347.97	422.19	457.71	875.13	983.67
	$\min_{HL} = 3, \min_{hel} = 2$	0.0000	8.2082	8.2918	51.573	884.49	993.06

(b) CS_{num}.

Approach	Parameters	Shape Level					
		0	1	2	3	4	5
PF	$\max_{bulge} = 30$	710.75	333.72	237.71	93.335	63.661	7.0951
SCFG	$\min_{HL} = 1, \min_{hel} = 1$	999.67	941.77	866.98	336.69	167.10	16.476
	$\min_{HL} = 1, \min_{hel} = 2$	999.18	884.49	764.79	249.02	129.35	14.198
	$\min_{HL} = 3, \min_{hel} = 1$	999.93	947.19	874.03	331.75	163.09	15.620
	$\min_{HL} = 3, \min_{hel} = 2$	999.68	885.81	762.67	239.28	123.91	13.558
LSCFG	$\min_{HL} = 1, \min_{hel} = 1$	318.99	24.878	19.283	8.2879	4.4246	1.2088
	$\min_{HL} = 1, \min_{hel} = 2$	148.01	10.076	8.5355	4.4627	3.5160	1.1297
	$\min_{HL} = 3, \min_{hel} = 1$	325.18	26.023	20.266	8.4599	4.4933	1.2114
	$\min_{HL} = 3, \min_{hel} = 2$	150.56	10.411	8.8139	4.5323	3.5690	1.1279

(c) DS_{num}.

Table C.4.: **Results related to shapes of sampled structures, obtained for our 5S rRNA data.** They were computed by 10-fold cross-validation procedures, using sample size 1000.

Approach	Parameters	Shape Level					
		0	1	2	3	4	5
PF	$\max_{bulge} = 30$	0.0661	0.1255	0.1586	0.2050	0.2183	0.4834
SCFG	$\min_{HL} = 1, \min_{hel} = 1$	0.0530	0.0993	0.1191	0.1324	0.1589	0.3776
	$\min_{HL} = 1, \min_{hel} = 2$	0.0398	0.1193	0.1457	0.1656	0.1856	0.4106
	$\min_{HL} = 3, \min_{hel} = 1$	0.0530	0.1259	0.1390	0.1590	0.1789	0.4107
	$\min_{HL} = 3, \min_{hel} = 2$	0.0530	0.1258	0.1522	0.1788	0.1985	0.4240
LSCFG	$\min_{HL} = 1, \min_{hel} = 1$	0.0199	0.0465	0.0597	0.0663	0.0995	0.3245
	$\min_{HL} = 1, \min_{hel} = 2$	0.0132	0.0465	0.0532	0.0664	0.0797	0.2982
	$\min_{HL} = 3, \min_{hel} = 1$	0.0199	0.0532	0.0664	0.0730	0.0995	0.3179
	$\min_{HL} = 3, \min_{hel} = 2$	0.0132	0.0532	0.0598	0.0664	0.0731	0.2916

(a) CSP_{freq} (selection principle MF struct.).

Approach	Parameters	Shape Level					
		0	1	2	3	4	5
PF	$\max_{bulge} = 30$	0.0660	0.1123	0.1453	0.1984	0.2051	0.4902
SCFG	$\min_{HL} = 1, \min_{hel} = 1$	0.0264	0.0927	0.0993	0.1125	0.1325	0.3778
	$\min_{HL} = 1, \min_{hel} = 2$	0.0264	0.1193	0.1391	0.1523	0.1789	0.4239
	$\min_{HL} = 3, \min_{hel} = 1$	0.0264	0.0927	0.0993	0.1125	0.1325	0.3777
	$\min_{HL} = 3, \min_{hel} = 2$	0.0197	0.1127	0.1391	0.1656	0.2055	0.4109
LSCFG	$\min_{HL} = 1, \min_{hel} = 1$	0.0132	0.0397	0.0530	0.0530	0.0927	0.2118
	$\min_{HL} = 1, \min_{hel} = 2$	0.0000	0.0332	0.0332	0.0398	0.0663	0.2254
	$\min_{HL} = 3, \min_{hel} = 1$	0.0132	0.0397	0.0530	0.0596	0.0794	0.2118
	$\min_{HL} = 3, \min_{hel} = 2$	0.0000	0.0398	0.0465	0.0465	0.0663	0.1854

(b) CSP_{freq} (selection principle MEA struct.).

Approach	Parameters	Shape Level					
		0	1	2	3	4	5
PF	$\max_{bulge} = 30$	0.0793	0.1321	0.1653	0.1917	0.2449	0.5100
SCFG	$\min_{HL} = 1, \min_{hel} = 1$	0.0197	0.0861	0.1059	0.1190	0.1258	0.3181
	$\min_{HL} = 1, \min_{hel} = 2$	0.0197	0.0795	0.0926	0.1191	0.1192	0.3578
	$\min_{HL} = 3, \min_{hel} = 1$	0.0197	0.0795	0.0926	0.1125	0.1125	0.3181
	$\min_{HL} = 3, \min_{hel} = 2$	0.0197	0.0927	0.1125	0.1390	0.1391	0.3577
LSCFG	$\min_{HL} = 1, \min_{hel} = 1$	0.0132	0.0397	0.0530	0.0530	0.0729	0.1656
	$\min_{HL} = 1, \min_{hel} = 2$	0.0000	0.0265	0.0332	0.0332	0.0663	0.1590
	$\min_{HL} = 3, \min_{hel} = 1$	0.0066	0.0397	0.0530	0.0596	0.0728	0.1722
	$\min_{HL} = 3, \min_{hel} = 2$	0.0000	0.0332	0.0398	0.0398	0.0596	0.1590

(c) CSP_{freq} (selection principle Centroid).

Table C.5.: **Results related to shapes of selected predictions, obtained for S-151Rfam database.** They were computed by two-fold cross-validation procedures, using sample size 1000.

Approach	Parameters	Shape Level					
		0	1	2	3	4	5
PF	$\max_{bulge} = 30$	0.3638	0.4433	0.4766	0.5231	0.6488	0.7947
SCFG	$\min_{HL} = 1, \min_{hel} = 1$	0.2520	0.5497	0.6095	0.6888	0.7683	0.9604
	$\min_{HL} = 1, \min_{hel} = 2$	0.2717	0.5630	0.6158	0.7284	0.8079	0.9605
	$\min_{HL} = 3, \min_{hel} = 1$	0.2518	0.5429	0.6093	0.7218	0.7815	0.9472
	$\min_{HL} = 3, \min_{hel} = 2$	0.2715	0.5564	0.6027	0.7087	0.7484	0.9604
LSCFG	$\min_{HL} = 1, \min_{hel} = 1$	0.0463	0.2518	0.4041	0.5496	0.5960	0.8408
	$\min_{HL} = 1, \min_{hel} = 2$	0.0397	0.2320	0.3381	0.4635	0.5033	0.7282
	$\min_{HL} = 3, \min_{hel} = 1$	0.0463	0.2582	0.3908	0.5295	0.5825	0.8075
	$\min_{HL} = 3, \min_{hel} = 2$	0.0331	0.1922	0.2982	0.4305	0.4635	0.6818

(a) CSO_{freq}.

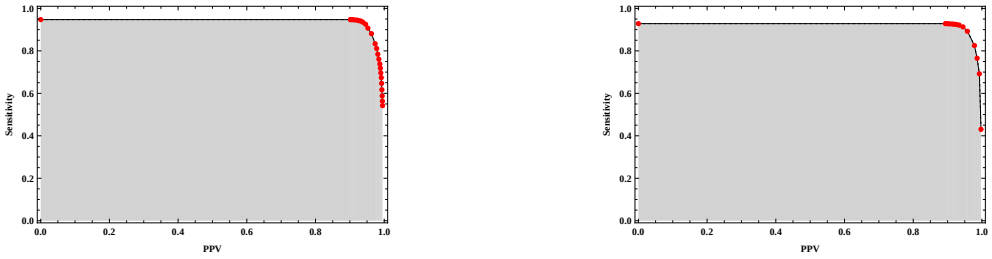
Approach	Parameters	Shape Level					
		0	1	2	3	4	5
PF	$\max_{bulge} = 30$	40.390	88.886	121.55	158.32	195.83	453.58
SCFG	$\min_{HL} = 1, \min_{hel} = 1$	10.743	47.281	63.587	97.088	121.64	362.44
	$\min_{HL} = 1, \min_{hel} = 2$	12.968	58.796	78.776	115.96	139.09	387.16
	$\min_{HL} = 3, \min_{hel} = 1$	12.468	51.569	67.603	104.67	125.50	365.84
	$\min_{HL} = 3, \min_{hel} = 2$	15.059	63.707	83.965	125.82	142.99	391.39
LSCFG	$\min_{HL} = 1, \min_{hel} = 1$	4.6818	30.691	44.362	62.552	92.031	305.66
	$\min_{HL} = 1, \min_{hel} = 2$	3.2041	36.090	48.338	62.027	98.212	293.97
	$\min_{HL} = 3, \min_{hel} = 1$	4.0326	28.718	41.792	59.675	86.897	300.72
	$\min_{HL} = 3, \min_{hel} = 2$	3.3858	35.005	46.601	57.815	92.288	286.40

(b) CS_{num}.

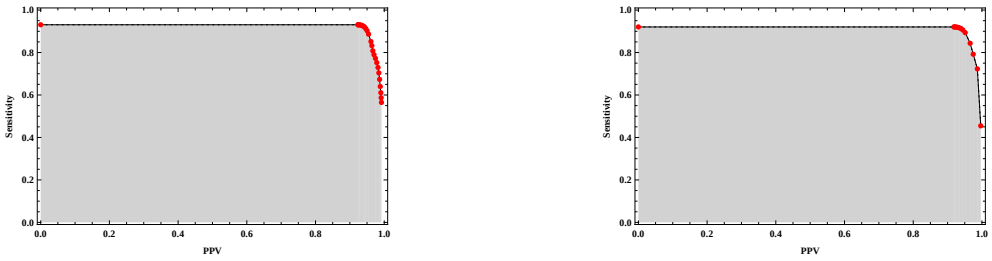
Approach	Parameters	Shape Level					
		0	1	2	3	4	5
PF	$\max_{bulge} = 30$	540.74	304.36	255.40	150.89	117.24	18.795
SCFG	$\min_{HL} = 1, \min_{hel} = 1$	892.14	600.39	526.36	368.49	322.88	99.601
	$\min_{HL} = 1, \min_{hel} = 2$	849.32	538.56	466.17	322.99	286.12	84.480
	$\min_{HL} = 3, \min_{hel} = 1$	888.89	588.97	516.66	358.72	315.25	94.603
	$\min_{HL} = 3, \min_{hel} = 2$	840.03	522.53	452.04	307.61	273.92	77.536
LSCFG	$\min_{HL} = 1, \min_{hel} = 1$	729.44	249.69	201.75	102.87	78.918	13.381
	$\min_{HL} = 1, \min_{hel} = 2$	568.66	172.46	143.60	72.662	57.327	9.5317
	$\min_{HL} = 3, \min_{hel} = 1$	725.66	264.33	217.20	110.27	85.455	13.484
	$\min_{HL} = 3, \min_{hel} = 2$	563.23	180.29	151.89	74.803	59.977	8.9805

(c) DS_{num}.

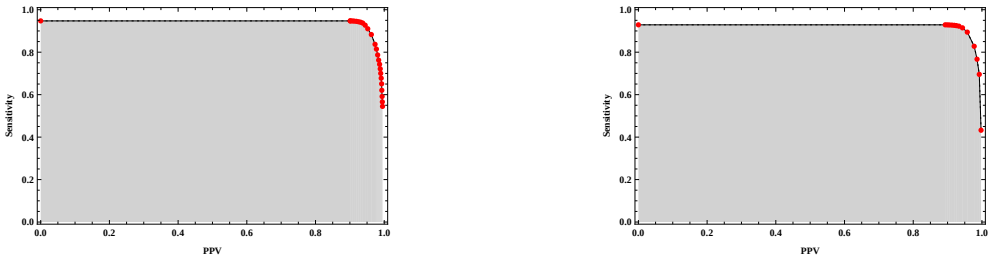
Table C.6.: **Results related to shapes of sampled structures, obtained for S-151Rfam database.** They were computed by two-fold cross-validation procedures, using sample size 1000.



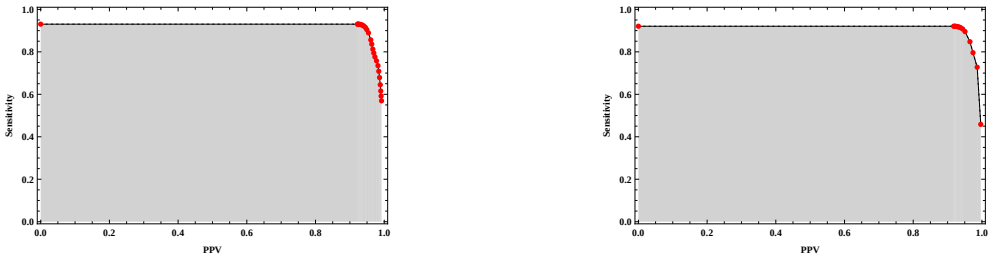
(a) LSCFG approach (with least realistic parameter combination $\min_{\mathrm{HL}} = 1$ and $\min_{\mathrm{hel}} = 1$).



(b) LSCFG approach (with parameter combination $\min_{\mathrm{HL}} = 1$ and $\min_{\mathrm{hel}} = 2$).

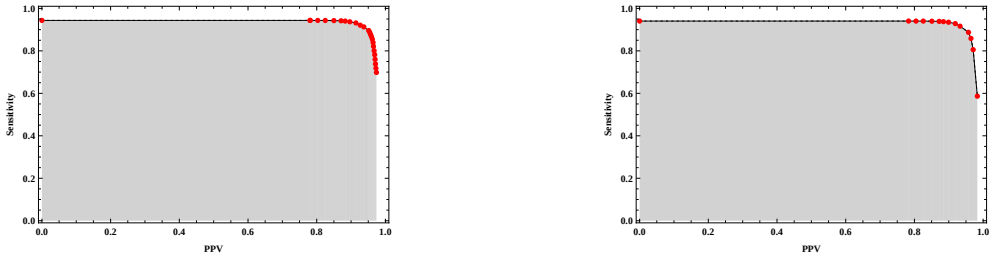


(c) LSCFG approach (with parameter combination $\min_{\mathrm{HL}} = 3$ and $\min_{\mathrm{hel}} = 1$).

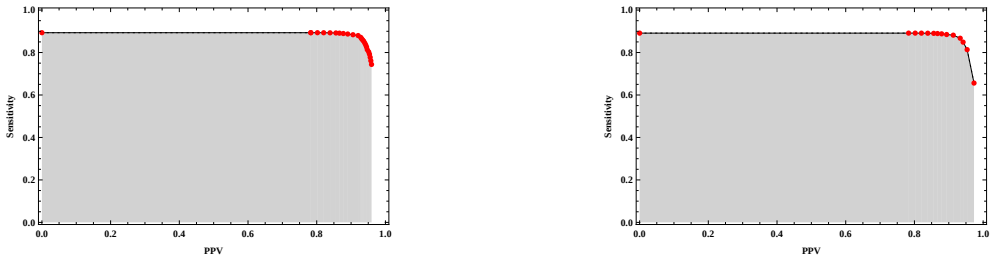


(d) LSCFG approach (with most realistic parameter combination $\min_{\mathrm{HL}} = 3$ and $\min_{\mathrm{hel}} = 2$).

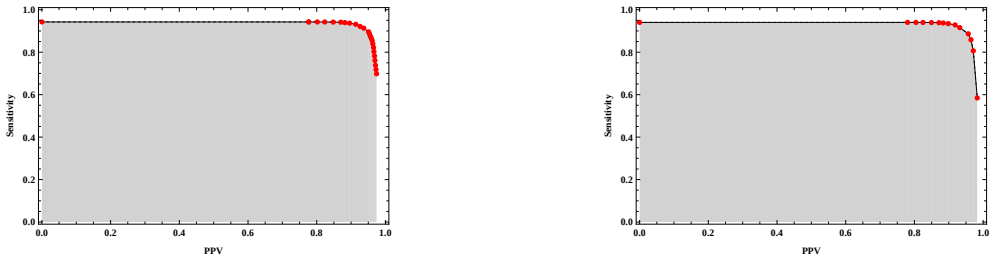
Figure C.1.: **(Areas under) ROC curves obtained for our tRNA database.** Results were computed by 10-fold cross-validation procedures, using sample size 1000. For each considered sampling variant, the corresponding ROC curves are shown for prediction principle MEA structure (figure on the left) and centroid (figure on the right), respectively.



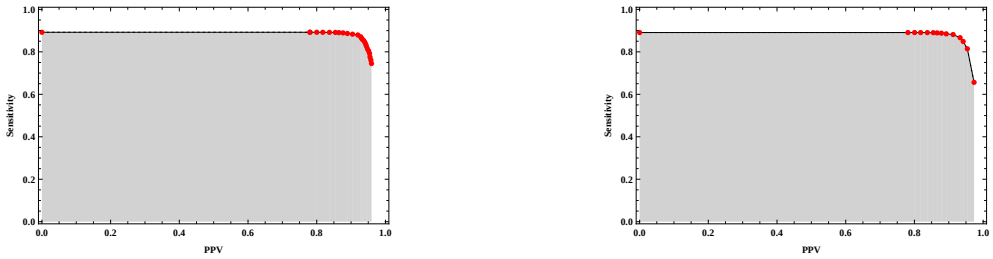
(a) LSCFG approach (with least realistic parameter combination $\min_{\mathrm{HL}} = 1$ and $\min_{\mathrm{hel}} = 1$).



(b) LSCFG approach (with parameter combination $\min_{\mathrm{HL}} = 1$ and $\min_{\mathrm{hel}} = 2$).

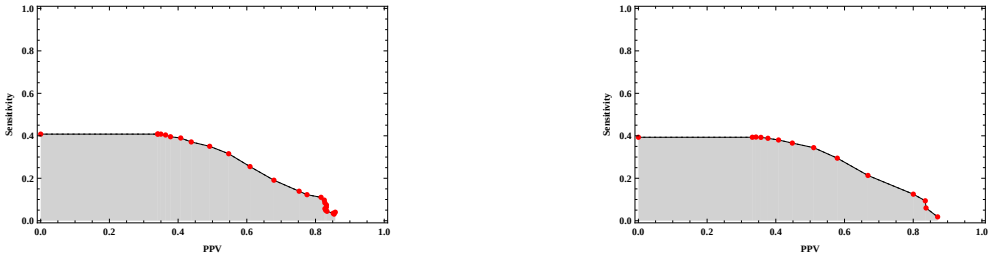


(c) LSCFG approach (with parameter combination $\min_{\mathrm{HL}} = 3$ and $\min_{\mathrm{hel}} = 1$).

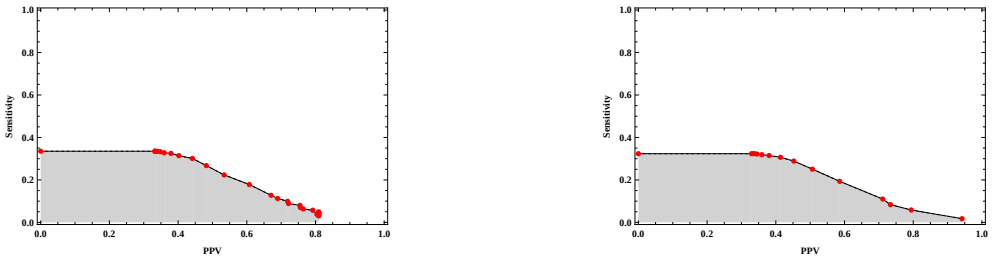


(d) LSCFG approach (with most realistic parameter combination $\min_{\mathrm{HL}} = 3$ and $\min_{\mathrm{hel}} = 2$).

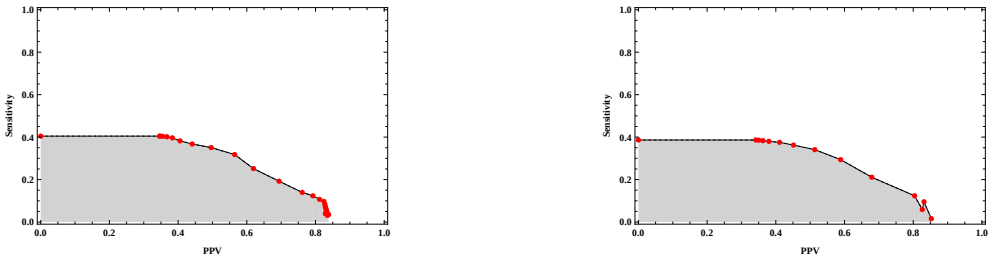
Figure C.2.: **(Areas under) ROC curves obtained for our 5S rRNA database.** Results were computed by 10-fold cross-validation procedures, using sample size 1000. For each considered sampling variant, the corresponding ROC curves are shown for prediction principle MEA structure (figure on the left) and centroid (figure on the right), respectively.



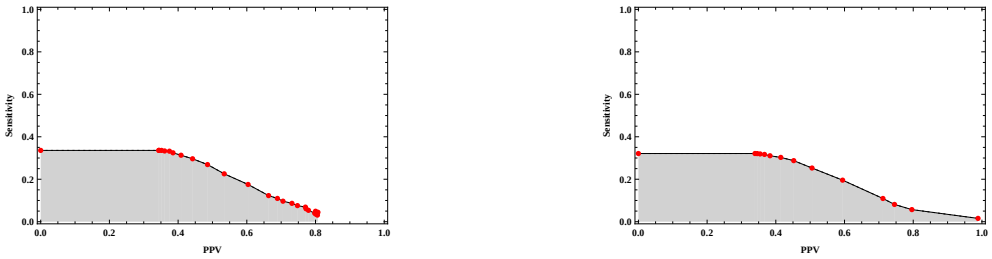
(a) LSCFG approach (with least realistic parameter combination $\min_{\mathrm{HL}} = 1$ and $\min_{\mathrm{hel}} = 1$).



(b) LSCFG approach (with parameter combination $\min_{\mathrm{HL}} = 1$ and $\min_{\mathrm{hel}} = 2$).



(c) LSCFG approach (with parameter combination $\min_{\mathrm{HL}} = 3$ and $\min_{\mathrm{hel}} = 1$).



(d) LSCFG approach (with most realistic parameter combination $\min_{\mathrm{HL}} = 3$ and $\min_{\mathrm{hel}} = 2$).

Figure C.3.: **(Areas under) ROC curves obtained for the mixed S-151Rfam database.** Results were computed by two-fold cross-validation procedures, using the same folds as in [DWB06] and sample size 1000. For each considered sampling variant, the corresponding ROC curves are shown for prediction principle MEA structure (figure on the left) and centroid (figure on the right), respectively.

Chapter D

Tables and Figures Relating to Chapter 8

Approach	Errors	MP struct.		MF struct.		MEA struct.		Centroid	
		Sens.	PPV	Sens.	PPV	Sens.	PPV	Sens.	PPV
SCFG	—	0.7818	0.8437	0.7792	0.8445	0.7324	0.8939	0.6754	0.9158
	mep(0.5)	0.7822	0.8447	0.7599	0.8370	0.7169	0.8927	0.6607	0.9140
	mep(0.75)	0.7793	0.8431	0.7303	0.8217	0.6935	0.8917	0.6356	0.9123
	mep(0.9)	0.7699	0.8409	0.7075	0.8117	0.6715	0.8893	0.6097	0.9115
	mep(0.99)	0.7590	0.8388	0.6768	0.8004	0.6414	0.8877	0.5817	0.9127
	fep(0.5)	0.7798	0.8440	0.7234	0.8184	0.6864	0.8896	0.6292	0.9134
	fep(0.75)	0.7442	0.8313	0.6414	0.7736	0.6066	0.8802	0.5507	0.9032
	fep(0.9)	0.6644	0.8106	0.5257	0.7229	0.4934	0.8652	0.4375	0.8952
	fep(0.99)	0.4101	0.7295	0.2864	0.5590	0.2532	0.7776	0.2157	0.8291
LSCFG	—	0.8545	0.9534	0.8542	0.9535	0.8335	0.9736	0.8250	0.9783
	mep(0.5)	0.8545	0.9534	0.8429	0.9524	0.8236	0.9731	0.8150	0.9773
	mep(0.75)	0.8542	0.9533	0.8281	0.9485	0.8098	0.9709	0.8018	0.9758
	mep(0.9)	0.8546	0.9539	0.8104	0.9425	0.7978	0.9697	0.7889	0.9744
	mep(0.99)	0.8519	0.9533	0.7988	0.9413	0.7833	0.9676	0.7735	0.9726
	fep(0.5)	0.8548	0.9536	0.8224	0.9486	0.8029	0.9707	0.7940	0.9758
	fep(0.75)	0.8524	0.9532	0.7763	0.9323	0.7674	0.9620	0.7589	0.9687
	fep(0.9)	0.8315	0.9492	0.7223	0.9162	0.7131	0.9523	0.7038	0.9601
	fep(0.99)	0.7530	0.9325	0.5769	0.8623	0.5668	0.9075	0.5567	0.9195

(a) Sensitivity and PPV.

Approach	Errors	MEA struct.	Centroid
SCFG	—	0.828522	0.833894
	mep(0.5)	0.819658	0.823811
	mep(0.75)	0.810331	0.813818
	mep(0.9)	0.801393	0.801842
	mep(0.99)	0.786645	0.788478
	fep(0.5)	0.805999	0.807240
	fep(0.75)	0.761806	0.759493
	fep(0.9)	0.682057	0.676879
	fep(0.99)	0.440021	0.422778
LSCFG	—	0.936285	0.919736
	mep(0.5)	0.932121	0.916321
	mep(0.75)	0.925639	0.907926
	mep(0.9)	0.919747	0.900505
	mep(0.99)	0.916540	0.896024
	fep(0.5)	0.924191	0.908943
	fep(0.75)	0.900592	0.884400
	fep(0.9)	0.872742	0.848190
	fep(0.99)	0.752030	0.722737

(b) AUC values.

Table D.1.: **Prediction results for our tRNA database.** They have been computed by 10-fold cross-validation procedures, using sample size 1000 and $\min_{\text{hel}} = \min_{\text{HL}} = 1$.

Approach	Errors	MP struct.		MF struct.		MEA struct.		Centroid	
		Sens.	PPV	Sens.	PPV	Sens.	PPV	Sens.	PPV
SCFG	—	0.4251	0.5372	0.4251	0.5363	0.3403	0.6967	0.2689	0.8044
	mep(0.5)	0.4143	0.5280	0.4160	0.5290	0.3334	0.6987	0.2643	0.8051
	mep(0.75)	0.4113	0.5303	0.4105	0.5289	0.3234	0.7031	0.2566	0.8098
	mep(0.9)	0.4071	0.5311	0.4064	0.5297	0.3120	0.7007	0.2466	0.8050
	mep(0.99)	0.3897	0.5227	0.3894	0.5216	0.2957	0.7069	0.2362	0.8072
	fep(0.5)	0.4055	0.5203	0.4049	0.5198	0.3209	0.7068	0.2532	0.8087
	fep(0.75)	0.3713	0.5070	0.3708	0.5050	0.2795	0.7121	0.2247	0.8183
	fep(0.9)	0.3321	0.4953	0.3261	0.4858	0.2296	0.7344	0.1829	0.8161
	fep(0.99)	0.2043	0.4410	0.1756	0.3788	0.1066	0.6867	0.0814	0.7666
LSCFG	—	0.8993	0.9412	0.8997	0.9409	0.8959	0.9513	0.8873	0.9574
	mep(0.5)	0.8993	0.9412	0.8909	0.9380	0.8903	0.9478	0.8819	0.9541
	mep(0.75)	0.8993	0.9411	0.8816	0.9348	0.8822	0.9459	0.8746	0.9528
	mep(0.9)	0.8993	0.9414	0.8745	0.9323	0.8739	0.9438	0.8666	0.9500
	mep(0.99)	0.8989	0.9414	0.8639	0.9269	0.8659	0.9408	0.8574	0.9482
	fep(0.5)	0.8993	0.9412	0.8796	0.9328	0.8798	0.9445	0.8716	0.9515
	fep(0.75)	0.8963	0.9400	0.8548	0.9217	0.8560	0.9346	0.8480	0.9432
	fep(0.9)	0.8854	0.9353	0.8240	0.9065	0.8260	0.9234	0.8170	0.9338
	fep(0.99)	0.8251	0.9052	0.7162	0.8375	0.7148	0.8661	0.6986	0.8879

(a) Sensitivity and PPV.

Approach	Errors	MEA struct.	Centroid
SCFG	—	0.409278	0.408549
	mep(0.5)	0.401914	0.400515
	mep(0.75)	0.397622	0.396770
	mep(0.9)	0.383750	0.383935
	mep(0.99)	0.376683	0.375488
	fep(0.5)	0.400827	0.397566
	fep(0.75)	0.363824	0.363257
	fep(0.9)	0.326873	0.325467
	fep(0.99)	0.189628	0.182902
LSCFG	—	0.914801	0.918933
	mep(0.5)	0.911963	0.915503
	mep(0.75)	0.908958	0.911579
	mep(0.9)	0.905646	0.908203
	mep(0.99)	0.902330	0.905126
	fep(0.5)	0.906507	0.911063
	fep(0.75)	0.893417	0.895371
	fep(0.9)	0.875529	0.877256
	fep(0.99)	0.776239	0.777355

(b) AUC values.

Table D.2.: **Prediction results for our 5S rRNA database.** They have been computed by 10-fold cross-validation procedures, using sample size 1000 and $\min_{\text{hel}} = \min_{\text{HL}} = 1$.

Approach	Errors	Shape Level					
		0	1	2	3	4	5
SCFG	—	0.2413	0.4082	0.5548	0.5548	0.5552	0.6278
	mep(0.5)	0.2409	0.4068	0.5548	0.5548	0.5552	0.6265
	mep(0.75)	0.2335	0.3990	0.5506	0.5506	0.5511	0.6246
	mep(0.9)	0.2159	0.3809	0.5446	0.5446	0.5451	0.6135
	mep(0.99)	0.1877	0.3551	0.5382	0.5382	0.5386	0.6075
	fep(0.5)	0.2339	0.4017	0.5511	0.5511	0.5516	0.6269
	fep(0.75)	0.1586	0.3269	0.5257	0.5257	0.5261	0.5908
	fep(0.9)	0.0564	0.1979	0.4401	0.4401	0.4401	0.4952
	fep(0.99)	0.0014	0.0384	0.1979	0.1979	0.1984	0.2326
LSCFG	—	0.3324	0.4956	0.6574	0.6574	0.6579	0.7351
	mep(0.5)	0.3324	0.4956	0.6574	0.6574	0.6579	0.7351
	mep(0.75)	0.3329	0.4952	0.6579	0.6579	0.6584	0.7351
	mep(0.9)	0.3315	0.4901	0.6574	0.6574	0.6579	0.7351
	mep(0.99)	0.3236	0.4892	0.6560	0.6560	0.6565	0.7332
	fep(0.5)	0.3324	0.4966	0.6588	0.6588	0.6593	0.7369
	fep(0.75)	0.3232	0.4827	0.6551	0.6551	0.6556	0.7341
	fep(0.9)	0.2358	0.4055	0.6394	0.6399	0.6399	0.7166
	fep(0.99)	0.0624	0.2626	0.6246	0.6250	0.6250	0.6967

(a) CSP_{freq} values (for selection principle MP struct.).

Approach	Errors	Shape Level					
		0	1	2	3	4	5
SCFG	—	0.2099	0.3699	0.5594	0.5594	0.5599	0.6302
	mep(0.5)	0.1683	0.3301	0.5372	0.5372	0.5377	0.6047
	mep(0.75)	0.1128	0.2700	0.5062	0.5062	0.5067	0.5682
	mep(0.9)	0.0712	0.2173	0.4808	0.4808	0.4813	0.5511
	mep(0.99)	0.0522	0.1822	0.4517	0.4517	0.4517	0.5215
	fep(0.5)	0.1049	0.2547	0.5155	0.5155	0.5160	0.5793
	fep(0.75)	0.0231	0.1317	0.4087	0.4087	0.4092	0.4623
	fep(0.9)	0.0032	0.0518	0.2918	0.2918	0.2918	0.3505
	fep(0.99)	0.0000	0.0125	0.1110	0.1110	0.1119	0.2062
LSCFG	—	0.3269	0.4892	0.6560	0.6565	0.6565	0.7337
	mep(0.5)	0.2534	0.4235	0.6708	0.6708	0.6713	0.7485
	mep(0.75)	0.1872	0.3666	0.6741	0.6741	0.6745	0.7550
	mep(0.9)	0.1502	0.3384	0.6694	0.6694	0.6699	0.7545
	mep(0.99)	0.1137	0.2954	0.6801	0.6801	0.6801	0.7568
	fep(0.5)	0.1794	0.3653	0.6704	0.6704	0.6709	0.7531
	fep(0.75)	0.0726	0.2492	0.6708	0.6717	0.6713	0.7596
	fep(0.9)	0.0301	0.1933	0.6847	0.6852	0.6857	0.7688
	fep(0.99)	0.0023	0.1262	0.6334	0.6334	0.6357	0.7240

(b) CSP_{freq} values (for selection principle MF struct.).

Approach	Errors	Shape Level					
		0	1	2	3	4	5
SCFG	—	0.0555	0.2094	0.4193	0.4193	0.4207	0.4679
	mep(0.5)	0.0416	0.1817	0.4045	0.4045	0.4055	0.4489
	mep(0.75)	0.0222	0.1456	0.3694	0.3699	0.3703	0.4147
	mep(0.9)	0.0148	0.1179	0.3555	0.3560	0.3583	0.4031
	mep(0.99)	0.0125	0.0989	0.3112	0.3112	0.3126	0.3570
	fep(0.5)	0.0245	0.1364	0.3662	0.3662	0.3666	0.4059
	fep(0.75)	0.0069	0.0712	0.2682	0.2686	0.2705	0.3070
	fep(0.9)	0.0005	0.0240	0.1655	0.1655	0.1669	0.2006
	fep(0.99)	0.0000	0.0014	0.0245	0.0245	0.0250	0.0546
LSCFG	—	0.1854	0.3574	0.4919	0.4919	0.4919	0.5465
	mep(0.5)	0.1405	0.3056	0.4998	0.4998	0.4998	0.5567
	mep(0.75)	0.1128	0.2760	0.4864	0.4873	0.4864	0.5432
	mep(0.9)	0.0924	0.2478	0.4827	0.4827	0.4827	0.5377
	mep(0.99)	0.0730	0.2191	0.4753	0.4753	0.4753	0.5284
	fep(0.5)	0.1003	0.2556	0.4836	0.4836	0.4836	0.5409
	fep(0.75)	0.0532	0.2011	0.4771	0.4776	0.4771	0.5423
	fep(0.9)	0.0213	0.1341	0.4508	0.4517	0.4508	0.5095
	fep(0.99)	0.0009	0.0781	0.3902	0.3902	0.3921	0.4508

(c) CSP_{freq} values (for selection principle MEA struct.).

Approach	Errors	Shape Level					
		0	1	2	3	4	5
SCFG	—	0.0374	0.1276	0.2973	0.2973	0.2977	0.3130
	mep(0.5)	0.0273	0.1045	0.2779	0.2779	0.2783	0.2908
	mep(0.75)	0.0139	0.0716	0.2362	0.2362	0.2362	0.2520
	mep(0.9)	0.0074	0.0656	0.2354	0.2354	0.2354	0.2502
	mep(0.99)	0.0083	0.0541	0.2007	0.2007	0.2007	0.2173
	fep(0.5)	0.0134	0.0795	0.2473	0.2473	0.2473	0.2603
	fep(0.75)	0.0037	0.0360	0.1609	0.1609	0.1609	0.1734
	fep(0.9)	0.0000	0.0069	0.0865	0.0865	0.0869	0.0939
	fep(0.99)	0.0000	0.0009	0.0120	0.0120	0.0120	0.0227
LSCFG	—	0.1729	0.3158	0.4300	0.4300	0.4300	0.4762
	mep(0.5)	0.1322	0.2728	0.4374	0.4374	0.4374	0.4859
	mep(0.75)	0.1100	0.2469	0.4258	0.4258	0.4258	0.4748
	mep(0.9)	0.0874	0.2140	0.4189	0.4189	0.4189	0.4660
	mep(0.99)	0.0693	0.1914	0.4101	0.4101	0.4101	0.4558
	fep(0.5)	0.0957	0.2261	0.4207	0.4207	0.4207	0.4642
	fep(0.75)	0.0481	0.1688	0.4046	0.4046	0.4046	0.4559
	fep(0.9)	0.0199	0.1146	0.3828	0.3833	0.3828	0.4262
	fep(0.99)	0.0009	0.0633	0.3264	0.3264	0.3269	0.3648

(d) CSP_{freq} values (for selection principle Centroid).

Approach	Errors	Shape Level					
		0	1	2	3	4	5
SCFG	—	0.6838	0.9459	0.9903	0.9903	0.9908	0.9995
	mep(0.5)	0.6274	0.9376	0.9880	0.9884	0.9889	0.9995
	mep(0.75)	0.5724	0.9274	0.9898	0.9908	0.9908	1.0000
	mep(0.9)	0.4707	0.9219	0.9866	0.9871	0.9875	1.0000
	mep(0.99)	0.3837	0.9057	0.9898	0.9903	0.9908	0.9995
	fep(0.5)	0.5534	0.9293	0.9884	0.9889	0.9889	0.9995
	fep(0.75)	0.2903	0.8849	0.9852	0.9857	0.9861	0.9995
	fep(0.9)	0.0883	0.8077	0.9838	0.9843	0.9843	0.9995
	fep(0.99)	0.0018	0.4808	0.9556	0.9575	0.9603	0.9931
LSCFG	—	0.8234	0.9288	0.9723	0.9750	0.9727	0.9986
	mep(0.5)	0.8169	0.9311	0.9658	0.9681	0.9663	0.9986
	mep(0.75)	0.7827	0.9260	0.9732	0.9760	0.9737	0.9986
	mep(0.9)	0.7291	0.9191	0.9718	0.9732	0.9723	0.9986
	mep(0.99)	0.6653	0.9122	0.9709	0.9746	0.9713	0.9986
	fep(0.5)	0.7735	0.9173	0.9704	0.9741	0.9709	0.9986
	fep(0.75)	0.6191	0.9048	0.9686	0.9713	0.9690	0.9986
	fep(0.9)	0.3777	0.8604	0.9732	0.9750	0.9736	0.9986
	fep(0.99)	0.0763	0.7106	0.9663	0.9686	0.9667	0.9986

(e) CSO_{freq} values.

Approach	Errors	Shape Level					
		0	1	2	3	4	5
SCFG	—	16.202	98.357	327.26	327.27	327.51	418.80
	mep(0.5)	13.511	90.408	314.52	314.53	314.83	405.33
	mep(0.75)	9.9097	77.641	295.10	295.12	295.47	387.67
	mep(0.9)	7.1723	66.885	278.33	278.35	278.81	373.45
	mep(0.99)	5.2356	56.709	255.51	255.54	256.20	354.90
	fep(0.5)	9.7356	77.552	294.89	294.91	295.31	387.38
	fep(0.75)	3.0058	48.215	239.50	239.53	240.34	333.77
	fep(0.9)	0.5193	22.122	171.61	171.72	173.31	270.84
	fep(0.99)	0.0028	5.3030	62.460	62.587	66.606	168.64
LSCFG	—	101.69	326.26	708.52	708.94	709.42	805.87
	mep(0.5)	90.408	307.25	712.29	712.73	713.23	810.52
	mep(0.75)	75.220	288.57	710.49	710.94	711.54	810.28
	mep(0.9)	62.276	270.87	708.82	709.20	709.93	809.72
	mep(0.99)	51.262	252.51	708.00	708.35	709.20	807.00
	fep(0.5)	70.493	281.00	710.79	711.18	711.75	810.24
	fep(0.75)	40.142	229.20	704.79	705.33	706.27	807.32
	fep(0.9)	20.373	193.27	695.77	696.29	698.20	802.86
	fep(0.99)	1.6771	118.55	612.73	612.89	620.77	752.68

(f) CS_{num} values.

Approach	Errors	Shape Level					
		0	1	2	3	4	5
SCFG	—	802.27	244.52	60.504	60.030	59.916	28.764
	mep(0.5)	805.91	250.25	63.247	62.668	62.520	29.780
	mep(0.75)	812.80	259.32	67.778	67.072	66.880	31.417
	mep(0.9)	812.01	267.01	72.572	71.611	71.323	32.755
	mep(0.99)	822.99	285.70	81.360	80.006	79.564	35.462
	fep(0.5)	813.01	261.18	68.849	68.053	67.839	31.848
	fep(0.75)	820.06	289.39	86.049	84.434	83.880	37.363
	fep(0.9)	823.37	338.60	120.77	116.73	114.89	47.031
	fep(0.99)	787.70	437.67	225.13	209.08	198.18	68.691
LSCFG	—	238.30	15.045	5.6854	5.4122	5.1806	3.2274
	mep(0.5)	237.12	15.061	5.7478	5.4543	5.2231	3.1970
	mep(0.75)	234.02	15.304	5.9289	5.6123	5.3695	3.2496
	mep(0.9)	230.18	15.572	6.0783	5.7631	5.5047	3.2811
	mep(0.99)	226.25	16.097	6.3746	6.0086	5.7673	3.3176
	fep(0.5)	234.80	15.367	6.0292	5.7213	5.4755	3.2501
	fep(0.75)	215.02	15.921	6.4449	6.0982	5.8435	3.2944
	fep(0.9)	199.10	17.503	7.5125	7.0207	6.7294	3.4531
	fep(0.99)	164.88	22.047	10.309	9.5385	9.1512	3.7212

(g) DS_{num} values.

Table D.3.: **Comparison of sampling quality for tRNAs.** Tables record specific values related to shapes of predictions and sampled structures, obtained from our tRNA database. All results were computed by 10-fold cross-validation procedures, using sample size 1000 and $\min_{hel} = \min_{HL} = 1$.

Approach	Errors	Shape Level					
		0	1	2	3	4	5
SCFG	—	0.0000	0.0026	0.0052	0.0131	0.0366	0.7110
	mep(0.5)	0.0000	0.0009	0.0026	0.0113	0.0287	0.7128
	mep(0.75)	0.0000	0.0017	0.0035	0.0105	0.0322	0.7050
	mep(0.9)	0.0000	0.0009	0.0017	0.0078	0.0331	0.7180
	mep(0.99)	0.0000	0.0026	0.0044	0.0095	0.0227	0.6919
	fep(0.5)	0.0000	0.0017	0.0043	0.0113	0.0374	0.6954
	fep(0.75)	0.0000	0.0000	0.0009	0.0113	0.0321	0.6710
	fep(0.9)	0.0000	0.0009	0.0009	0.0052	0.0261	0.6536
	fep(0.99)	0.0000	0.0000	0.0000	0.0017	0.0096	0.5474
LSCFG	—	0.2141	0.4256	0.4744	0.4900	0.9408	0.9843
	mep(0.5)	0.2141	0.4256	0.4744	0.4900	0.9408	0.9843
	mep(0.75)	0.2141	0.4248	0.4726	0.4892	0.9399	0.9843
	mep(0.9)	0.2089	0.4274	0.4761	0.4926	0.9399	0.9843
	mep(0.99)	0.1941	0.4221	0.4761	0.4892	0.9452	0.9852
	fep(0.5)	0.2124	0.4248	0.4726	0.4883	0.9417	0.9852
	fep(0.75)	0.1898	0.4213	0.4674	0.4831	0.9408	0.9843
	fep(0.9)	0.1314	0.4013	0.4518	0.4726	0.9321	0.9869
	fep(0.99)	0.0209	0.3029	0.3725	0.4186	0.8529	0.9809

(a) CSP_{freq} values (for selection principle MP struct.).

Approach	Errors	Shape Level					
		0	1	2	3	4	5
SCFG	—	0.0000	0.0026	0.0052	0.0131	0.0357	0.7128
	mep(0.5)	0.0000	0.0009	0.0026	0.0122	0.0305	0.7180
	mep(0.75)	0.0000	0.0017	0.0043	0.0113	0.0331	0.7067
	mep(0.9)	0.0000	0.0009	0.0017	0.0078	0.0357	0.7215
	mep(0.99)	0.0000	0.0026	0.0044	0.0105	0.0235	0.6902
	fep(0.5)	0.0000	0.0017	0.0043	0.0113	0.0383	0.6971
	fep(0.75)	0.0000	0.0000	0.0009	0.0113	0.0296	0.6745
	fep(0.9)	0.0000	0.0000	0.0000	0.0035	0.0261	0.6631
	fep(0.99)	0.0000	0.0000	0.0000	0.0035	0.0200	0.5439
LSCFG	—	0.2002	0.4256	0.4700	0.4866	0.9417	0.9861
	mep(0.5)	0.1332	0.3960	0.4439	0.4587	0.9434	0.9869
	mep(0.75)	0.0923	0.3847	0.4448	0.4639	0.9356	0.9861
	mep(0.9)	0.0575	0.3508	0.4135	0.4352	0.9373	0.9887
	mep(0.99)	0.0365	0.3630	0.4308	0.4491	0.9304	0.9861
	fep(0.5)	0.0801	0.3847	0.4404	0.4561	0.9400	0.9861
	fep(0.75)	0.0339	0.3630	0.4230	0.4430	0.9208	0.9843
	fep(0.9)	0.0131	0.3160	0.3743	0.4204	0.8442	0.9843
	fep(0.99)	0.0035	0.1497	0.2106	0.3325	0.5440	0.9730

(b) CSP_{freq} values (for selection principle MF struct.).

Approach	Errors	Shape Level					
		0	1	2	3	4	5
SCFG	—	0.0000	0.0000	0.0000	0.0000	0.0261	0.3821
	mep(0.5)	0.0000	0.0000	0.0000	0.0000	0.0209	0.3698
	mep(0.75)	0.0000	0.0000	0.0000	0.0000	0.0209	0.3559
	mep(0.9)	0.0000	0.0000	0.0000	0.0000	0.0131	0.3290
	mep(0.99)	0.0000	0.0000	0.0000	0.0000	0.0122	0.3003
	fep(0.5)	0.0000	0.0000	0.0000	0.0000	0.0252	0.3438
	fep(0.75)	0.0000	0.0000	0.0000	0.0000	0.0139	0.2463
	fep(0.9)	0.0000	0.0000	0.0000	0.0000	0.0070	0.1619
	fep(0.99)	0.0000	0.0000	0.0000	0.0000	0.0026	0.0444
LSCFG	—	0.1062	0.3891	0.4291	0.4378	0.9051	0.9835
	mep(0.5)	0.1010	0.3751	0.4134	0.4239	0.8921	0.9782
	mep(0.75)	0.0749	0.3647	0.4047	0.4282	0.8894	0.9774
	mep(0.9)	0.0470	0.3290	0.3769	0.3917	0.8834	0.9817
	mep(0.99)	0.0392	0.3429	0.3986	0.4213	0.8712	0.9791
	fep(0.5)	0.0740	0.3839	0.4239	0.4387	0.8877	0.9791
	fep(0.75)	0.0287	0.3516	0.3943	0.4134	0.8616	0.9713
	fep(0.9)	0.0139	0.2968	0.3490	0.3855	0.8120	0.9739
	fep(0.99)	0.0017	0.1358	0.1863	0.2942	0.4970	0.9634

(c) CSP_{freq} values (for selection principle MEA struct.).

Approach	Errors	Shape Level					
		0	1	2	3	4	5
SCFG	—	0.0000	0.0000	0.0000	0.0000	0.0104	0.1097
	mep(0.5)	0.0000	0.0000	0.0000	0.0000	0.0104	0.1062
	mep(0.75)	0.0000	0.0000	0.0000	0.0000	0.0078	0.0923
	mep(0.9)	0.0000	0.0000	0.0000	0.0000	0.0044	0.0896
	mep(0.99)	0.0000	0.0000	0.0000	0.0000	0.0078	0.0827
	fep(0.5)	0.0000	0.0000	0.0000	0.0000	0.0061	0.0932
	fep(0.75)	0.0000	0.0000	0.0000	0.0000	0.0026	0.0696
	fep(0.9)	0.0000	0.0000	0.0000	0.0000	0.0017	0.0479
	fep(0.99)	0.0000	0.0000	0.0000	0.0000	0.0009	0.0078
LSCFG	—	0.0966	0.2916	0.3238	0.3316	0.8703	0.9686
	mep(0.5)	0.0879	0.3142	0.3516	0.3621	0.8625	0.9686
	mep(0.75)	0.0644	0.3029	0.3403	0.3551	0.8407	0.9678
	mep(0.9)	0.0427	0.2829	0.3194	0.3299	0.8451	0.9678
	mep(0.99)	0.0322	0.2924	0.3377	0.3595	0.8294	0.9651
	fep(0.5)	0.0662	0.3194	0.3551	0.3638	0.8512	0.9695
	fep(0.75)	0.0261	0.2907	0.3255	0.3516	0.8103	0.9608
	fep(0.9)	0.0113	0.2411	0.2872	0.3194	0.7650	0.9565
	fep(0.99)	0.0017	0.1053	0.1471	0.2219	0.4831	0.9339

(d) CSP_{freq} values (for selection principle Centroid).

Approach	Errors	Shape Level					
		0	1	2	3	4	5
SCFG	—	0.0000	0.2855	0.4526	0.9852	0.9974	1.0000
	mep(0.5)	0.0000	0.2750	0.4256	0.9835	0.9991	1.0000
	mep(0.75)	0.0000	0.2367	0.3768	0.9774	0.9982	1.0000
	mep(0.9)	0.0000	0.2185	0.3394	0.9696	0.9965	1.0000
	mep(0.99)	0.0000	0.1715	0.2977	0.9756	0.9991	1.0000
	fep(0.5)	0.0000	0.2237	0.3543	0.9739	0.9991	1.0000
	fep(0.75)	0.0000	0.1584	0.2472	0.9574	0.9957	1.0000
	fep(0.9)	0.0000	0.0749	0.1497	0.9147	0.9930	1.0000
	fep(0.99)	0.0000	0.0174	0.0296	0.6763	0.9608	1.0000
LSCFG	—	0.6258	0.8912	0.9295	0.9504	0.9948	1.0000
	mep(0.5)	0.6084	0.8947	0.9286	0.9469	0.9948	1.0000
	mep(0.75)	0.5727	0.8886	0.9269	0.9521	0.9957	1.0000
	mep(0.9)	0.5231	0.8851	0.9252	0.9521	0.9948	1.0000
	mep(0.99)	0.4630	0.8894	0.9199	0.9452	0.9939	1.0000
	fep(0.5)	0.5553	0.8868	0.9234	0.9504	0.9948	1.0000
	fep(0.75)	0.4248	0.8894	0.9225	0.9521	0.9948	1.0000
	fep(0.9)	0.2393	0.8720	0.9077	0.9504	0.9957	1.0000
	fep(0.99)	0.0279	0.7580	0.8460	0.9617	0.9939	1.0000

(e) CSO_{freq} values.

Approach	Errors	Shape Level					
		0	1	2	3	4	5
SCFG	—	0.0000	0.5432	1.1811	20.640	51.834	573.72
	mep(0.5)	0.0000	0.4980	1.0481	19.498	49.614	566.13
	mep(0.75)	0.0000	0.3977	0.8859	18.385	47.128	556.17
	mep(0.9)	0.0000	0.3655	0.7353	16.331	43.735	544.22
	mep(0.99)	0.0000	0.2768	0.5850	14.689	40.062	527.12
	fep(0.5)	0.0000	0.3865	0.7982	17.401	45.868	552.82
	fep(0.75)	0.0000	0.2481	0.4961	13.092	38.088	507.97
	fep(0.9)	0.0000	0.0957	0.2141	8.7270	27.742	443.65
	fep(0.99)	0.0000	0.0191	0.0348	2.9269	12.064	285.20
LSCFG	—	42.599	347.33	421.29	455.78	881.11	983.88
	mep(0.5)	38.324	346.45	419.23	455.04	875.92	983.26
	mep(0.75)	31.890	338.26	411.90	451.93	865.28	983.84
	mep(0.9)	23.180	316.85	389.48	434.11	853.47	984.36
	mep(0.99)	17.873	312.64	386.51	436.20	832.96	983.29
	fep(0.5)	29.194	342.28	413.57	454.07	861.46	983.23
	fep(0.75)	14.829	304.49	376.16	430.85	811.76	980.03
	fep(0.9)	7.7946	250.74	312.98	391.21	713.62	980.94
	fep(0.99)	0.9219	93.694	133.47	260.11	418.20	970.01

(f) CS_{num} values.

Approach	Errors	Shape Level					
		0	1	2	3	4	5
SCFG	—	999.67	941.77	866.98	336.69	167.10	16.476
	mep(0.5)	999.59	943.47	871.08	345.35	171.57	16.766
	mep(0.75)	999.61	946.99	878.58	358.32	179.50	17.508
	mep(0.9)	999.53	949.65	884.73	372.57	188.32	18.198
	mep(0.99)	999.49	953.90	894.21	393.84	201.28	18.948
	fep(0.5)	999.53	947.08	879.39	363.20	182.17	17.663
	fep(0.75)	999.39	955.12	898.94	414.68	213.39	20.622
	fep(0.9)	998.86	962.12	917.65	484.74	258.19	25.174
	fep(0.99)	996.37	966.76	933.73	632.35	367.71	40.976
LSCFG	—	318.99	24.878	19.283	8.2879	4.4246	1.2088
	mep(0.5)	320.15	25.352	19.707	8.3590	4.4759	1.2245
	mep(0.75)	318.55	26.391	20.555	8.5940	4.6395	1.2271
	mep(0.9)	320.83	27.964	21.834	8.8192	4.7788	1.2219
	mep(0.99)	326.13	30.176	23.552	9.1464	4.9729	1.2463
	fep(0.5)	321.32	26.848	21.000	8.6976	4.6169	1.2202
	fep(0.75)	324.45	31.466	24.610	9.4810	5.1226	1.2445
	fep(0.9)	336.16	41.060	32.527	11.069	6.0148	1.2880
	fep(0.99)	401.88	84.057	69.689	18.023	9.1200	1.3690

(g) DS_{num} values.

Table D.4.: **Comparison of sampling quality for 5S rRNAs.** Tables record specific values related to shapes of predictions and sampled structures, obtained from our 5S rRNA database. All results were computed by 10-fold cross-validation procedures, using sample size 1000 and $\min_{hel} = \min_{HL} = 1$.

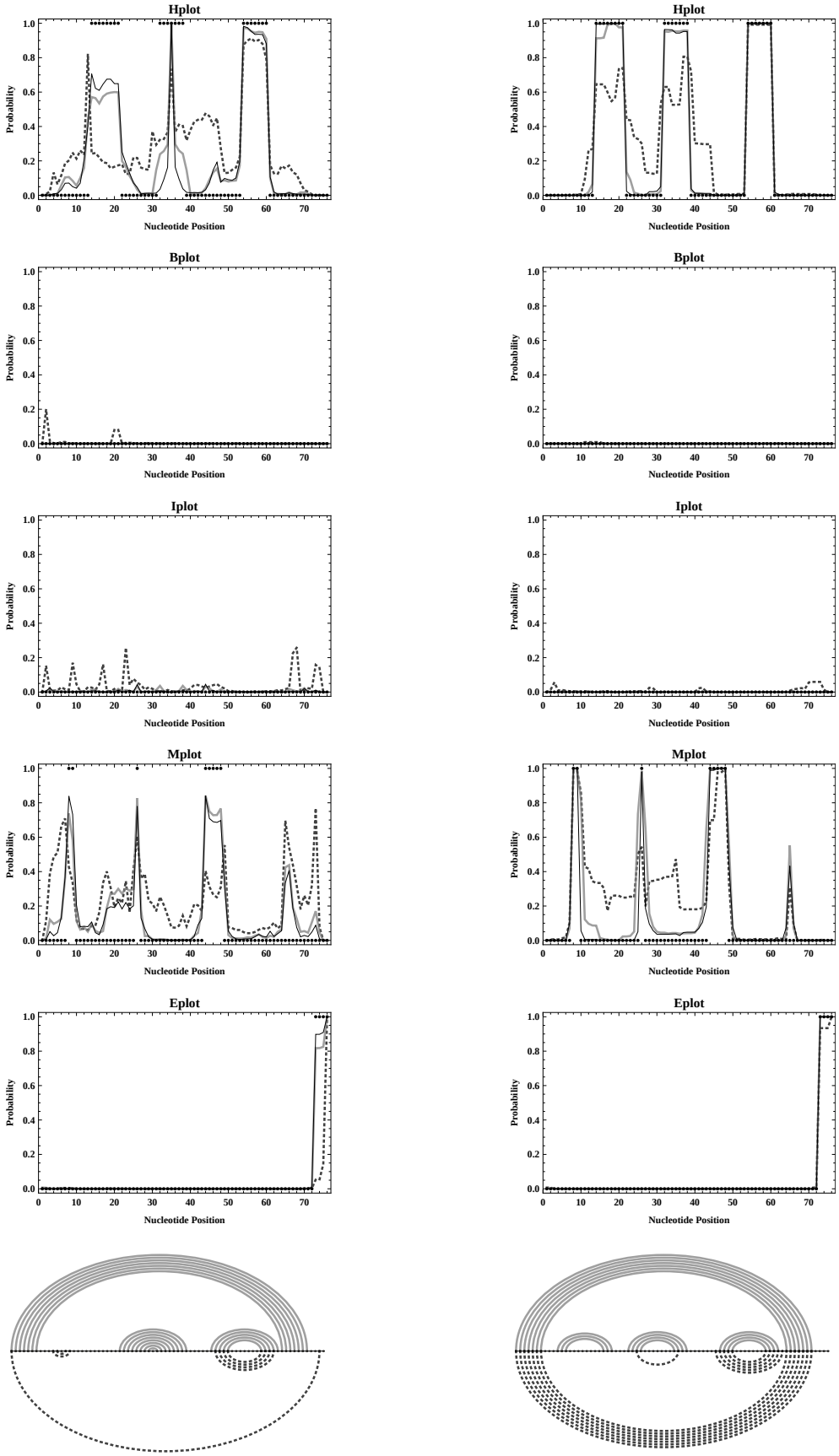


Figure D.1.: **Profiling results obtained with relative disturbances.** Figure shows loop profiles and centroid for *E.coli* tRNA^{Ala} derived according to $\text{mep}(\texttt{prob})$ (thick gray lines) and $\text{fep}(\texttt{prob})$ (thick dotted darker gray lines) under the assumption of the SCFG (figures on the left) and LSCFG (figures on the right) model, respectively, where percentage $\texttt{prob} = 0.99$ has been used for generating the relative errors. Hplot, Bplot, Iplot, Mplot and Eplot display the probability that an unpaired base lies in a hairpin, bulge, interior, multi-branched and exterior loop, respectively.

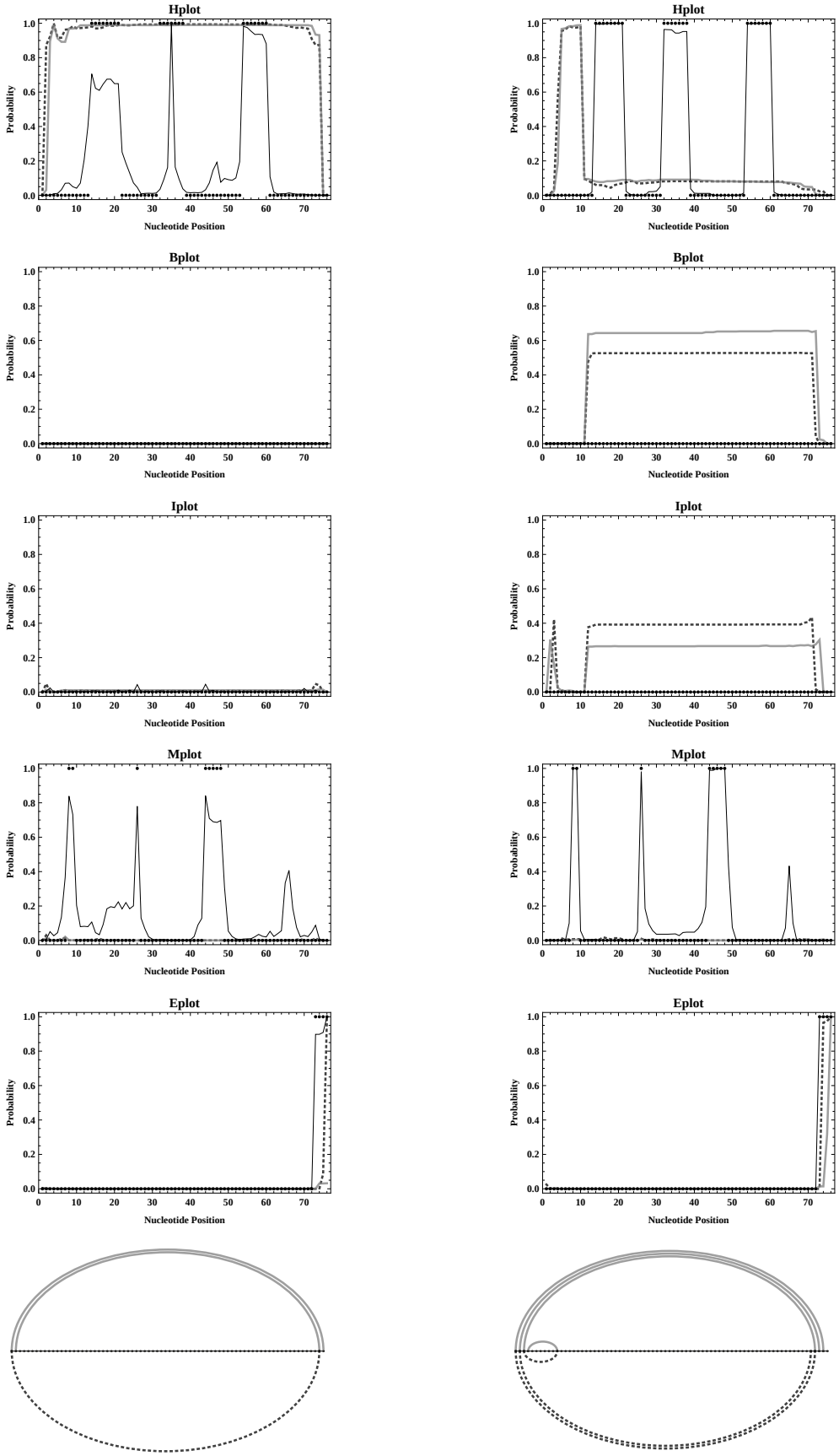


Figure D.2.: **Profiling results obtained with absolute disturbances.** Figure shows loop profiles and centroid for *E.coli* $\mathrm{tRNA}^{\mathrm{Ala}}$ derived according to $\mathrm{mev}(\mathtt{prob})$ (thick gray lines) and $\mathrm{fev}(\mathtt{prob})$ (thick dotted darker gray lines) under the assumption of the SCFG (figures on the left) and LSCFG (figures on the right) model, respectively, where fixed value $\mathtt{prob} = 10^{-9}$ has been used for generating the absolute errors. Hplot, Bplot, Iplot, Mplot and Eplot display the probability that an unpaired base lies in a hairpin, bulge, interior, multi-branched and exterior loop, respectively.

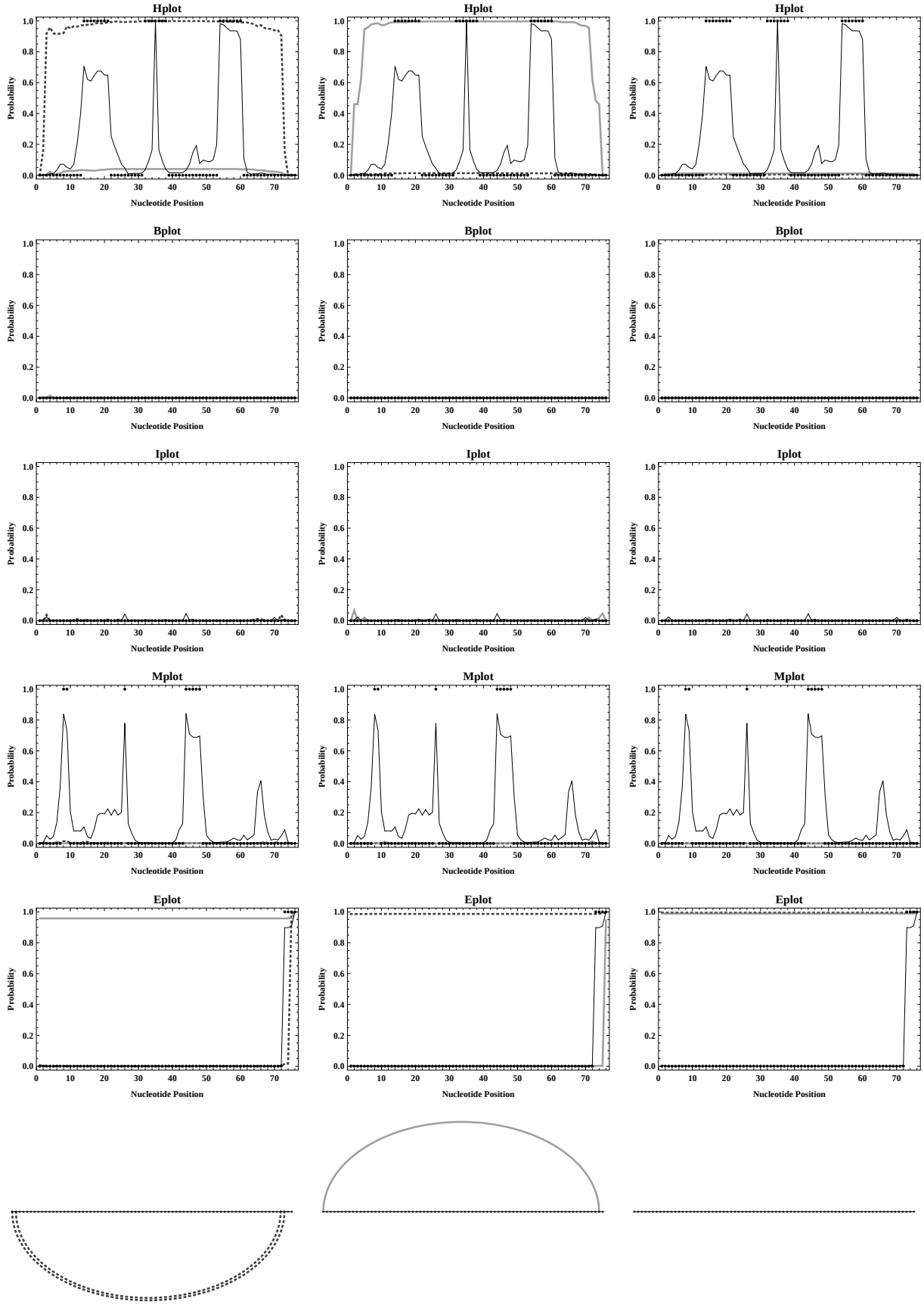


Figure D.3.: **Profiling results for SCFG approach with absolute errors only for long fragments.** Figure shows loop profiles and centroid derived according to $\mathrm{mev}^{win,+}(\mathtt{prob})$ (thick gray lines) and $\mathrm{fev}^{win,+}(\mathtt{prob})$ (thick dotted darker gray lines) for the traditional SCFG model, respectively, where $\mathtt{prob} = 10^{-9}$ and $win \in \{15, 38, 60\}$ (figures from left to right).

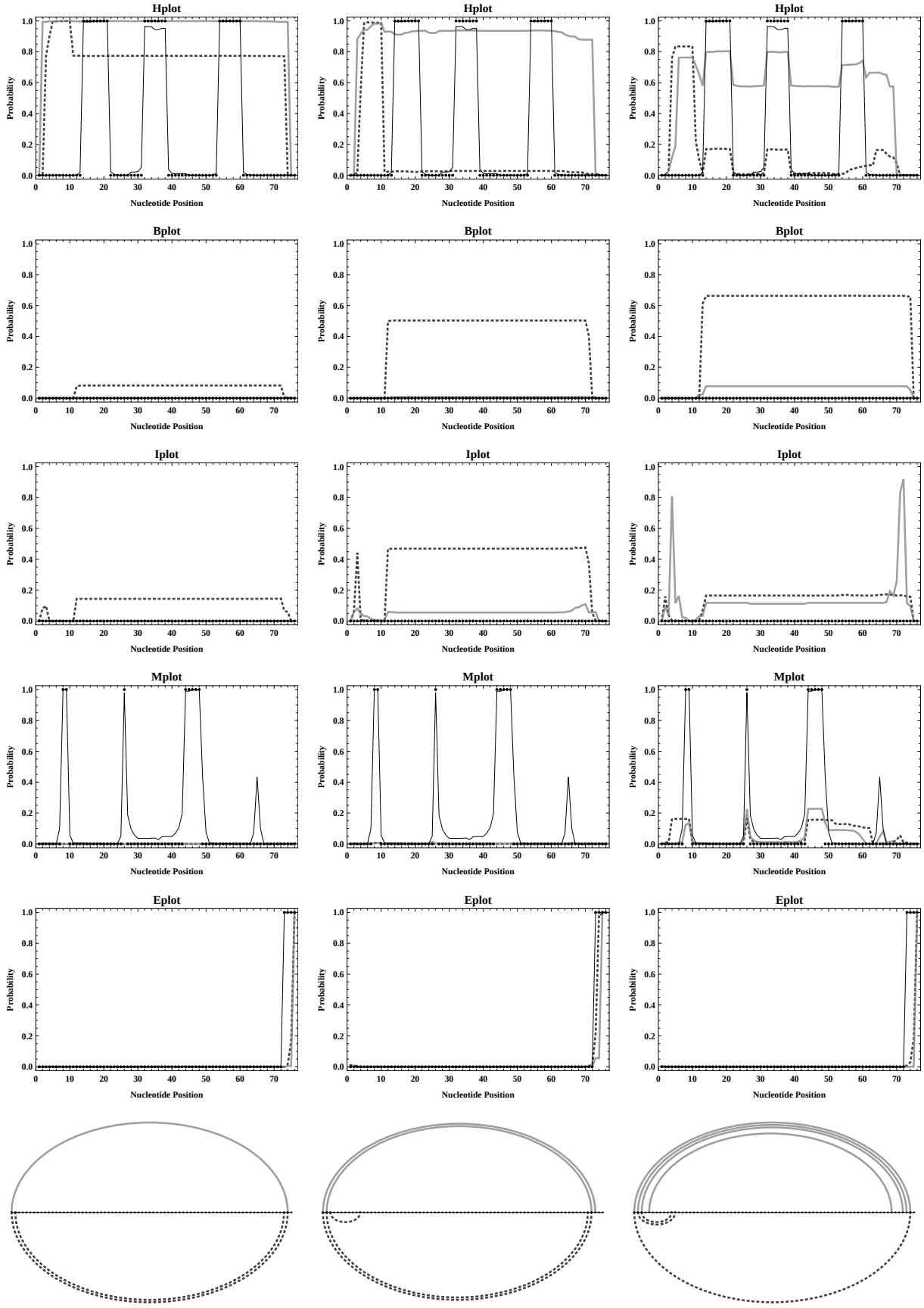


Figure D.4.: **Profiling results for LSCFG approach with absolute errors only for long fragments.** Figure shows loop profiles and centroid derived according to $\mathrm{mev}^{win,+}(\mathtt{prob})$ (thick gray lines) and $\mathrm{fev}^{win,+}(\mathtt{prob})$ (thick dotted darker gray lines) for the LSCFG model, respectively, where $\mathtt{prob} = 10^{-9}$ and $win \in \{15, 38, 60\}$ (figures from left to right).

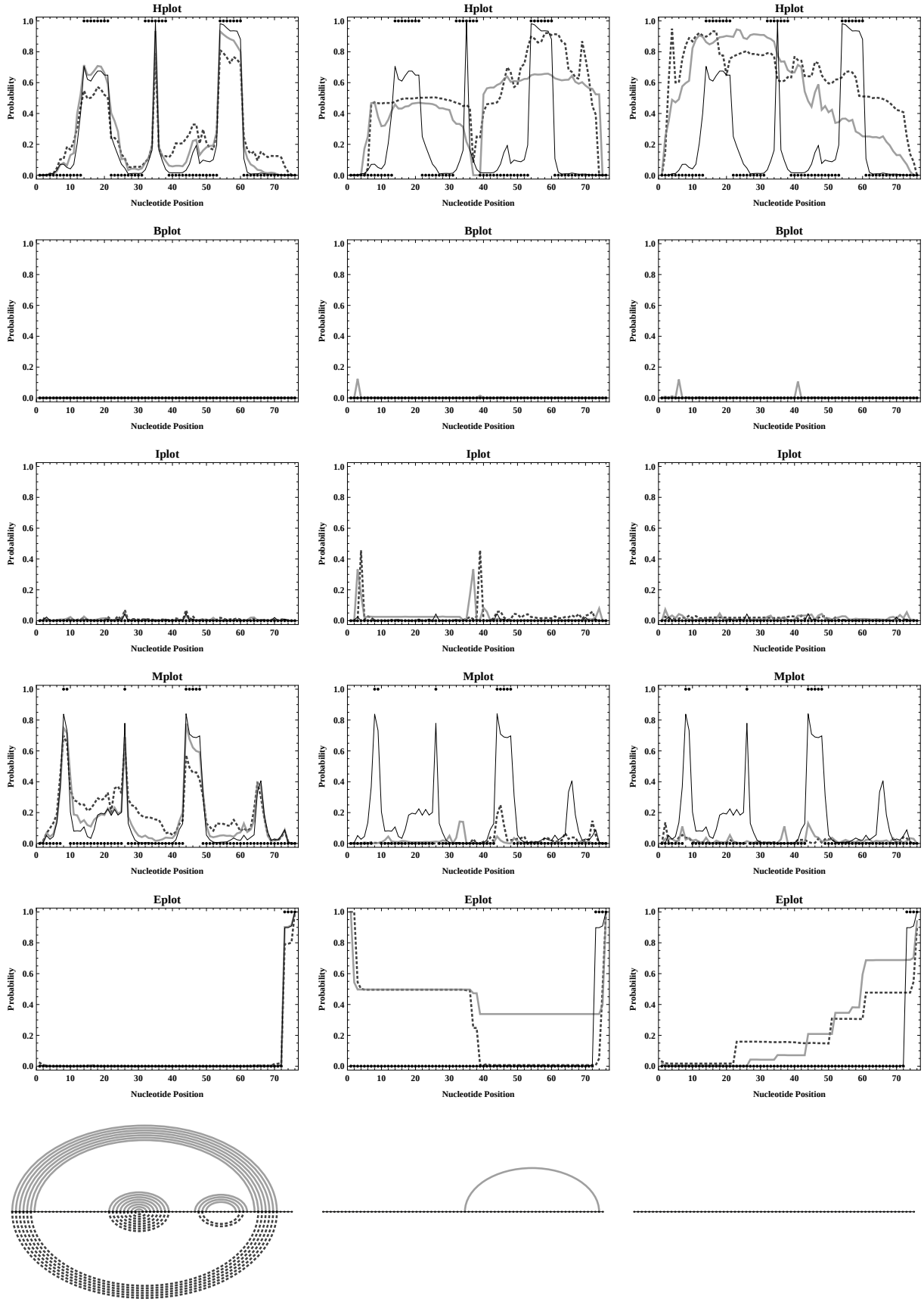


Figure D.5.: **Profiling results for SCFG approach with absolute errors only for short fragments.** Figure shows loop profiles and centroid derived according to $\mathrm{mev}^{win,-}(\mathtt{prob})$ (thick gray lines) and $\mathrm{fev}^{win,-}(\mathtt{prob})$ (thick dotted darker gray lines) for the traditional SCFG model, respectively, where $\mathtt{prob} = 10^{-9}$ and $win \in \{15, 38, 60\}$ (figures from left to right).

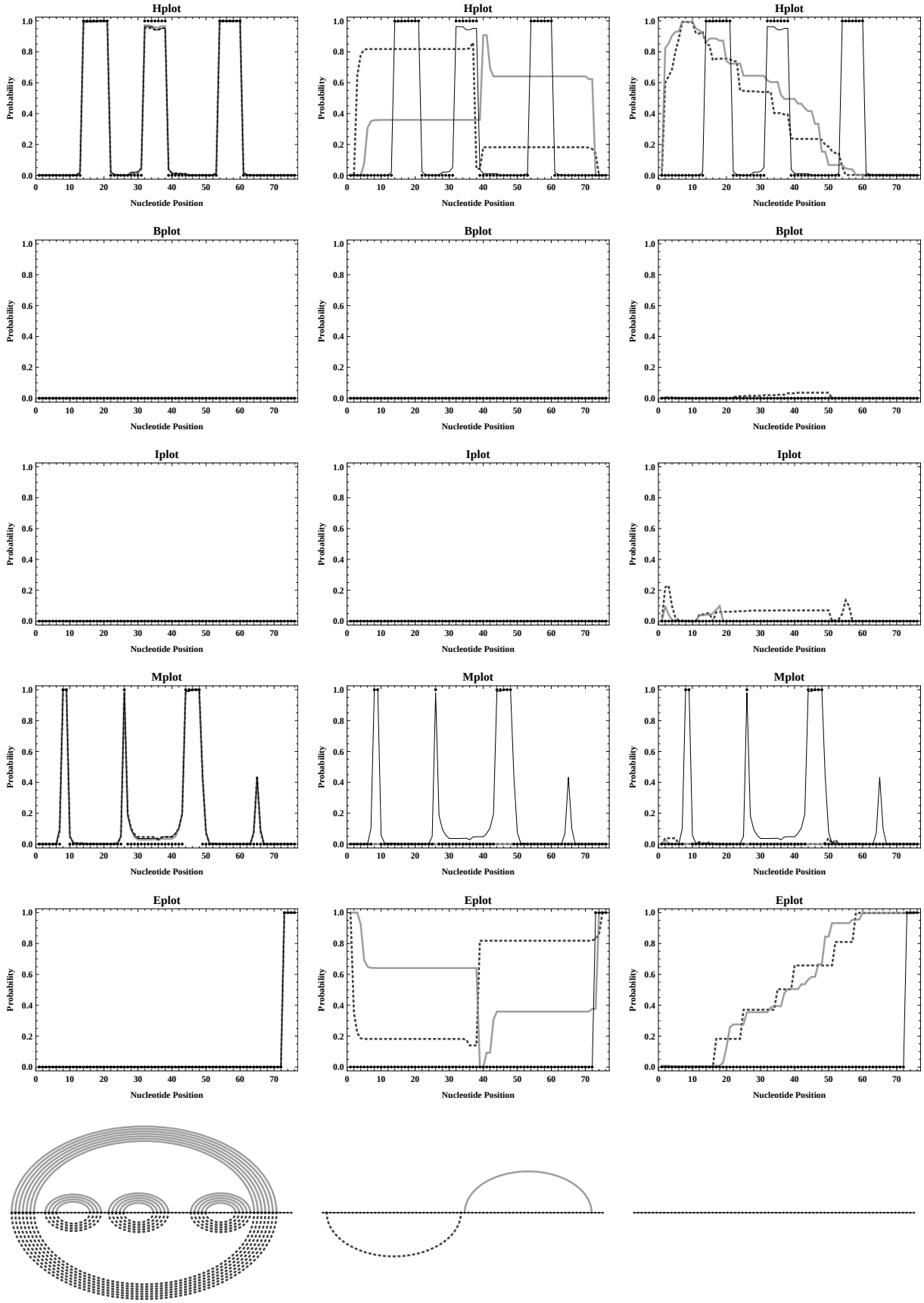


Figure D.6.: **Profiling results for LSCFG approach with absolute errors only for short fragments.** Figure shows loop profiles and centroid derived according to $\mathrm{mev}^{win,-}(\mathtt{prob})$ (thick gray lines) and $\mathrm{fev}^{win,-}(\mathtt{prob})$ (thick dotted darker gray lines) for the LSCFG model, respectively, where $\mathtt{prob} = 10^{-9}$ and $win \in \{15, 38, 60\}$ (figures from left to right).

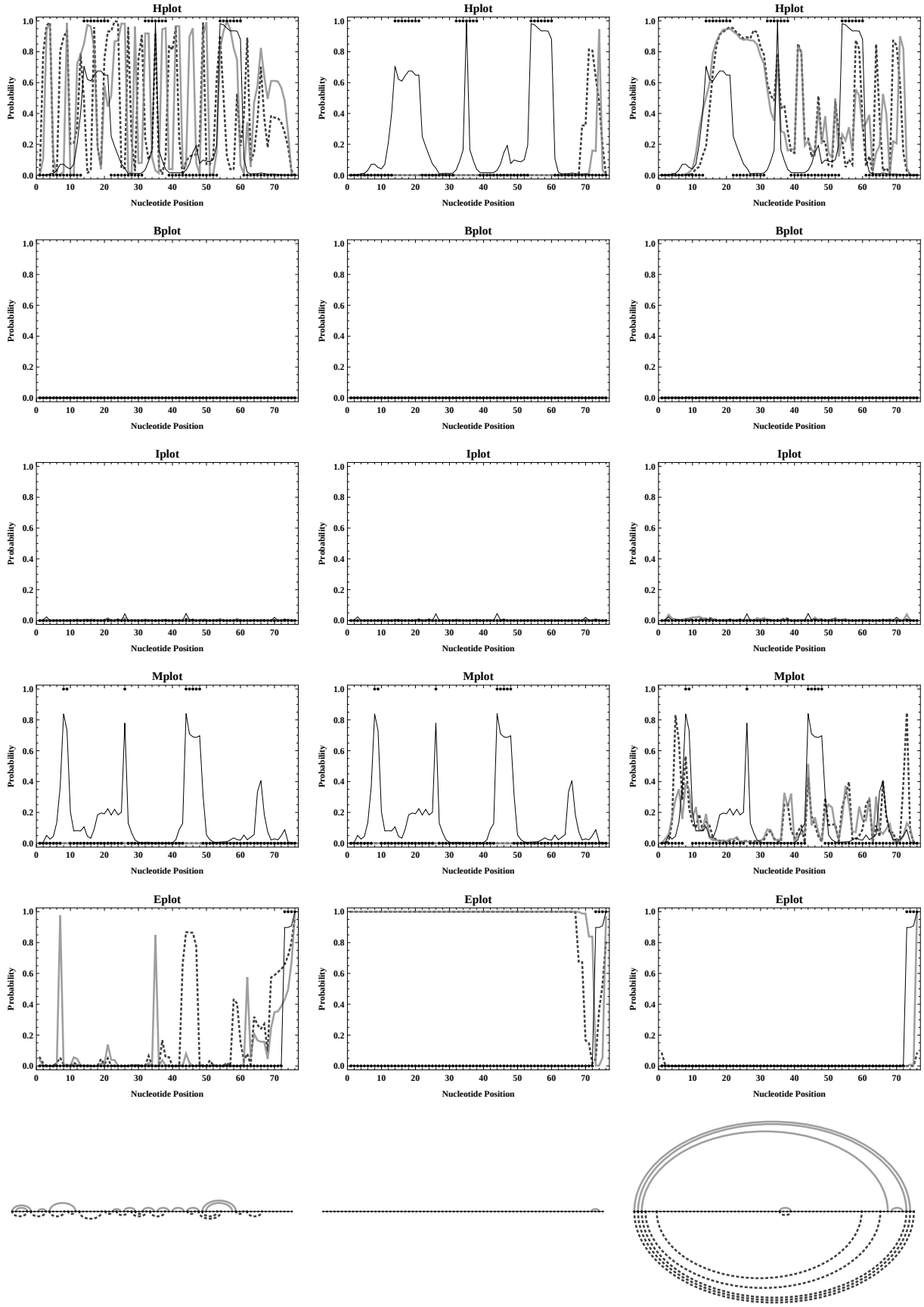


Figure D.7.: **Profiling results for SCFG approach with absolute errors only for** $x \in \{T, C, A\}$**.** Figure shows loop profiles and centroid derived according to $\mathrm{mev}_{\mathcal{J}}(\mathtt{prob})$ (thick gray lines) and $\mathrm{fev}_{\mathcal{J}}(\mathtt{prob})$ (thick dotted darker gray lines) for the traditional SCFG model, respectively, where $\mathtt{prob} = 10^{-9}$ and $\mathcal{J} \in \{\{T\}, \{C\}, \{A\}\}$ (figures from left to right).

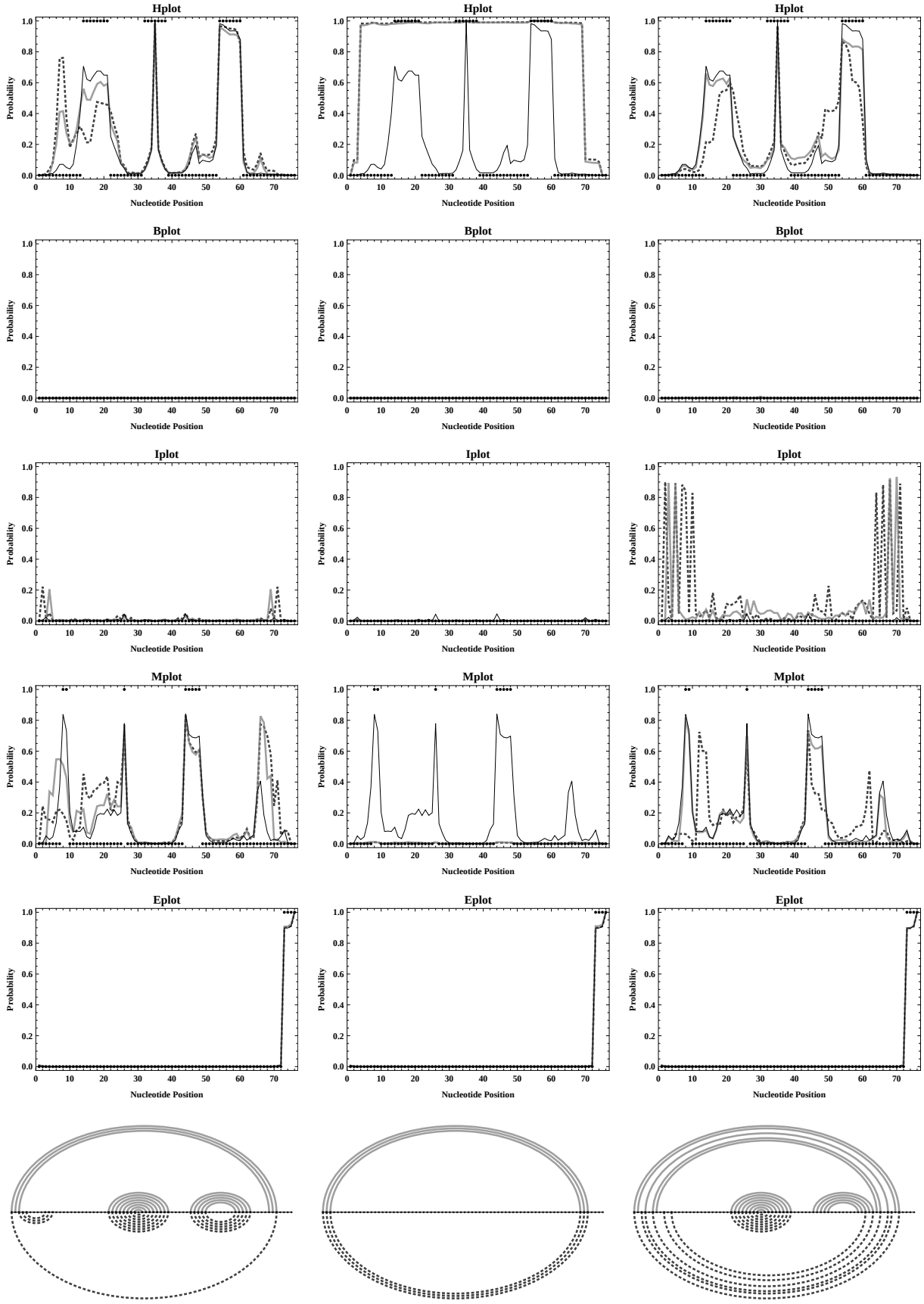


Figure D.8.: **Profiling results for SCFG approach with absolute errors only for** $x \in \{P, F, G\}$**.** Figure shows loop profiles and centroid derived according to $\mathrm{mev}_{\mathcal{J}}(\mathtt{prob})$ (thick gray lines) and $\mathrm{fev}_{\mathcal{J}}(\mathtt{prob})$ (thick dotted darker gray lines) for the traditional SCFG model, respectively, where $\mathtt{prob} = 10^{-9}$ and $\mathcal{J} \in \{\{P\},\{F\},\{G\}\}$ (figures from left to right).

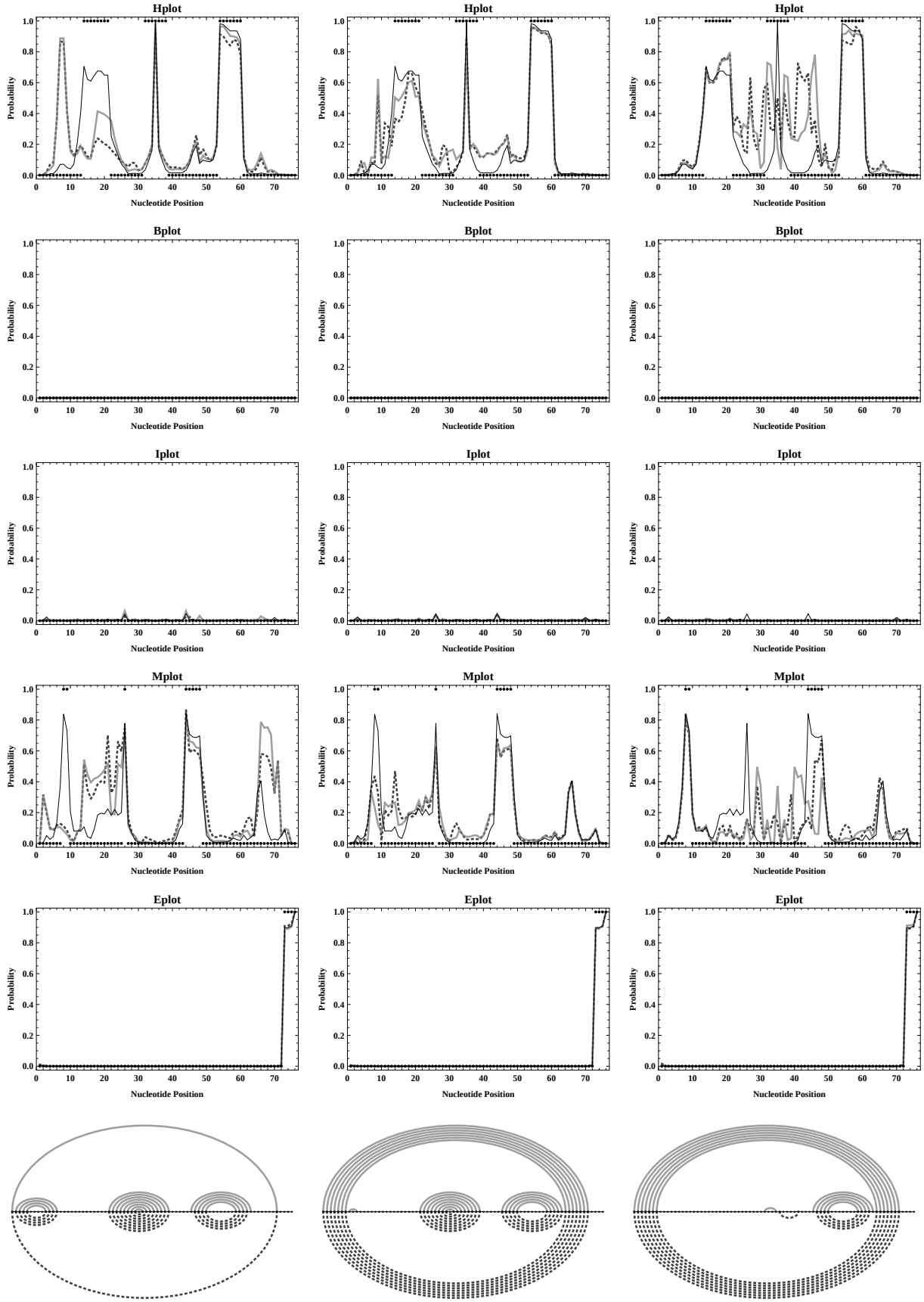


Figure D.9.: **Profiling results for SCFG approach with absolute errors only for** $x \in \{M, O, N\}$**.** Figure shows loop profiles and centroid derived according to $\mathrm{mev}_{\mathcal{J}}(\mathtt{prob})$ (thick gray lines) and $\mathrm{fev}_{\mathcal{J}}(\mathtt{prob})$ (thick dotted darker gray lines) for the traditional SCFG model, respectively, where $\mathtt{prob} = 10^{-9}$ and $\mathcal{J} \in \{\{M\}, \{O\}, \{N\}\}$ (figures from left to right).

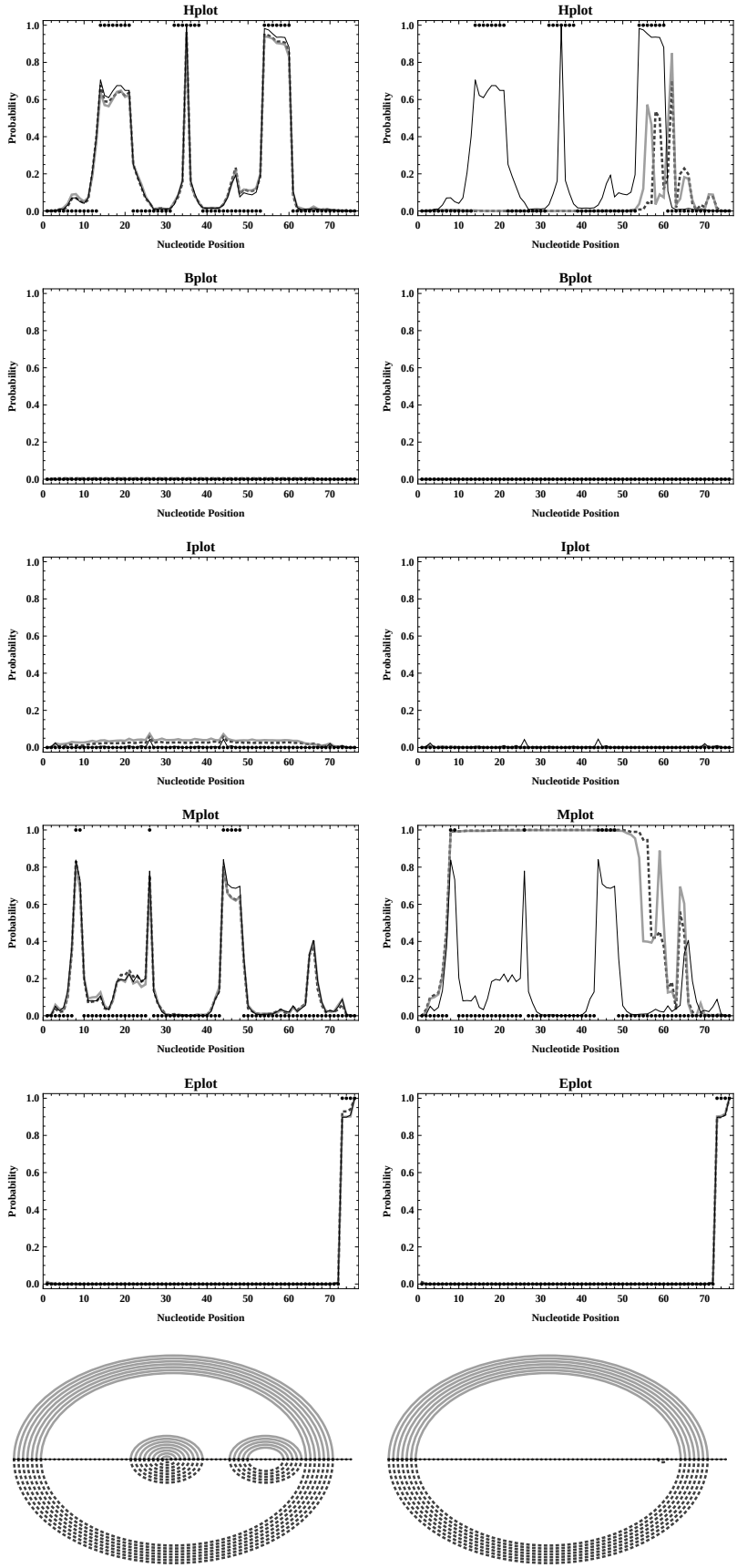


Figure D.10.: **Profiling results for SCFG approach with absolute errors only for** $x \in \{B, U\}$**.** Figure shows loop profiles and centroid derived according to $\mathrm{mev}_{\mathcal{J}}(\mathtt{prob})$ (thick gray lines) and $\mathrm{fev}_{\mathcal{J}}(\mathtt{prob})$ (thick dotted darker gray lines) for the traditional SCFG model, respectively, where $\mathtt{prob} = 10^{-9}$ and $\mathcal{J} \in \{\{B\}, \{U\}\}$ (figures from left to right).

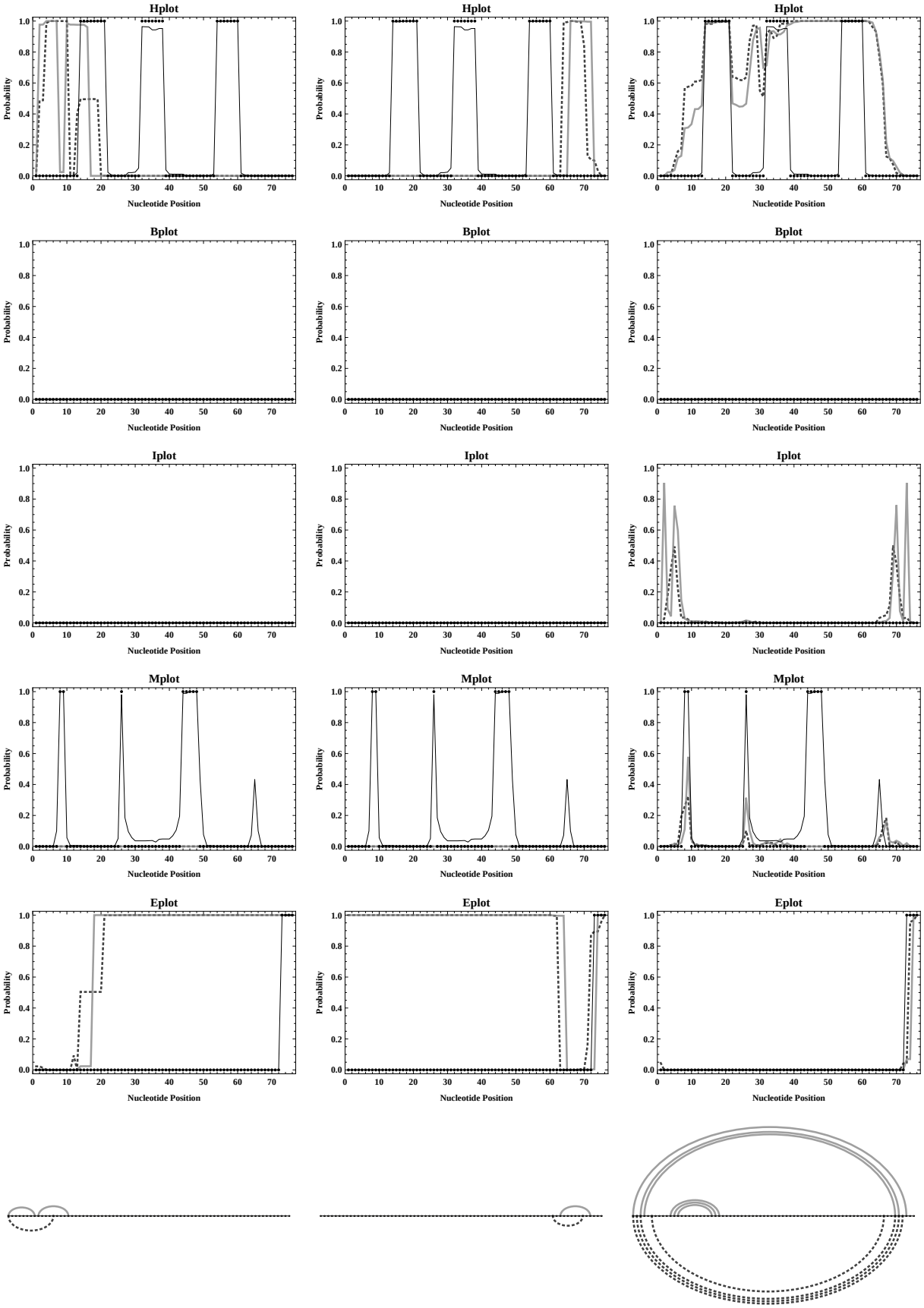


Figure D.11.: **Profiling results for LSCFG approach with absolute errors only for** $x \in \{T, C, A\}$**.** Figure shows loop profiles and centroid derived according to $\text{mev}_{\mathcal{J}}(\texttt{prob})$ (thick gray lines) and $\text{fev}_{\mathcal{J}}(\texttt{prob})$ (thick dotted darker gray lines) for the LSCFG model, respectively, where $\texttt{prob} = 10^{-9}$ and $\mathcal{J} \in \{\{T\}, \{C\}, \{A\}\}$ (figures from left to right).

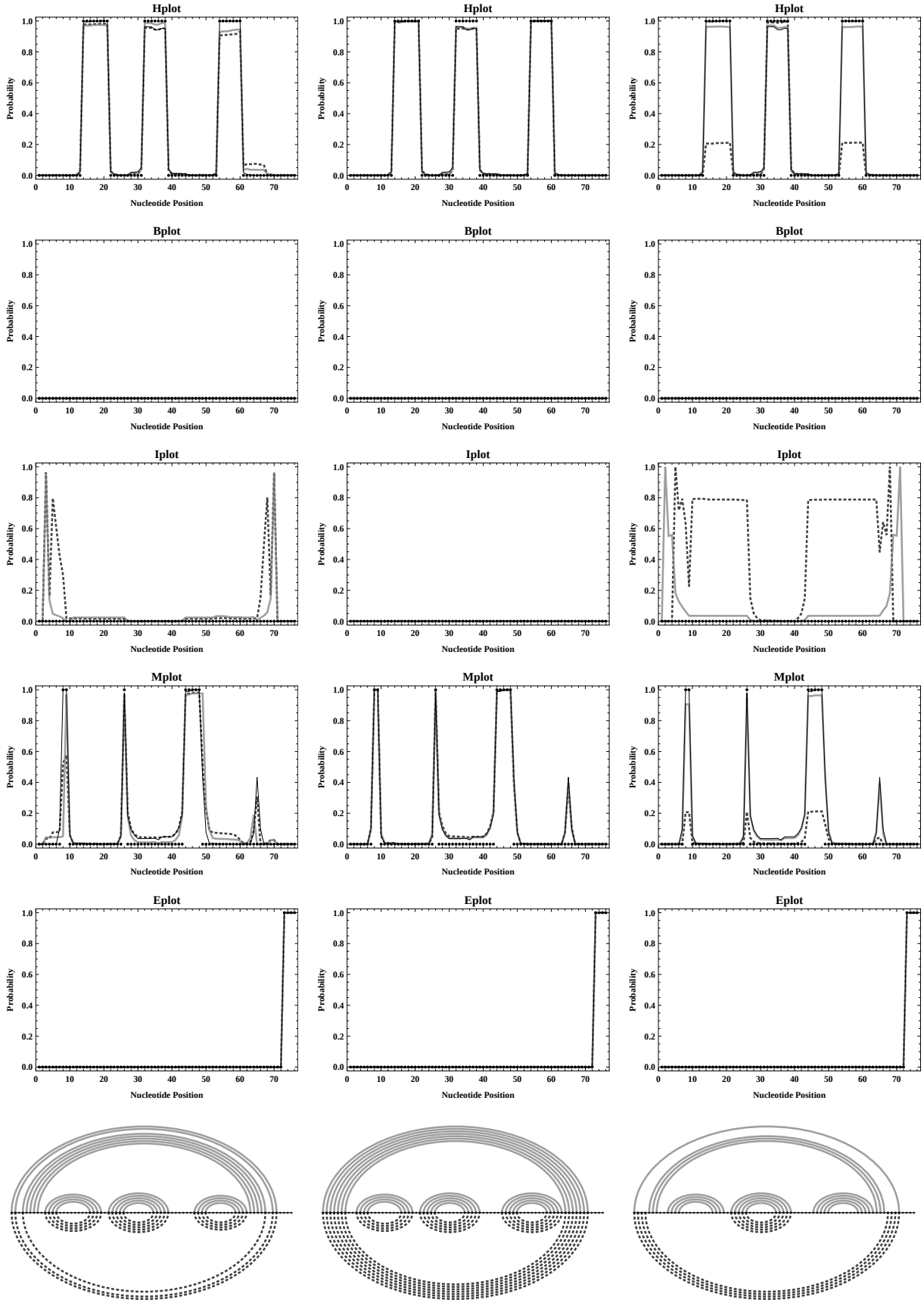


Figure D.12.: **Profiling results for LSCFG approach with absolute errors only for** $x \in \{P, F, G\}$**.** Figure shows loop profiles and centroid derived according to $\mathrm{mev}_{\mathcal{J}}(\mathtt{prob})$ (thick gray lines) and $\mathrm{fev}_{\mathcal{J}}(\mathtt{prob})$ (thick dotted darker gray lines) for the LSCFG model, respectively, where $\mathtt{prob} = 10^{-9}$ and $\mathcal{J} \in \{\{P\}, \{F\}, \{G\}\}$ (figures from left to right).

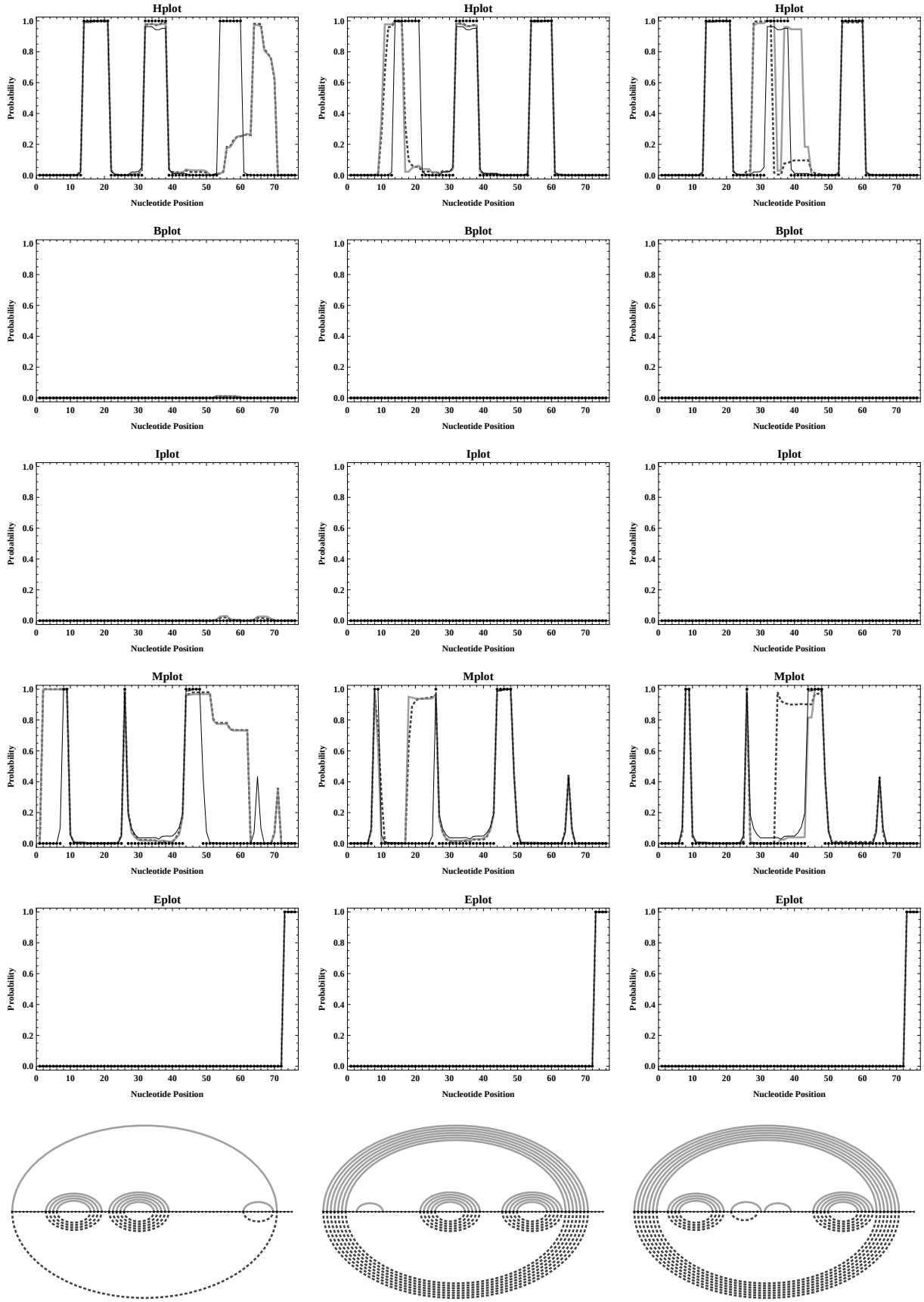


Figure D.13.: **Profiling results for LSCFG approach with absolute errors only for** $x \in \{M, O, N\}$**.** Figure shows loop profiles and centroid derived according to $\mathrm{mev}_{\mathcal{J}}(\mathtt{prob})$ (thick gray lines) and $\mathrm{fev}_{\mathcal{J}}(\mathtt{prob})$ (thick dotted darker gray lines) for the LSCFG model, respectively, where $\mathtt{prob} = 10^{-9}$ and $\mathcal{J} \in \{\{M\},\{O\},\{N\}\}$ (figures from left to right).

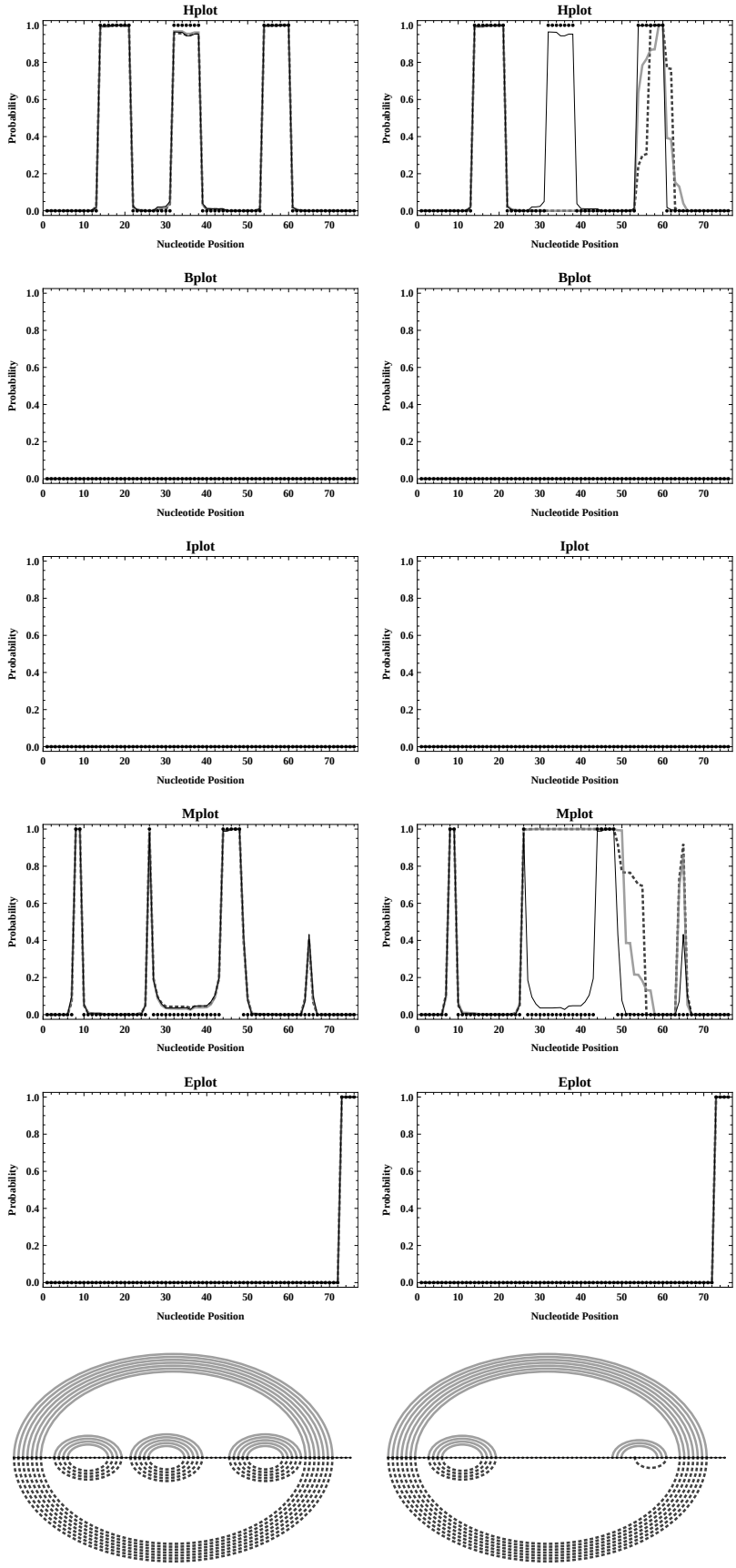
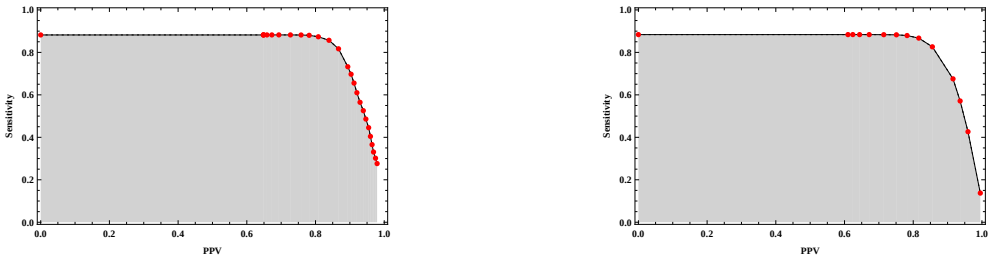
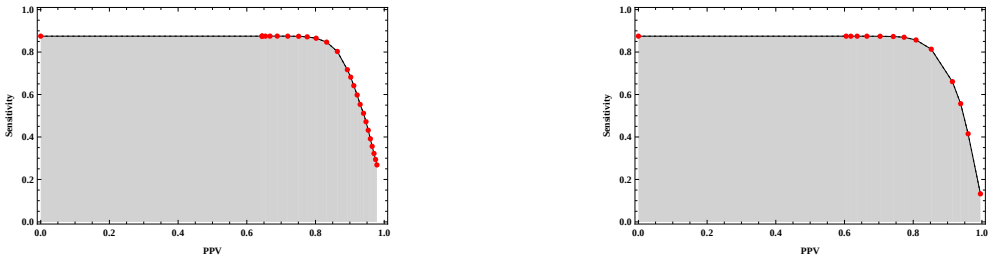


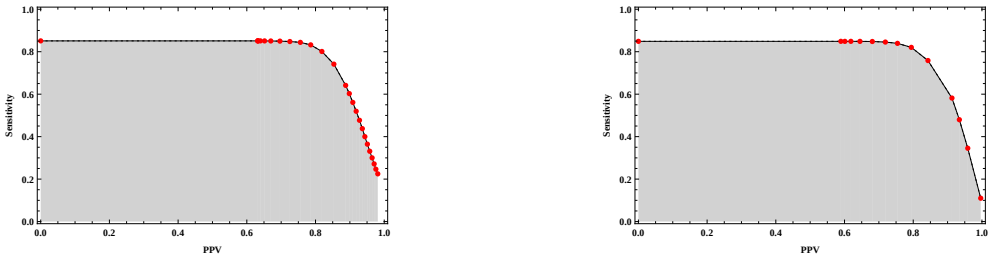
Figure D.14.: **Profiling results for LSCFG approach with absolute errors only for** $x \in \{B, U\}$**.** Figure shows loop profiles and centroid derived according to $\mathrm{mev}_{\mathcal{J}}(\mathtt{prob})$ (thick gray lines) and $\mathrm{fev}_{\mathcal{J}}(\mathtt{prob})$ (thick dotted darker gray lines) for the LSCFG model, respectively, where $\mathtt{prob} = 10^{-9}$ and $\mathcal{J} \in \{\{B\}, \{U\}\}$ (figures from left to right).



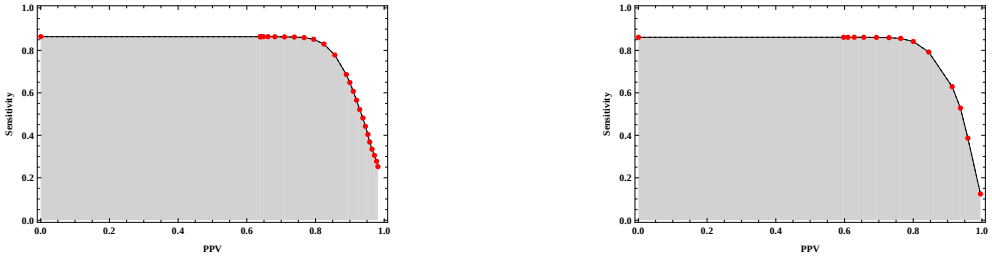
(a) No disturbances.



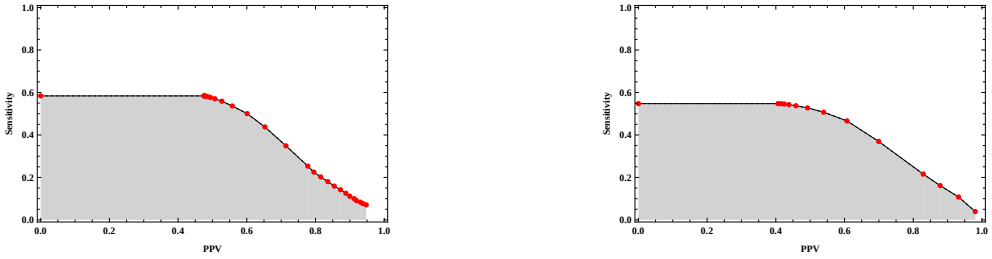
(b) Disturbances according to mep(0.5).



(c) Disturbances according to mep(0.99).

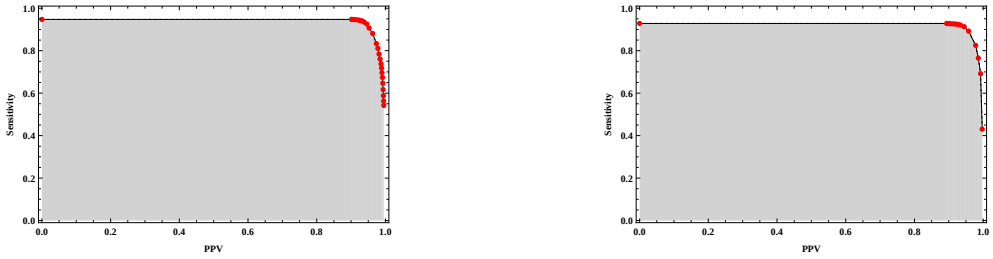


(d) Disturbances according to fep(0.5).

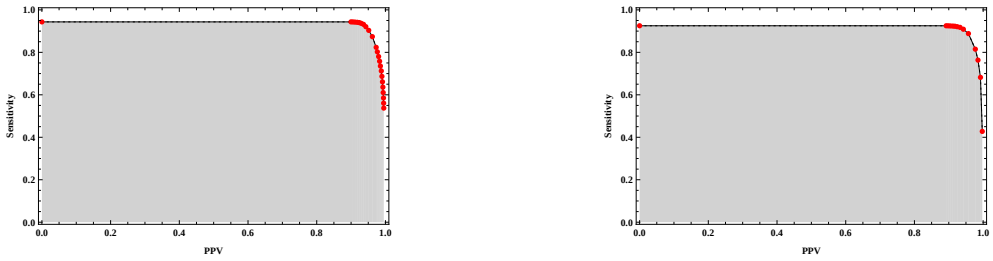


(e) Disturbances according to fep(0.99).

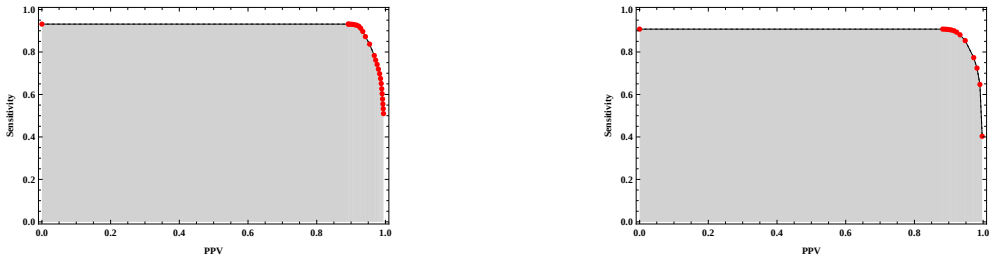
Figure D.15.: **(Areas under) ROC curves for our tRNA database, using the SCFG approach.** Presented curves were derived without disturbances (top line) and by considering random relative disturbances according to mep(0.5), mep(0.99), fep(0.5) and fep(0.99) (from top to bottom line) under the assumption of the traditional SCFG model (for $\min_{\text{hel}} = 1$ and $\min_{\text{HL}} = 1$). For each preprocessing variant, corresponding ROC curves are shown for prediction principle MEA structure (figure on the left) and centroid (figure on the right), respectively.



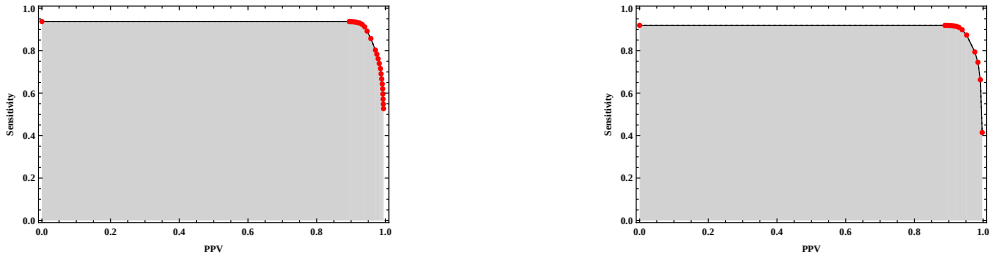
(a) No disturbances.



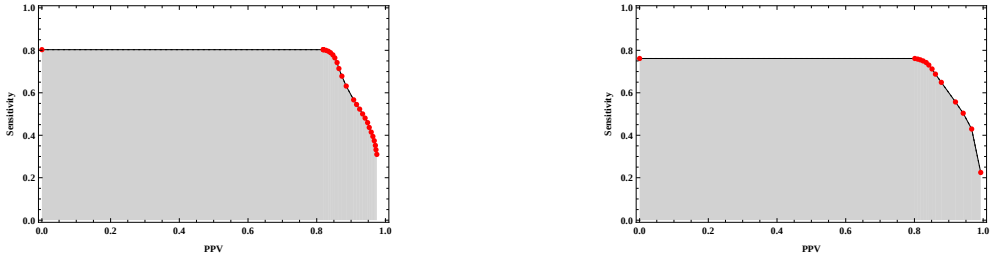
(b) Disturbances according to mep(0.5).



(c) Disturbances according to mep(0.99).

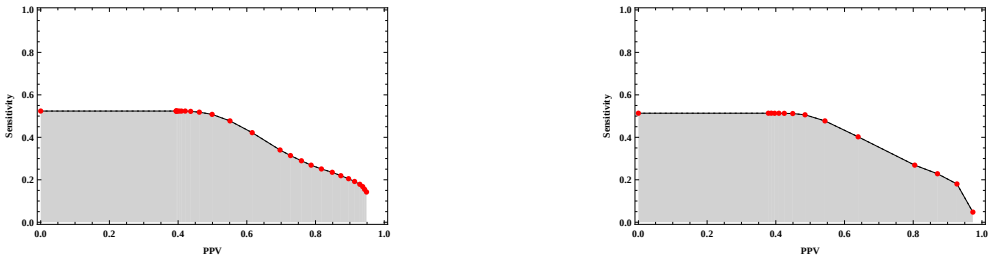


(d) Disturbances according to fep(0.5).

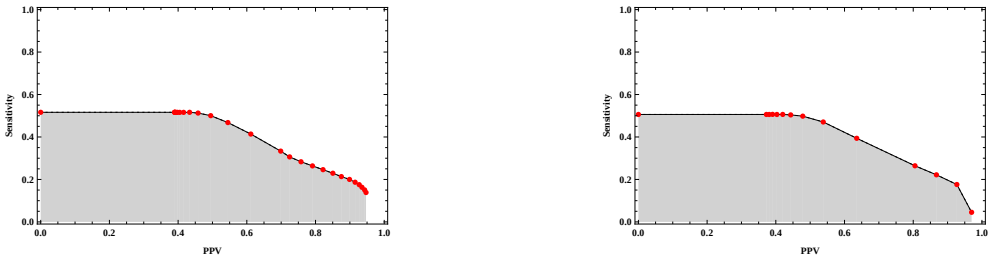


(e) Disturbances according to fep(0.99).

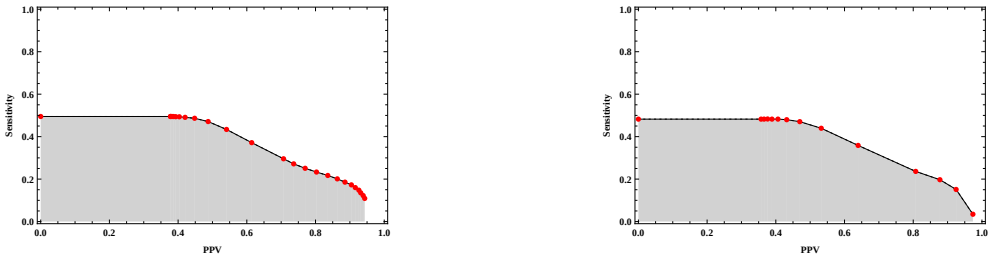
Figure D.16.: **(Areas under) ROC curves for our tRNA database, using the LSCFG approach.** Results correspond to those of Figure B.1, but were derived under the assumption of the LSCFG model.



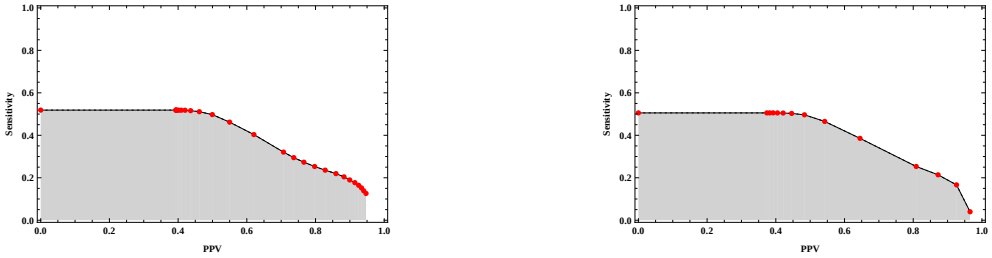
(a) No disturbances.



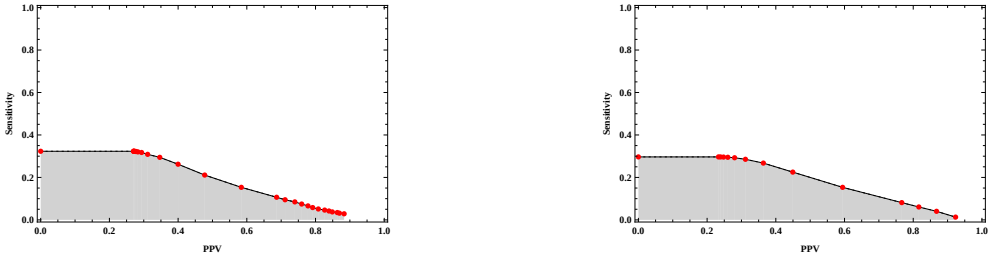
(b) Disturbances according to mep(0.5).



(c) Disturbances according to mep(0.99).

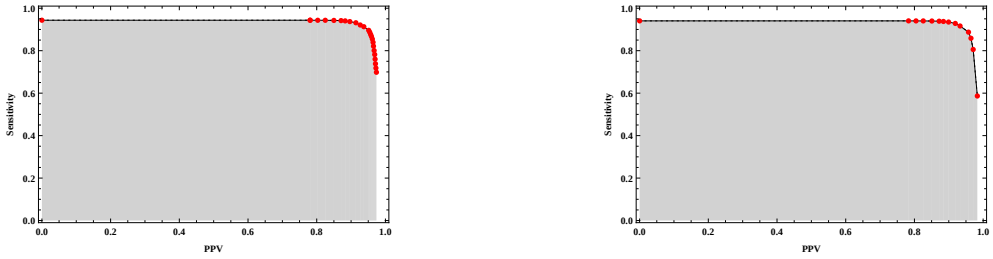


(d) Disturbances according to fep(0.5).

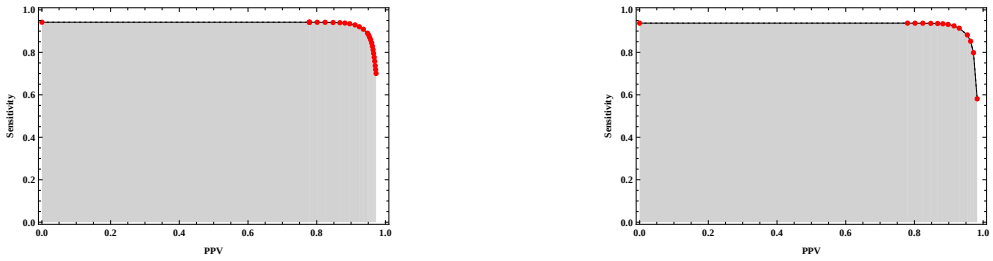


(e) Disturbances according to fep(0.99).

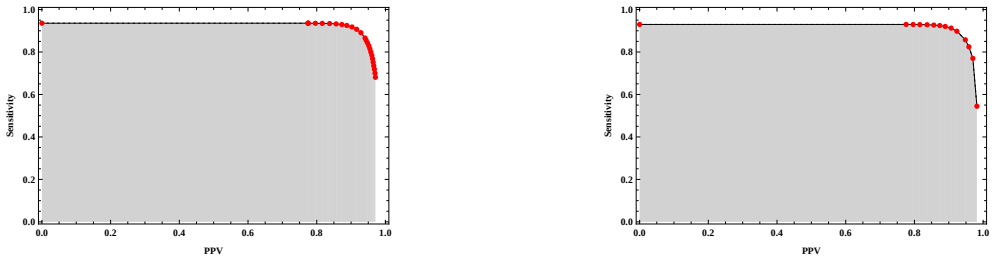
Figure D.17.: **(Areas under) ROC curves for our 5S rRNA database, using the SCFG approach.** Presented curves were derived without disturbances (top line) and by considering random relative disturbances according to mep(0.5), mep(0.99), fep(0.5) and fep(0.99) (from top to bottom line) under the assumption of the traditional SCFG model (for $\min_{\text{hel}} = 1$ and $\min_{\text{HL}} = 1$). For each preprocessing variant, corresponding ROC curves are shown for prediction principle MEA structure (figure on the left) and centroid (figure on the right), respectively.



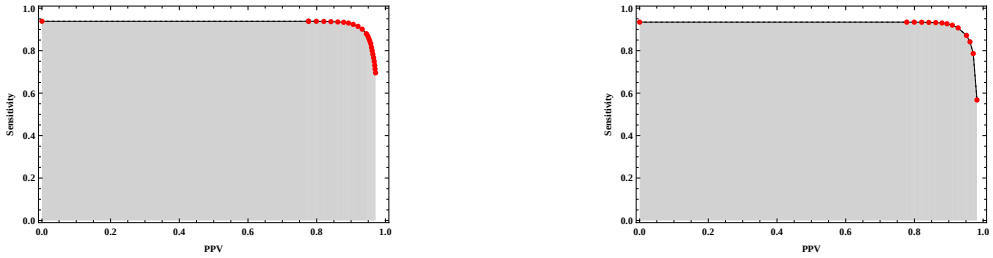
(a) No disturbances.



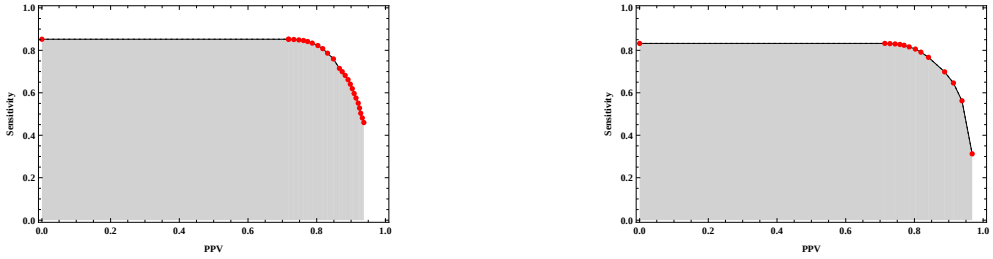
(b) Disturbances according to mep(0.5).



(c) Disturbances according to mep(0.99).



(d) Disturbances according to fep(0.5).



(e) Disturbances according to fep(0.99).

Figure D.18.: **(Areas under) ROC curves for our 5S rRNA database, using the LSCFG approach.** Results corresponding to those of Figure B.2, but were derived under the assumption of the LSCFG model.

Chapter E

Tables and Figures Relating to Chapter 9

X	$\mathtt{card}(\mathcal{X}^e)$	$\mathtt{card}(\mathcal{X}^a)$		$\mathtt{card}(\mathcal{X}^e \cap \mathcal{X}^a)$		$\mathtt{card}(\mathcal{X}^e \setminus \mathcal{X}^a)$		$\mathtt{card}(\mathcal{X}^a \setminus \mathcal{X}^e)$	
		$W_e = 30$	$W_e = -1$	$W_e = 30$	$W_e = -1$	$W_e = 30$	$W_e = -1$	$W_e = 30$	$W_e = -1$
A	70	70	70	70	70	0	0	0	0
B	76	76	76	76	76	0	0	0	0
C	76	76	76	76	76	0	0	0	0
F	74	74	74	74	74	0	0	0	0
G	68	68	68	68	68	0	0	0	0
M	63	63	63	63	63	0	0	0	0
N	77	77	77	77	77	0	0	0	0
O	70	70	70	70	70	0	0	0	0
P	72	72	72	72	72	0	0	0	0
T	76	76	76	76	76	0	0	0	0
U	77	77	77	77	77	0	0	0	0
AT	69	69	69	69	69	0	0	0	0
AB	67	67	67	67	67	0	0	0	0
AO	62	62	62	62	62	0	0	0	0
AN	69	69	69	69	69	0	0	0	0

(a) Traditional SCFG model (for $\min_{\mathrm{HL}} = 3$ and $\min_{\mathrm{hel}} = 2$).

X	$\mathtt{card}(\mathcal{X}^e)$	$\mathtt{card}(\mathcal{X}^a)$		$\mathtt{card}(\mathcal{X}^e \cap \mathcal{X}^a)$		$\mathtt{card}(\mathcal{X}^e \setminus \mathcal{X}^a)$		$\mathtt{card}(\mathcal{X}^a \setminus \mathcal{X}^e)$	
		$W_e = 30$	$W_e = -1$	$W_e = 30$	$W_e = -1$	$W_e = 30$	$W_e = -1$	$W_e = 30$	$W_e = -1$
A	26	26	26	26	26	0	0	0	0
B	26	26	26	26	26	0	0	0	0
C	16	16	16	16	16	0	0	0	0
F	11	11	11	11	11	0	0	0	0
G	17	17	17	17	17	0	0	0	0
M	22	22	22	22	22	0	0	0	0
N	25	25	25	25	25	0	0	0	0
O	23	23	23	23	23	0	0	0	0
P	28	28	28	28	28	0	0	0	0
T	22	22	22	22	22	0	0	0	0
U	24	24	24	24	24	0	0	0	0
AT	44	44	44	44	44	0	0	0	0
AB	49	49	49	49	49	0	0	0	0
AO	29	29	29	29	29	0	0	0	0
AN	57	57	57	57	57	0	0	0	0

(b) LSCFG model (for $\min_{\mathrm{HL}} = 3$ and $\min_{\mathrm{hel}} = 2$).

Table E.1.: **Comparison of relevant inside probabilities for conventional sampling strategy.** Tabulated values are the numbers of relevant inside probabilities (being greater than zero) that are considered by the conventional sampling strategy for *E.coli* $\mathrm{tRNA}^{\mathrm{Ala}}$, where $\mathcal{X}^z := \{j-i+1 \mid 1 \leqslant i, j \leqslant n \text{ and } \alpha_X(j-i+1) \neq 0\}$, with $z = e$ and $z = a$ denoting the exact and approximated values, respectively. Note that for $z = e$, we consider the corresponding averaged value $\alpha_X(\mathtt{dist}) = \sum_{1 \leqslant i,j \leqslant n \mid j-i+1=\mathtt{dist}} \alpha_X(i,j) \cdot \mathtt{card}(\{1 \leqslant i, j \leqslant n \mid j - i + 1 = \mathtt{dist}\}^{-1}$.

$\mathcal{X}$	$\texttt{card}(\mathcal{X}^e)$	$\texttt{card}(\mathcal{X}^a)$		$\texttt{card}(\mathcal{X}^e \cap \mathcal{X}^a)$		$\texttt{card}(\mathcal{X}^e \setminus \mathcal{X}^a)$		$\texttt{card}(\mathcal{X}^a \setminus \mathcal{X}^e)$	
		$W_e = 30$	$W_e = -1$	$W_e = 30$	$W_e = -1$	$W_e = 30$	$W_e = -1$	$W_e = 30$	$W_e = -1$
$\mathcal{T}_C$	76	76	76	76	76	0	0	0	0
$\mathcal{T}_A$	40	40	40	40	40	0	0	0	0
$\mathcal{T}_{CA}$	1326	1326	1326	1326	1326	0	0	0	0
$\mathcal{T}_{AT}$	2058	2058	2058	2058	2058	0	0	0	0
$\mathcal{T}_{CAT}$	2346	2346	2346	2346	2346	0	0	0	0
$\mathcal{AT}$	46425	46425	46425	46425	46425	0	0	0	0
$\mathcal{L}_F$	2485	2485	2485	2485	2485	0	0	0	0
$\mathcal{L}_P$	2299	2299	2299	2299	2299	0	0	0	0
$\mathcal{L}_G$	2204	2204	2211	2204	2204	0	0	0	7
$\mathcal{L}_M$	1880	1880	1891	1880	1880	0	0	0	11
$\mathcal{G}_{BA}$	0	0	0	0	0	0	0	0	0
$\mathcal{G}_{AB}$	0	0	0	0	0	0	0	0	0
$\mathcal{G}_{BAB}$	49634	49634	50116	49634	49634	0	0	0	482
$\mathcal{AB}$	724817	724817	724817	724817	724817	0	0	0	0
$\mathcal{M}_{UAO}$	39156	39156	39711	39156	39156	0	0	0	555
$\mathcal{AO}$	522542	522542	527712	522542	522542	0	0	0	5170
$\mathcal{O}_{UAN}$	39297	39297	39711	39297	39297	0	0	0	414
$\mathcal{N}_{UAN}$	27383	27383	27720	27383	27383	0	0	0	337
$\mathcal{AN}$	536520	536520	536520	536520	536520	0	0	0	0

(a) Traditional SCFG model (for $\min_{HL} = 3$ and $\min_{hel} = 2$).

$\mathcal{X}$	$\texttt{card}(\mathcal{X}^e)$	$\texttt{card}(\mathcal{X}^a)$		$\texttt{card}(\mathcal{X}^e \cap \mathcal{X}^a)$		$\texttt{card}(\mathcal{X}^e \setminus \mathcal{X}^a)$		$\texttt{card}(\mathcal{X}^a \setminus \mathcal{X}^e)$	
		$W_e = 30$	$W_e = -1$	$W_e = 30$	$W_e = -1$	$W_e = 30$	$W_e = -1$	$W_e = 30$	$W_e = -1$
$\mathcal{T}_C$	6	6	6	6	6	0	0	0	0
$\mathcal{T}_A$	0	0	0	0	0	0	0	0	0
$\mathcal{T}_{CA}$	9	9	9	9	9	0	0	0	0
$\mathcal{T}_{AT}$	17	17	17	17	17	0	0	0	0
$\mathcal{T}_{CAT}$	88	150	154	88	88	0	0	62	66
$\mathcal{AT}$	962	962	962	962	962	0	0	0	0
$\mathcal{L}_F$	715	715	715	715	715	0	0	0	0
$\mathcal{L}_P$	347	347	347	347	347	0	0	0	0
$\mathcal{L}_G$	191	191	246	191	191	0	0	0	55
$\mathcal{L}_M$	210	210	210	210	210	0	0	0	0
$\mathcal{G}_{BA}$	0	0	0	0	0	0	0	0	0
$\mathcal{G}_{AB}$	0	0	0	0	0	0	0	0	0
$\mathcal{G}_{BAB}$	1006	1605	1750	1006	1006	0	0	599	744
$\mathcal{AB}$	61567	61567	61567	61567	61567	0	0	0	0
$\mathcal{M}_{UAO}$	2236	3200	3200	2236	2236	0	0	964	964
$\mathcal{AO}$	5585	5585	5585	5585	5585	0	0	0	0
$\mathcal{O}_{UAN}$	5906	8143	8832	5906	5906	0	0	2237	2926
$\mathcal{N}_{UAN}$	4335	5021	8628	4335	4335	0	0	686	4293
$\mathcal{AN}$	22328	22328	22328	22328	22328	0	0	0	0

(b) LSCFG model (for $\min_{HL} = 3$ and $\min_{hel} = 2$).

Table E.2.: **Comparison of relevant sampling probabilities for conventional sampling strategy.** Tabulated values are the numbers of relevant sampling probabilities (being greater than zero) that are considered by the conventional sampling strategy for *E.coli* tRNA^{Ala}, where $\mathcal{X}_y^z := \bigcup_{1 \leqslant i,j \leqslant n} \mathfrak{ac}X_y(i,j)$ and $\mathcal{Y}^z := \bigcup_{1 \leqslant i \leqslant h \leqslant j \leqslant n} \mathfrak{ac}_Y^*(h,j)$, with $z = e$ and $z = e$ denoting the exact and approximated values, respectively.

X	$\mathtt{card}(\mathcal{X}^e)$	$\mathtt{card}(\mathcal{X}^a)$		$\mathtt{card}(\mathcal{X}^e \cap \mathcal{X}^a)$		$\mathtt{card}(\mathcal{X}^e \setminus \mathcal{X}^a)$		$\mathtt{card}(\mathcal{X}^a \setminus \mathcal{X}^e)$	
		$W_e = 30$	$W_e = -1$	$W_e = 30$	$W_e = -1$	$W_e = 30$	$W_e = -1$	$W_e = 30$	$W_e = -1$
A	70	70	70	70	70	0	0	0	0
B	76	76	76	76	76	0	0	0	0
C	76	76	76	76	76	0	0	0	0
F	74	74	74	74	74	0	0	0	0
M	63	63	63	63	63	0	0	0	0
N	77	77	77	77	77	0	0	0	0
O	70	70	70	70	70	0	0	0	0
P	72	72	72	72	72	0	0	0	0
T	76	76	76	76	76	0	0	0	0
U	77	77	77	77	77	0	0	0	0

(a) Traditional SCFG model (for $\min_{HL} = 3$ and $\min_{hel} = 2$).

X	$\mathtt{card}(\mathcal{X}^e)$	$\mathtt{card}(\mathcal{X}^a)$		$\mathtt{card}(\mathcal{X}^e \cap \mathcal{X}^a)$		$\mathtt{card}(\mathcal{X}^e \setminus \mathcal{X}^a)$		$\mathtt{card}(\mathcal{X}^a \setminus \mathcal{X}^e)$	
		$W_e = 30$	$W_e = -1$	$W_e = 30$	$W_e = -1$	$W_e = 30$	$W_e = -1$	$W_e = 30$	$W_e = -1$
A	26	26	26	26	26	0	0	0	0
B	26	26	26	26	26	0	0	0	0
C	16	16	16	16	16	0	0	0	0
F	11	11	11	11	11	0	0	0	0
M	22	22	22	22	22	0	0	0	0
N	25	25	25	25	25	0	0	0	0
O	23	23	23	23	23	0	0	0	0
P	28	28	28	28	28	0	0	0	0
T	22	22	22	22	22	0	0	0	0
U	24	24	24	24	24	0	0	0	0

(b) LSCFG model (for $\min_{HL} = 3$ and $\min_{hel} = 2$).

Table E.3.: **Comparison of relevant inside probabilities for the dynamic sampling strategy.** Tabulated values are the numbers of relevant inside probabilities (being greater than zero) that are considered by the dynamic strategy for *E.coli* $\mathrm{tRNA}^{\mathrm{Ala}}$, where $\mathcal{X}^z := \{j - i + 1 \mid 1 \leqslant i, j \leqslant n \text{ and } \alpha_X(j - i + 1) \neq 0\}$, with $z = e$ and $z = a$ denoting the exact and approximated values, respectively. Note that for $z = e$, we consider the corresponding averaged value $\alpha_X(\mathtt{dist}) = \sum_{1 \leqslant i,j \leqslant n \mid j-i+1=\mathtt{dist}} \alpha_X(i,j) \cdot \mathtt{card}(\{1 \leqslant i, j \leqslant n \mid j - i + 1 = \mathtt{dist}\}^{-1}$.

X	$\mathtt{card}(\mathcal{X}^e)$	$\mathtt{card}(\mathcal{X}^a)$		$\mathtt{card}(\mathcal{X}^e \cap \mathcal{X}^a)$		$\mathtt{card}(\mathcal{X}^e \setminus \mathcal{X}^a)$		$\mathtt{card}(\mathcal{X}^a \setminus \mathcal{X}^e)$	
		$W_e = 30$	$W_e = -1$	$W_e = 30$	$W_e = -1$	$W_e = 30$	$W_e = -1$	$W_e = 30$	$W_e = -1$
A	70	70	70	70	70	0	0	0	0
G	65	64	65	64	65	1	0	0	0
L	70	70	70	70	70	0	0	0	0
M	59	59	59	59	59	0	0	0	0
N	59	59	59	59	59	0	0	0	0
O	59	59	59	59	59	0	0	0	0
T	70	70	70	70	70	0	0	0	0

(a) Traditional SCFG model (for $\min_{HL} = 3$ and $\min_{hel} = 2$).

X	$\mathtt{card}(\mathcal{X}^e)$	$\mathtt{card}(\mathcal{X}^a)$		$\mathtt{card}(\mathcal{X}^e \cap \mathcal{X}^a)$		$\mathtt{card}(\mathcal{X}^e \setminus \mathcal{X}^a)$		$\mathtt{card}(\mathcal{X}^a \setminus \mathcal{X}^e)$	
		$W_e = 30$	$W_e = -1$	$W_e = 30$	$W_e = -1$	$W_e = 30$	$W_e = -1$	$W_e = 30$	$W_e = -1$
A	69	50	69	50	69	19	0	0	0
G	12	12	13	12	12	0	0	0	1
L	26	26	28	26	26	0	0	0	2
M	15	15	17	15	15	0	0	0	2
N	46	30	48	30	46	16	0	0	2
O	49	29	54	29	49	20	0	0	5
T	56	40	66	36	56	20	0	4	10

(b) LSCFG model (for $\min_{HL} = 3$ and $\min_{hel} = 2$).

Table E.4.: **Comparison of relevant outside probabilities for the dynamic sampling strategy.** Tabulated values are the numbers of relevant outside probabilities (being greater than zero) that are considered by the dynamic strategy for *E.coli* $\mathrm{tRNA}^{\mathrm{Ala}}$, where $\mathcal{X}^z := \{j - i + 1 \mid 1 \leqslant i, j \leqslant n \text{ and } \beta_X(j - i + 1) \neq 0\}$, with $z = e$ and $z = a$ denoting the exact and approximated values, respectively. Note that for $z = e$, we consider the corresponding sum $\beta_X(\mathtt{dist}) = \sum_{1 \leqslant i,j \leqslant n | j-i+1=\mathtt{dist}} \beta_X(i, j)$.

$\min_{hel}$	$\mathtt{card}(\mathcal{X}^e)$	$\mathtt{card}(\mathcal{X}^a)$		$\mathtt{card}(\mathcal{X}^e \cap \mathcal{X}^a)$		$\mathtt{card}(\mathcal{X}^e \setminus \mathcal{X}^a)$		$\mathtt{card}(\mathcal{X}^a \setminus \mathcal{X}^e)$	
		$W_e = 30$	$W_e = -1$	$W_e = 30$	$W_e = -1$	$W_e = 30$	$W_e = -1$	$W_e = 30$	$W_e = -1$
1	1558	1558	1558	1558	1558	0	0	0	0
2	1321	1321	1321	1321	1321	0	0	0	0
3	939	939	939	939	939	0	0	0	0
4	754	754	754	754	754	0	0	0	0

(a) Traditional SCFG model.

$\min_{hel}$	$\mathtt{card}(\mathcal{X}^e)$	$\mathtt{card}(\mathcal{X}^a)$		$\mathtt{card}(\mathcal{X}^e \cap \mathcal{X}^a)$		$\mathtt{card}(\mathcal{X}^e \setminus \mathcal{X}^a)$		$\mathtt{card}(\mathcal{X}^a \setminus \mathcal{X}^e)$	
		$W_e = 30$	$W_e = -1$	$W_e = 30$	$W_e = -1$	$W_e = 30$	$W_e = -1$	$W_e = 30$	$W_e = -1$
1	339	351	351	339	339	0	0	12	12
2	123	148	148	123	123	0	0	25	25
3	34	52	52	34	34	0	0	18	18
4	8	14	14	8	8	0	0	6	6

(b) LSCFG model.

Table E.5.: **Comparison of relevant sampling probabilities for the dynamic sampling strategy.** Tabulated values are the numbers of relevant sampling probabilities (being greater than zero) for choosing the first hairpin loop that are considered by the dynamic sampling strategy for input sequence $r = $ *E.coli* $\mathrm{tRNA}^{\mathrm{Ala}}$ (assuming $\min_{HL} = 3$), that is $\mathcal{X}^z := \mathrm{pcHL}(1, |r|, \emptyset, \emptyset)$, with $z = e$ and $z = a$ denoting the exact and approximated values, respectively.

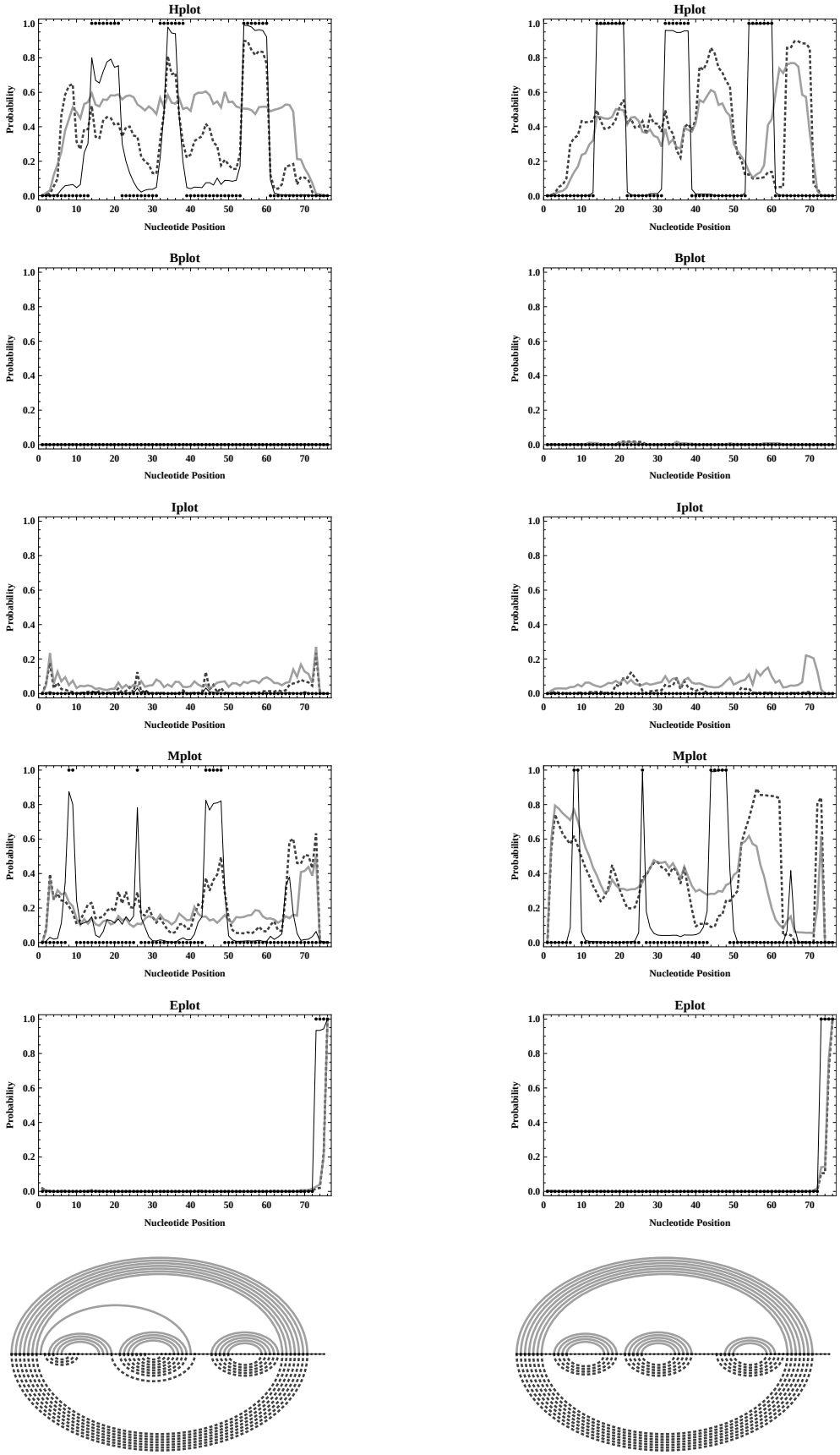


Figure E.1.: **Profiling results derived with the common sampling strategy for** $\min_{\text{hel}} = 1$**.** Figure shows loop profiles and MP structure for *E.coli* tRNA^{Ala}, derived with the common strategy (under the assumption of $\min_{\text{hel}} = 1$ and $\min_{\text{HL}} = 3$) on the basis of the traditional SCFG model (figures on the left) and the LSCFG model (figures on the right), where we used sample size 100 000, 10 000 and 1 000 for $W_e = -1$ (no window, thick gray lines), $W_e = 30$ (moderate window, thick dotted darker gray lines) and $W_e = +\infty$ (complete window, thin black lines).

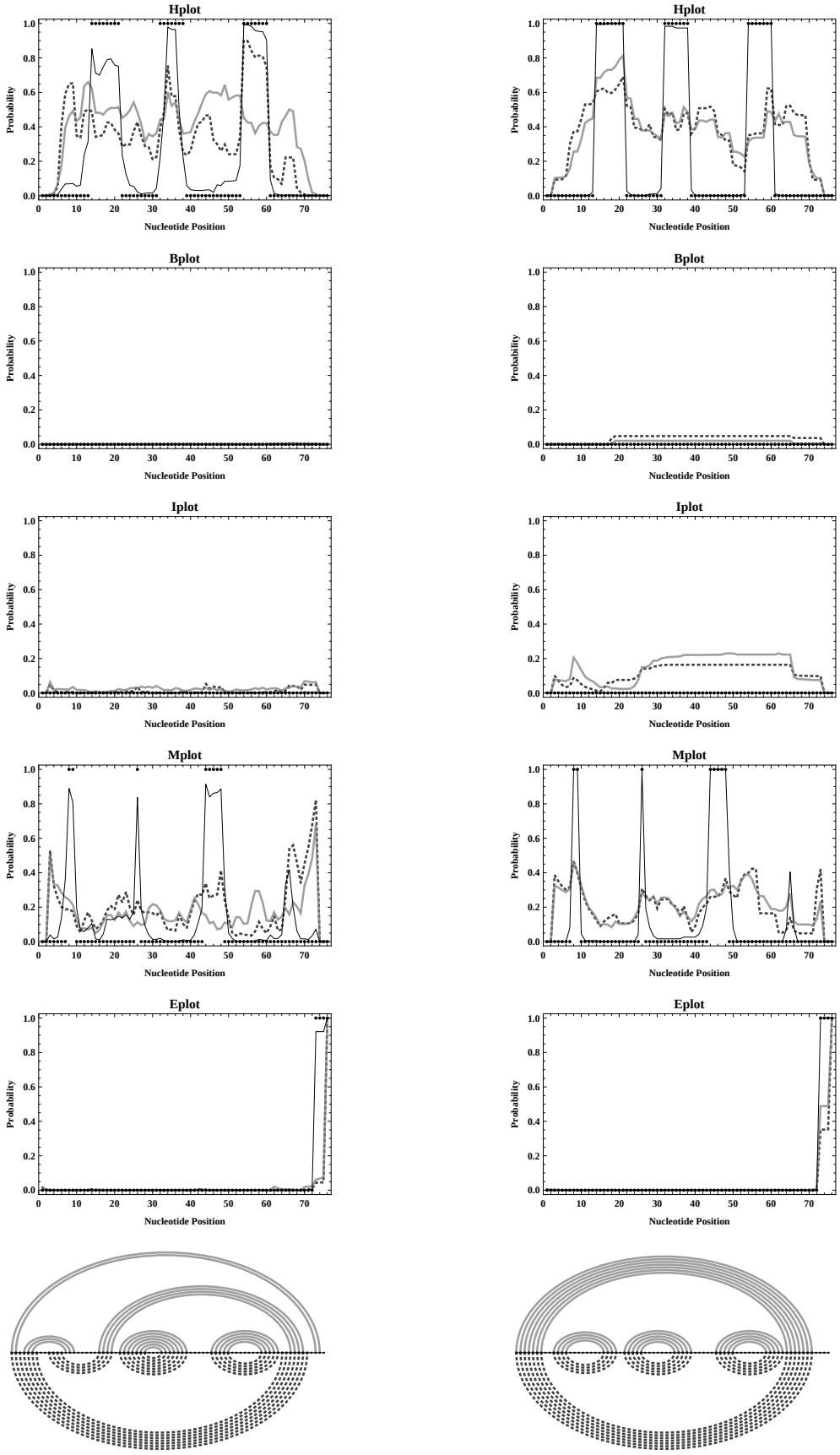


Figure E.2.: **Profiling results derived with the common sampling strategy for** $\min_{\text{hel}} = 2$**.** Figure shows loop profiles and MP structure corresponding to those of Figure E.1, derived with the common strategy for $\min_{\text{hel}} = 2$ and $\min_{\text{HL}} = 3$.

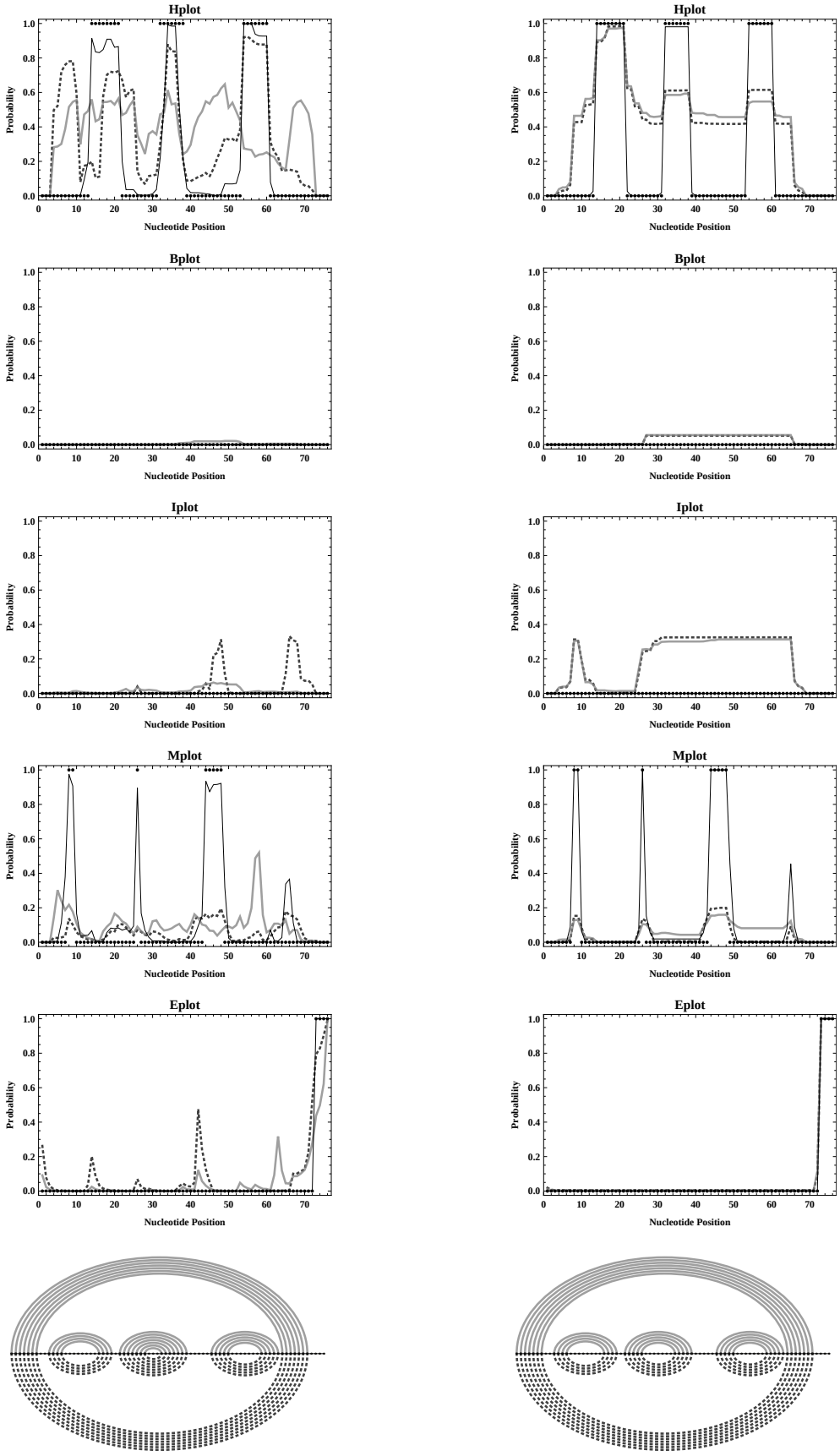


Figure E.3.: **Profiling results derived with the common sampling strategy for** $\min_{\text{hel}} = 3$**.** Figure shows loop profiles and MP structure corresponding to those of Figure E.1, derived with the common strategy for $\min_{\text{hel}} = 3$ and $\min_{\text{HL}} = 3$.

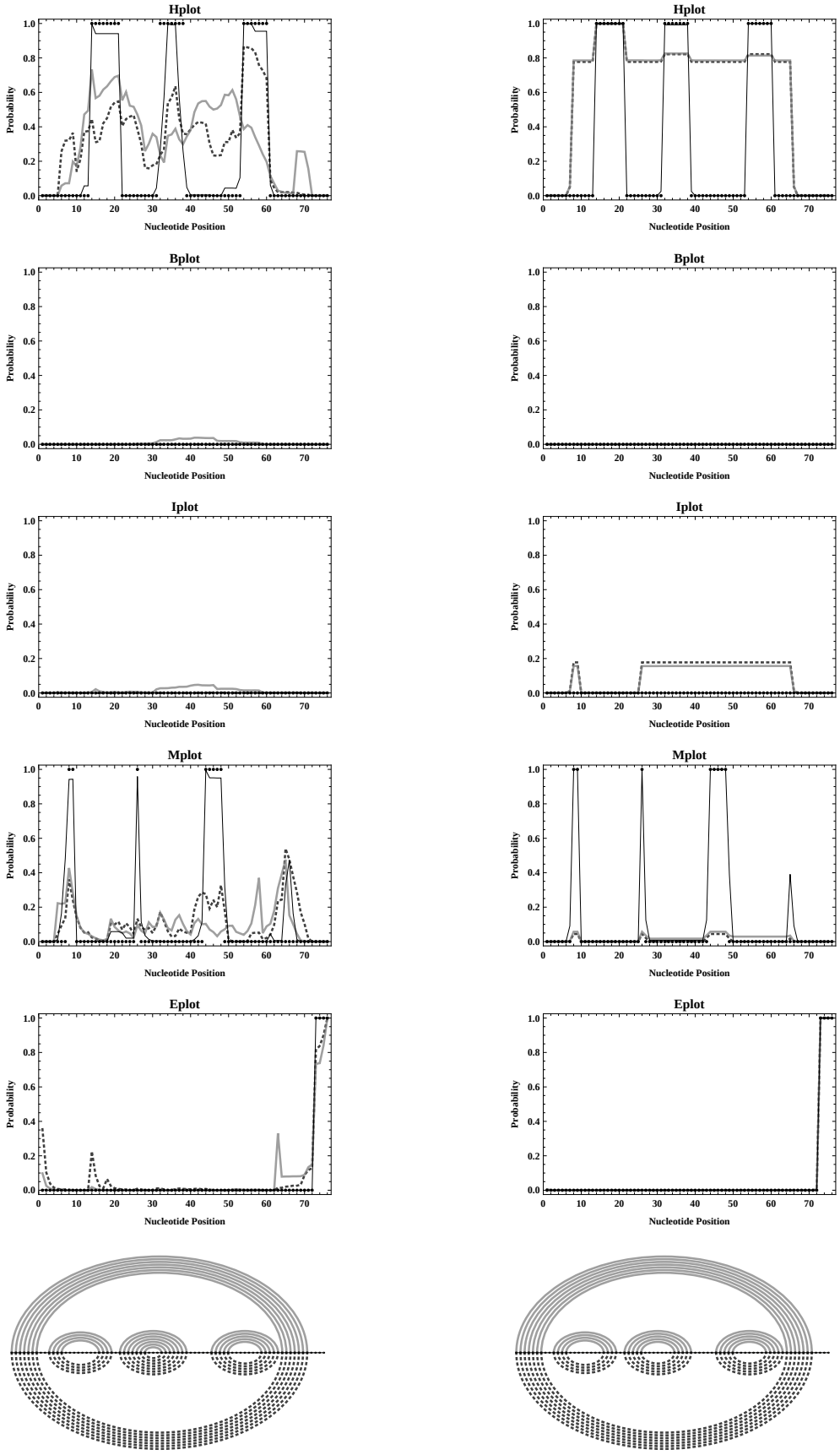


Figure E.4.: **Profiling results derived with the common sampling strategy for** $\min_{\text{hel}} = 4$**.** Figure shows loop profiles and MP structure corresponding to those of Figure E.1, derived with the common strategy for $\min_{\text{hel}} = 4$ and $\min_{\text{HL}} = 3$.

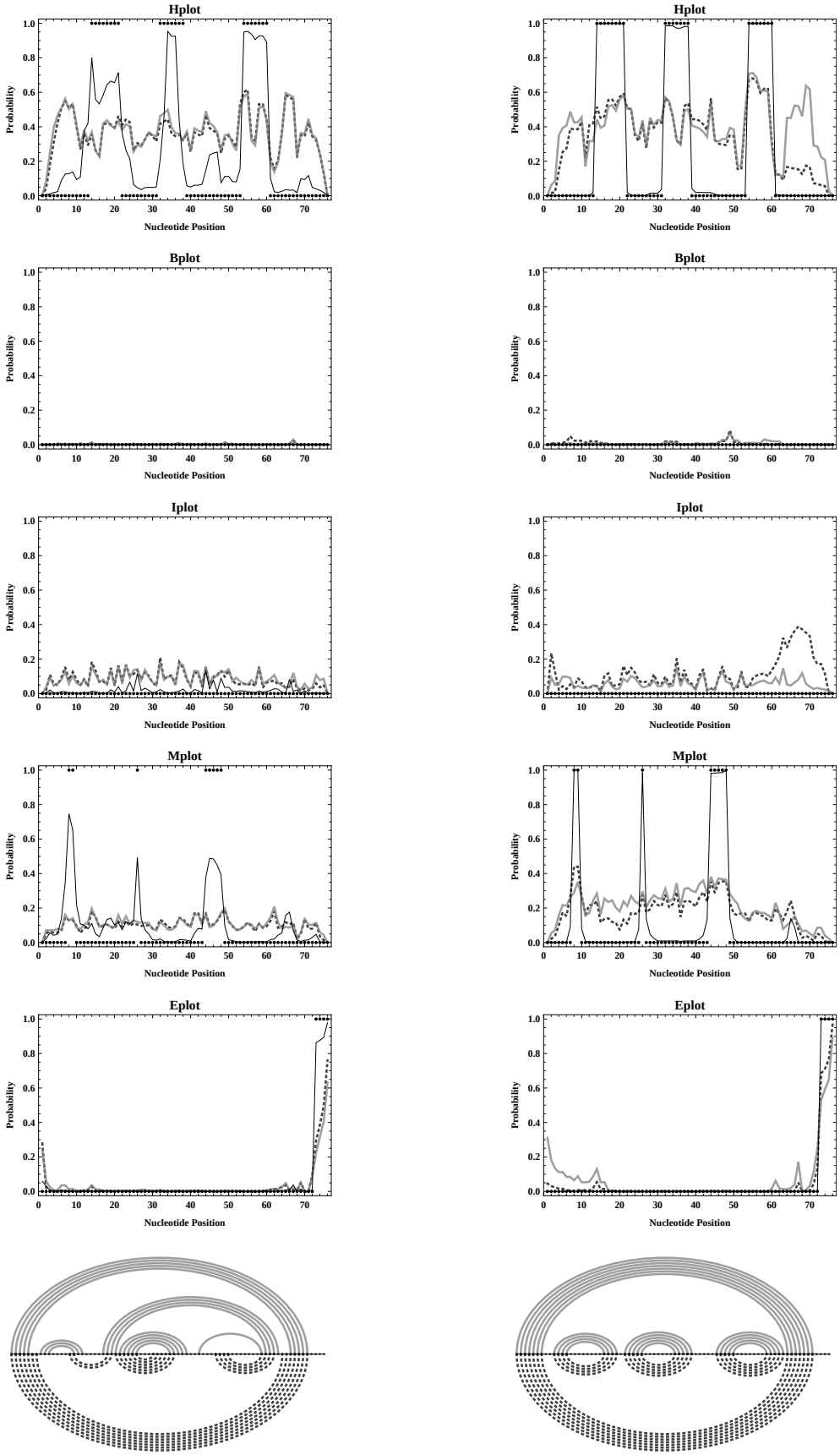


Figure E.5.: **Profiling results derived with the alternative sampling strategy for** $\min_{\text{hel}} = 1$**.** Figure shows loop profiles and MP structure corresponding to those of Figure E.1, obtained by employing the alternative sampling strategy (under the assumption of $\min_{\text{hel}} = 1$ and $\min_{\text{HL}} = 3$) on the basis of the traditional SCFG model (figures on the left) and the LSCFG model (figures on the right), where the common restrictions $\max_{\texttt{hairpin}} = \max_{\texttt{bulge}} = \max_{\texttt{strand}} = 30$ are considered in any case.

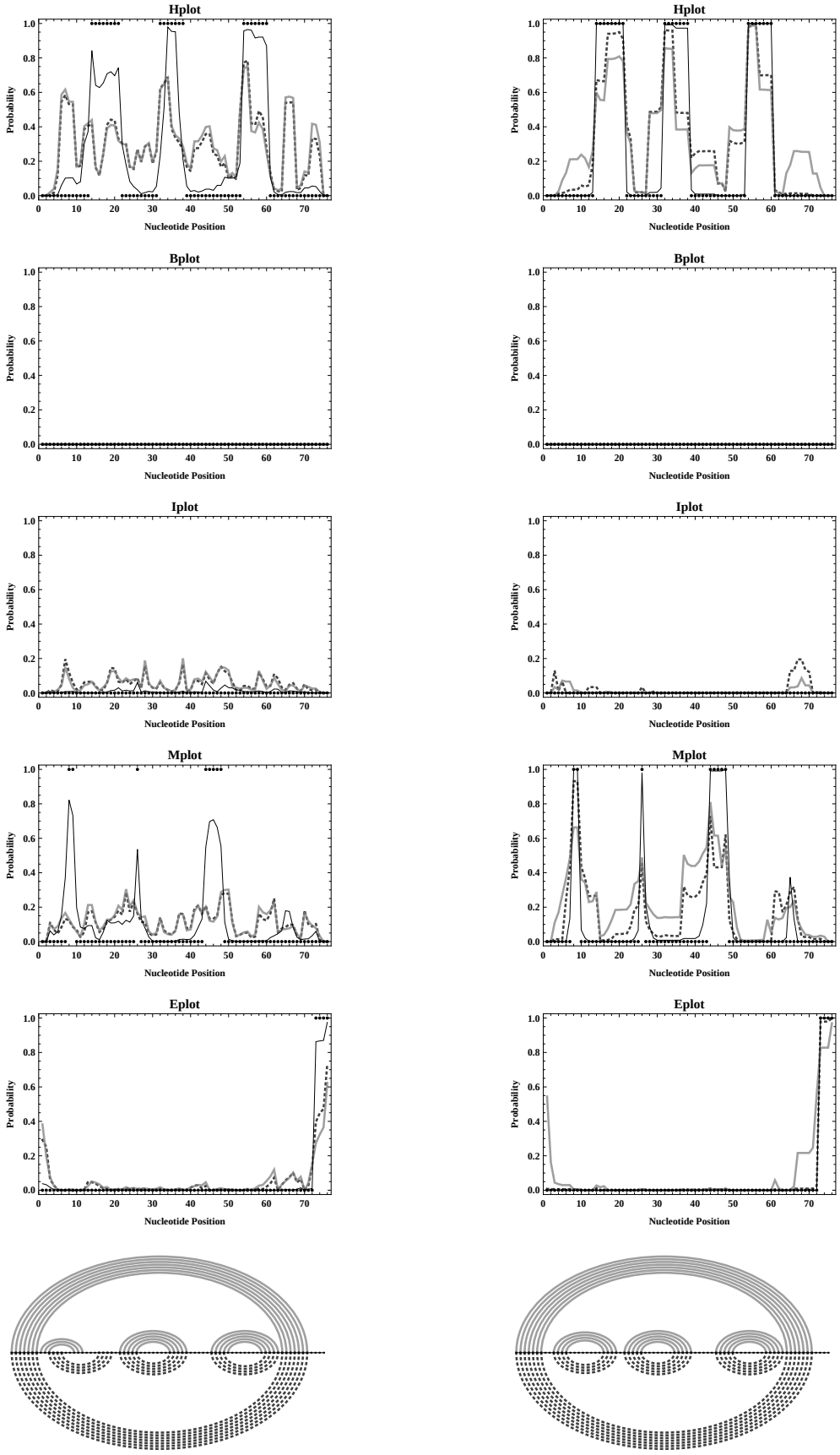


Figure E.6.: **Profiling results derived with the alternative sampling strategy for** $\min_{\text{hel}} = 2$**.** Figure shows loop profiles and MP structure corresponding to those of Figure E.5, obtained with the alternative sampling strategy for $\min_{\text{hel}} = 2$ and $\min_{\text{HL}} = 3$.

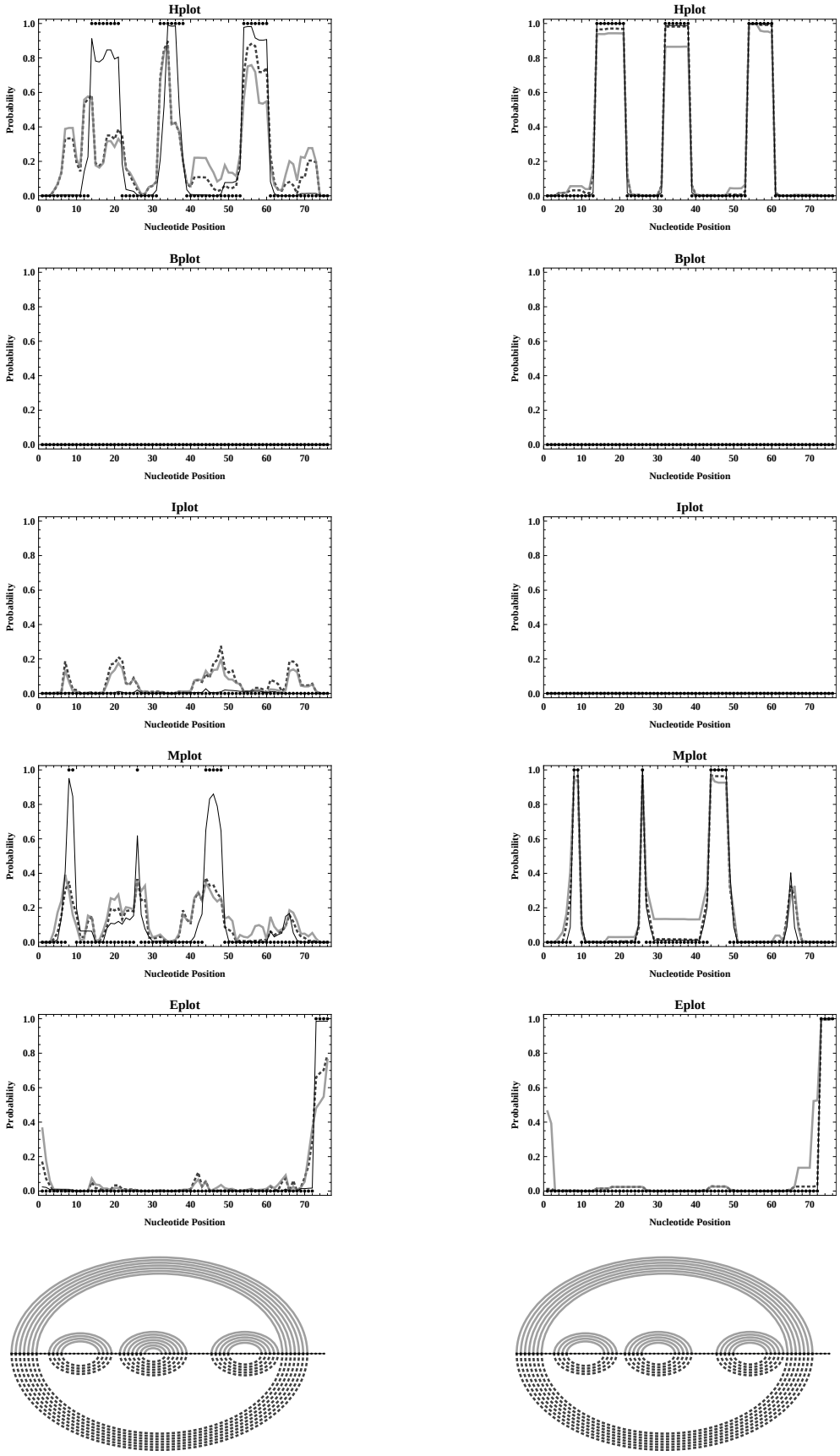


Figure E.7.: **Profiling results derived with the alternative sampling strategy for** $\min_{\text{hel}} = 3$**.** Figure shows loop profiles and MP structure corresponding to those of Figure E.5, obtained with the alternative sampling strategy for $\min_{\text{hel}} = 3$ and $\min_{\text{HL}} = 3$.

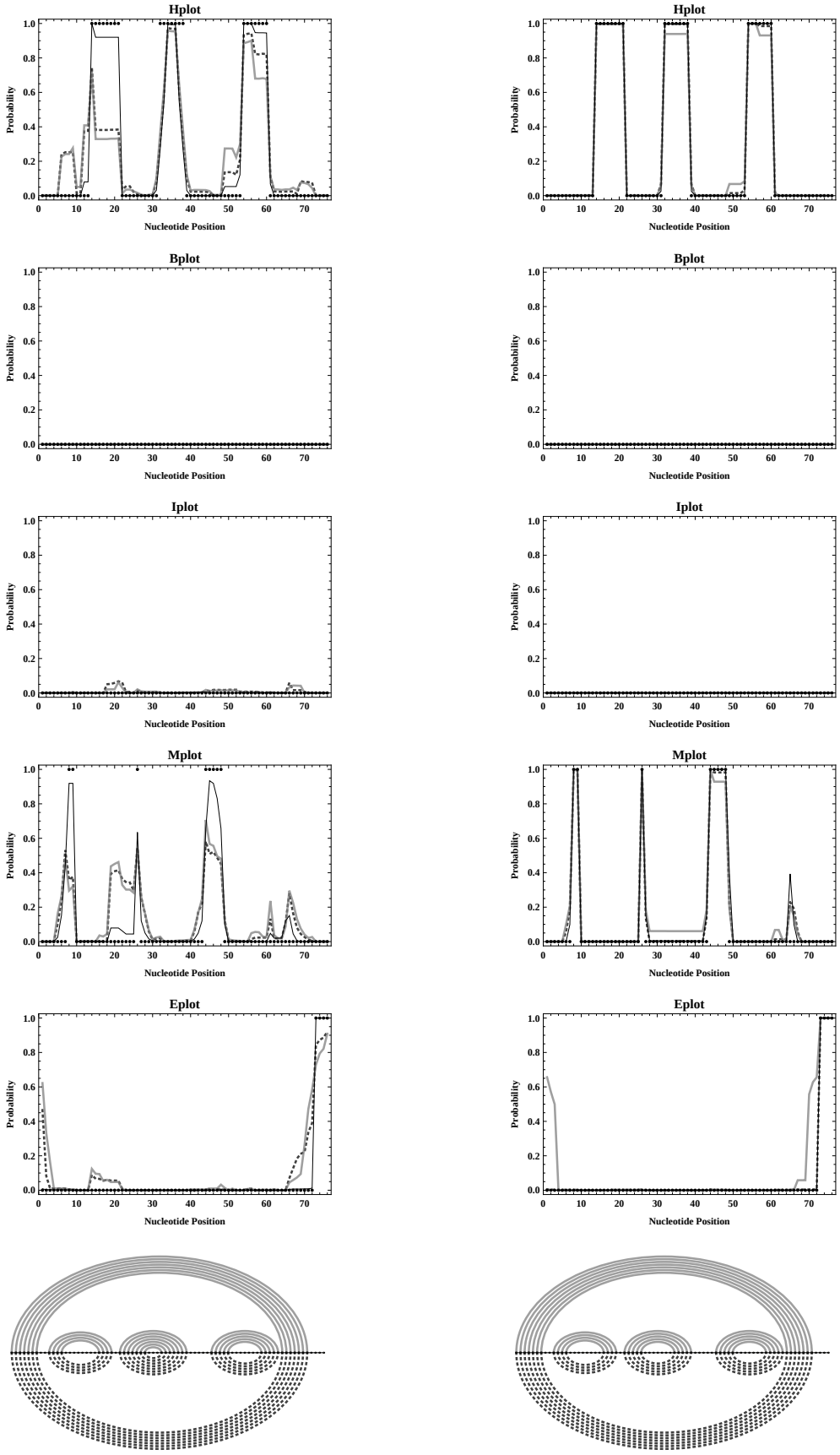
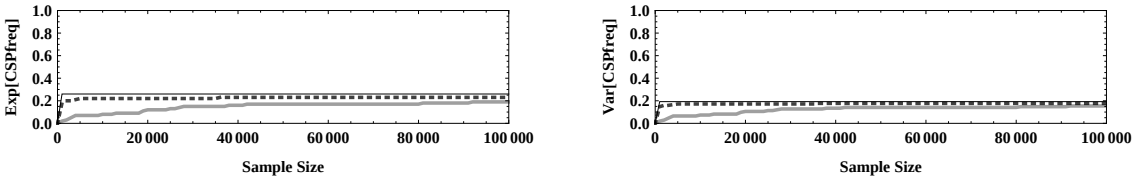
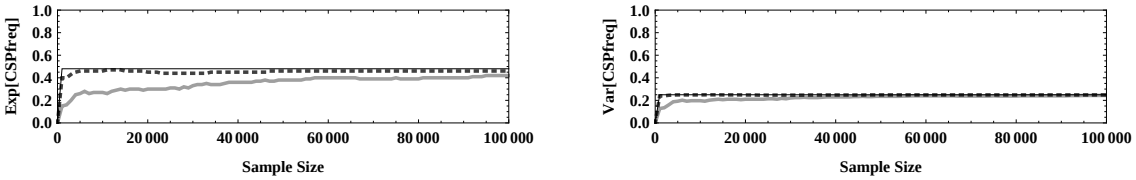


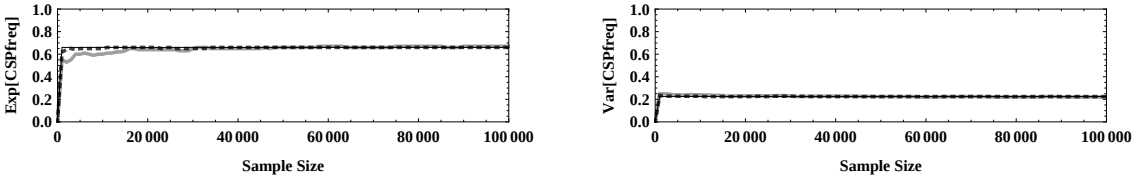
Figure E.8.: **Profiling results derived with the alternative sampling strategy for** $\min_{\text{hel}} = 4$**.** Figure shows loop profiles and MP structure corresponding to those of Figure E.5, obtained with the alternative sampling strategy for $\min_{\text{hel}} = 4$ and $\min_{\text{HL}} = 3$.



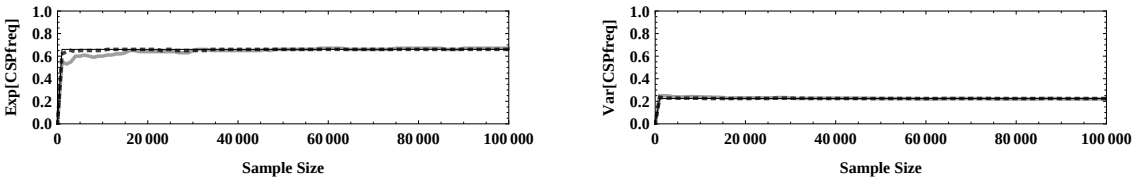
(a) Level 0.



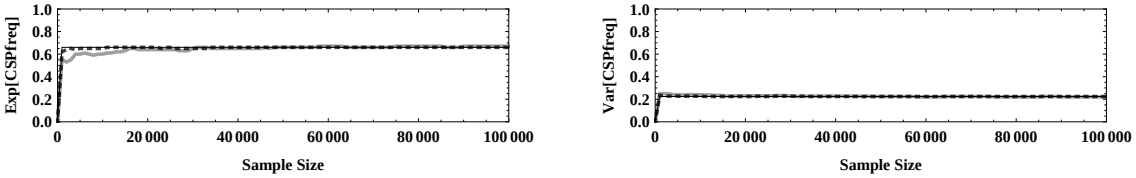
(b) Level 1.



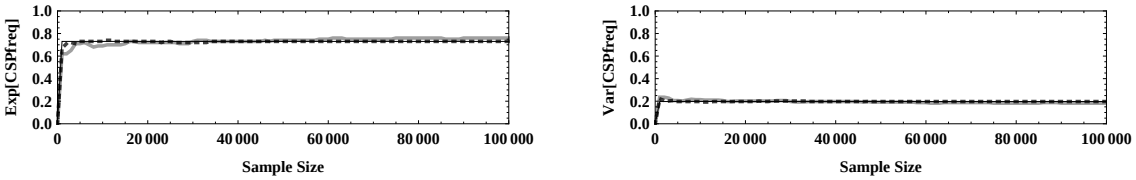
(c) Level 2.



(d) Level 3.

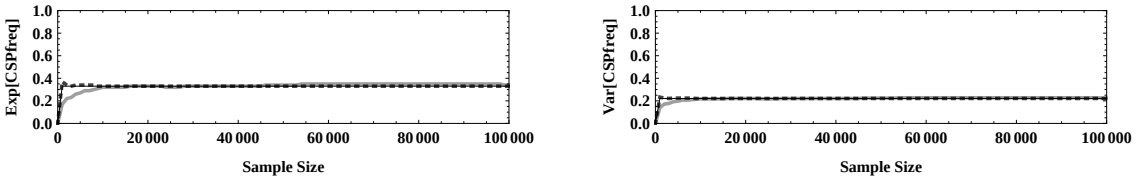


(e) Level 4.

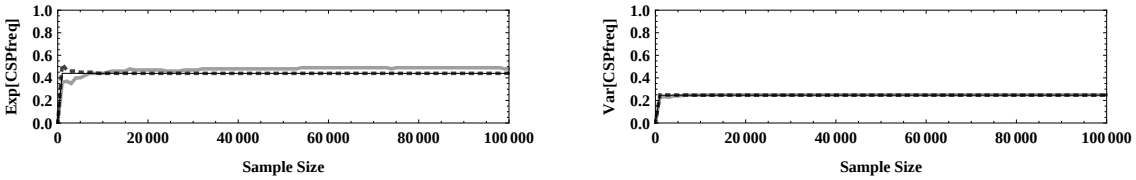


(f) Level 5.

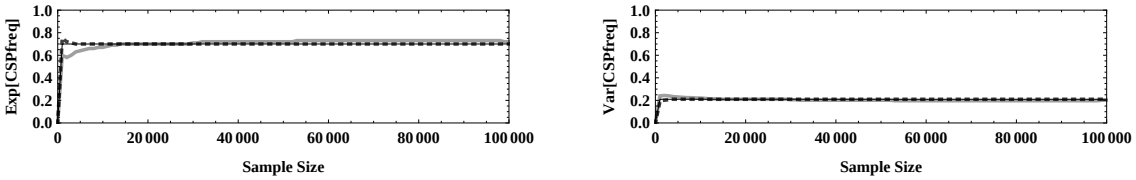
Figure E.9.: **Shape results for tRNA benchmark, with SCFG model and common strategy.** Figures show averaged CSP_{freq} values for MP predictions as functions of sample size (left) and corresponding variances (right), derived on the basis of our tRNA benchmark set by considering the traditional SCFG model and employing the common sampling strategy. Plots correspond to those presented in Figures 9.7a and 9.7b.



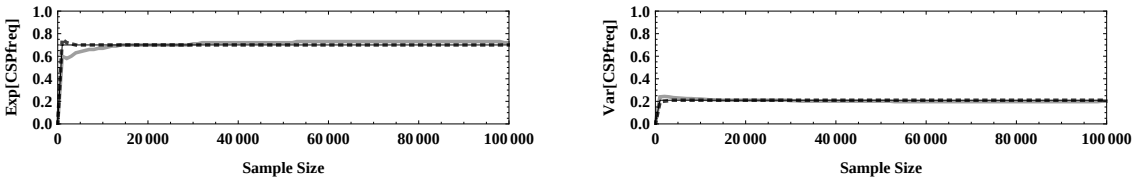
(a) Level 0.



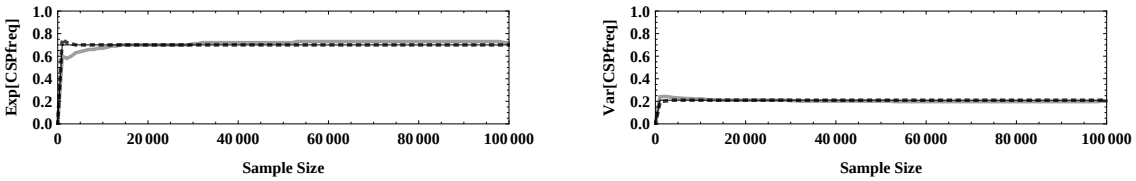
(b) Level 1.



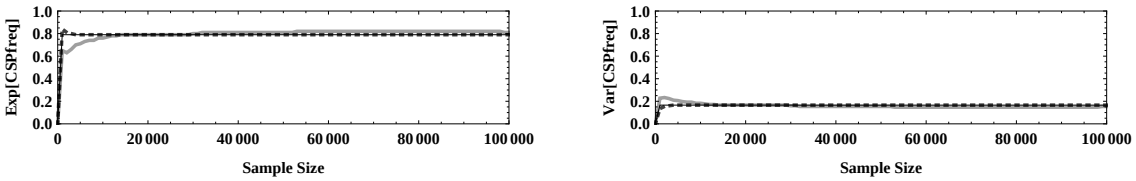
(c) Level 2.



(d) Level 3.

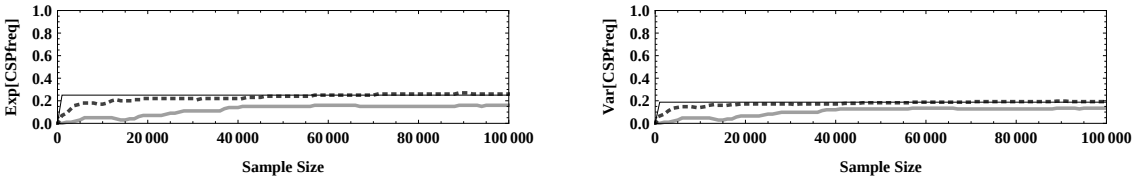


(e) Level 4.

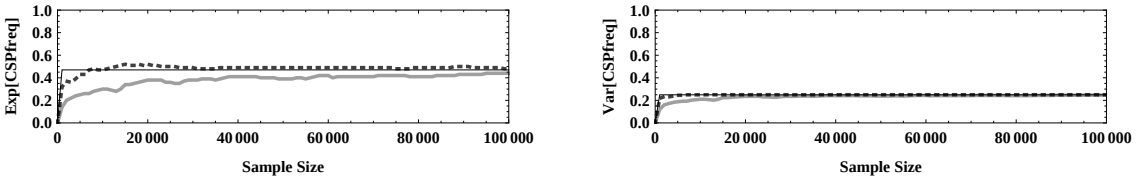


(f) Level 5.

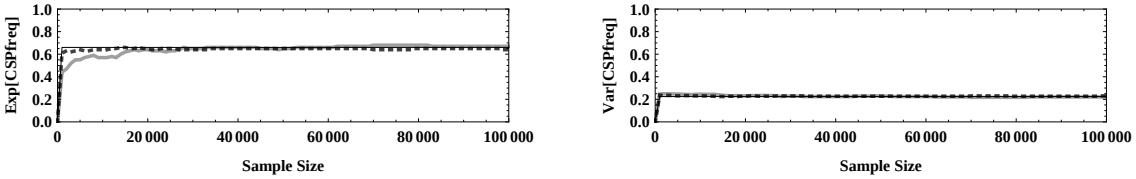
Figure E.10.: **Shape results for tRNA benchmark, with LSCFG model and common strategy.** Figures show averaged CSP_{freq} values for MP predictions as functions of sample size (left) and corresponding variances (right), derived on the basis of our tRNA benchmark set by considering the LSCFG model and employing the common sampling strategy. Plots correspond to those presented in Figures 9.7c and 9.7d.



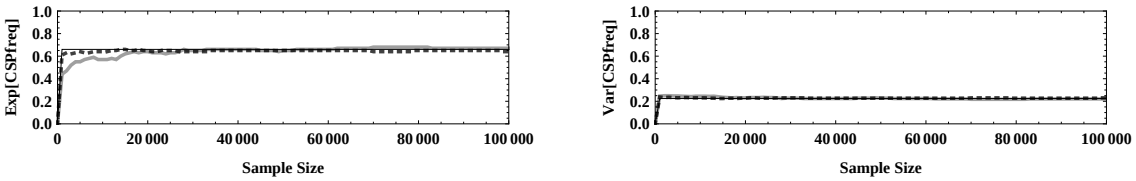
(a) Level 0.



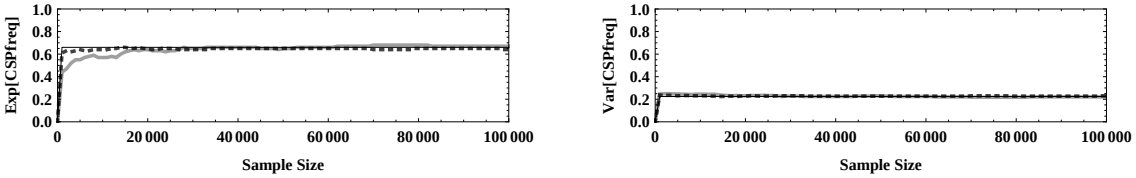
(b) Level 1.



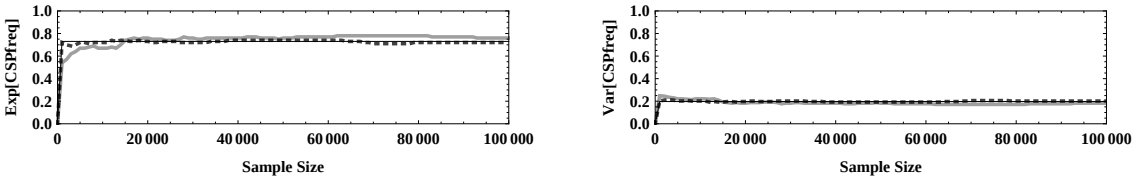
(c) Level 2.



(d) Level 3.

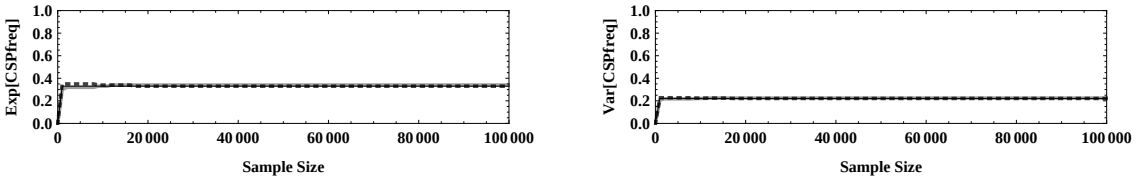


(e) Level 4.

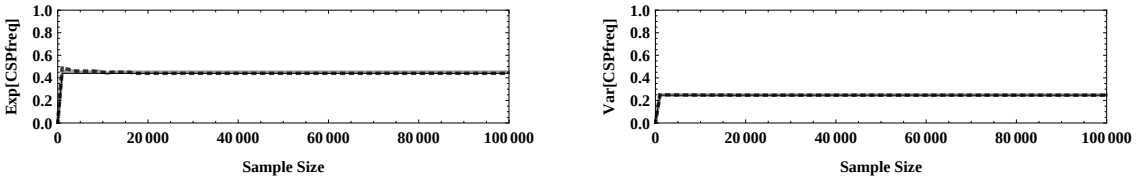


(f) Level 5.

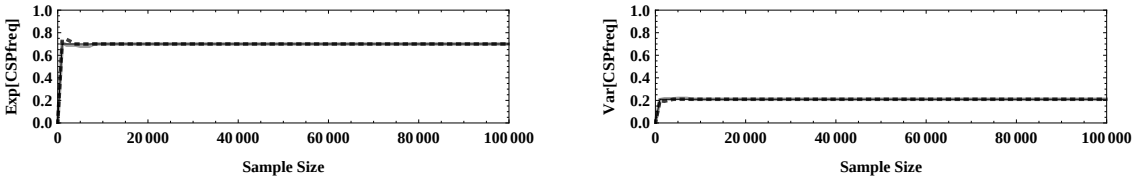
Figure E.11.: **Shape results for tRNA benchmark, with SCFG model and dynamic strategy.** Figures show averaged CSP_{freq} values for MP predictions as functions of sample size (left) and corresponding variances (right), derived on the basis of our tRNA benchmark set by considering the traditional SCFG model and employing the dynamic sampling strategy. Plots correspond to those presented in Figures 9.8a and 9.8b.



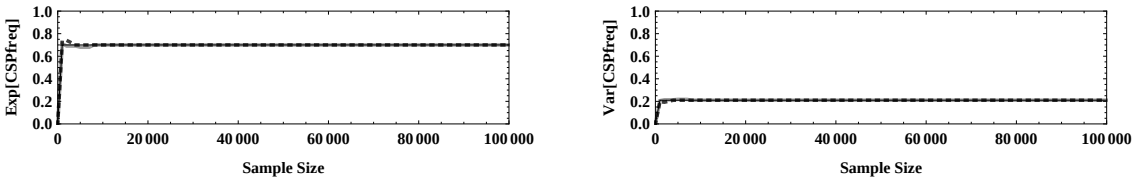
(a) Level 0.



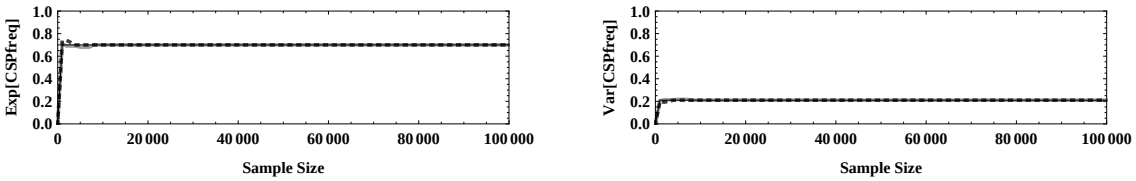
(b) Level 1.



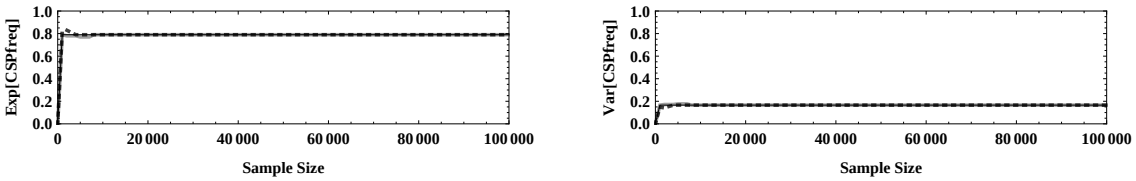
(c) Level 2.



(d) Level 3.

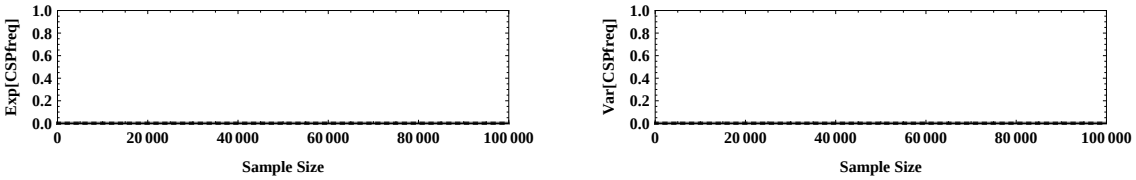


(e) Level 4.

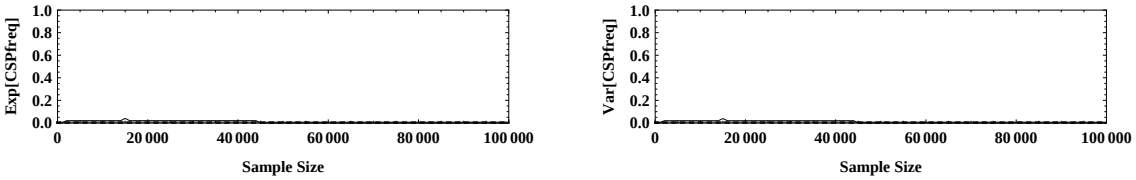


(f) Level 5.

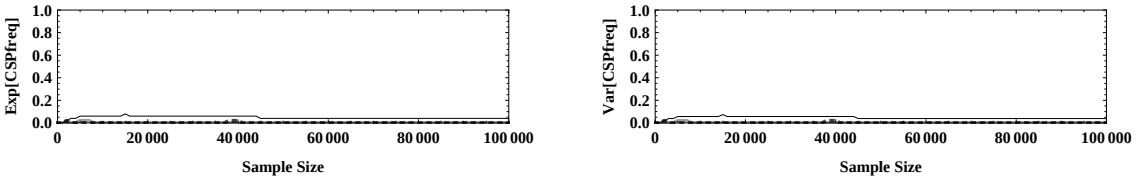
Figure E.12.: **Shape results for tRNA benchmark, with LSCFG model and dynamic strategy.** Figures show averaged CSP_{freq} values for MP predictions as functions of sample size (left) and corresponding variances (right), derived on the basis of our tRNA benchmark set by considering the LSCFG model and employing the dynamic sampling strategy. Plots correspond to those presented in Figures 9.8c and 9.8d.



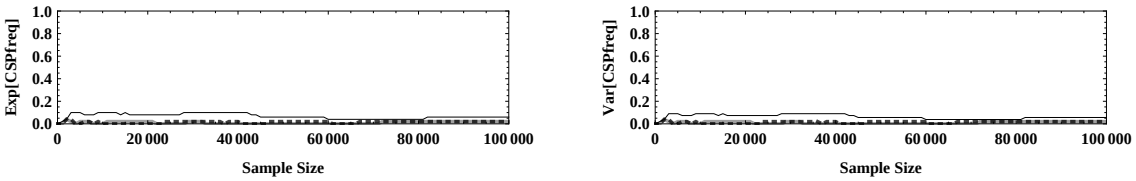
(a) Level 0.



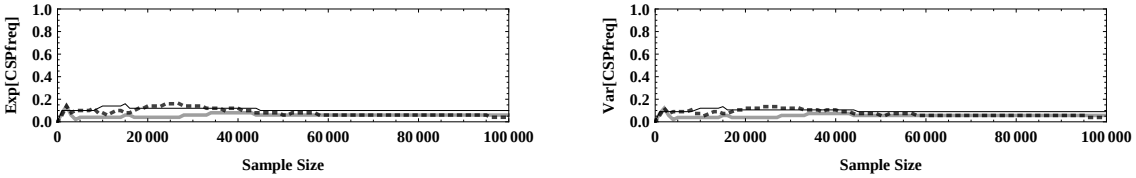
(b) Level 1.



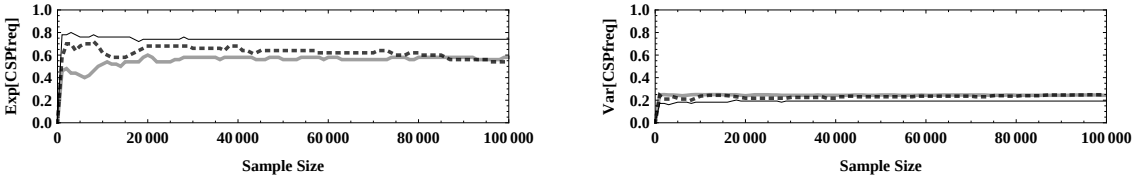
(c) Level 2.



(d) Level 3.

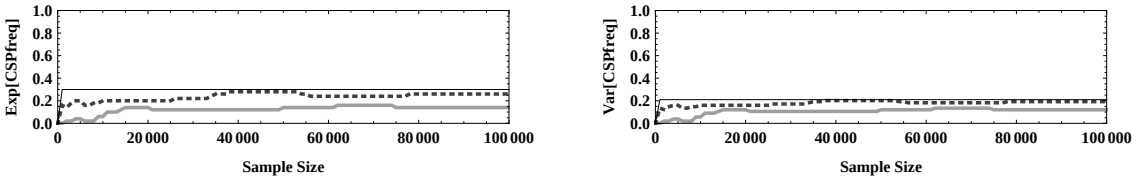


(e) Level 4.

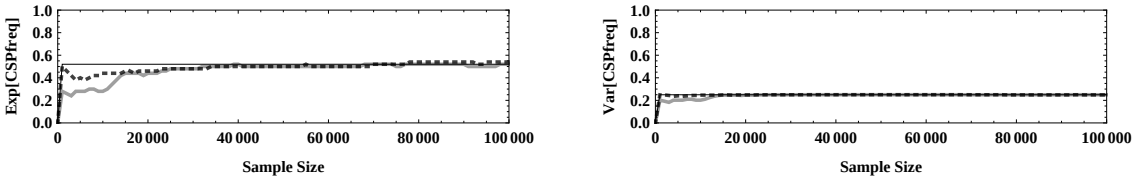


(f) Level 5.

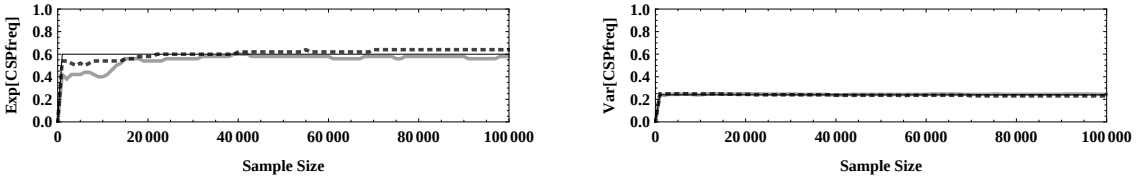
Figure E.13.: **Shape results for 5S rRNA benchmark, with SCFG model and common strategy.** Figures show averaged CSP_{freq} values for MP predictions as functions of sample size (left) and corresponding variances (right), derived on the basis of our 5S rRNA benchmark set by considering the traditional SCFG model and employing the common sampling strategy. Plots correspond to those presented in Figures 9.9a and 9.9b.



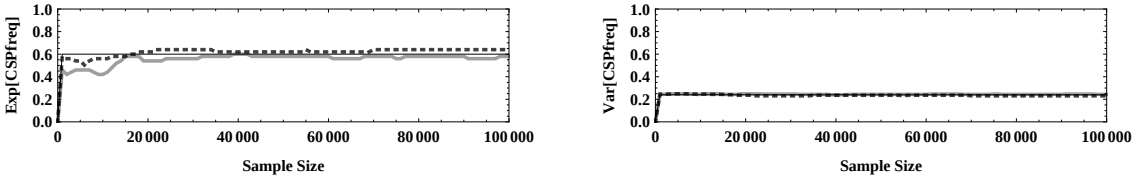
(a) Level 0.



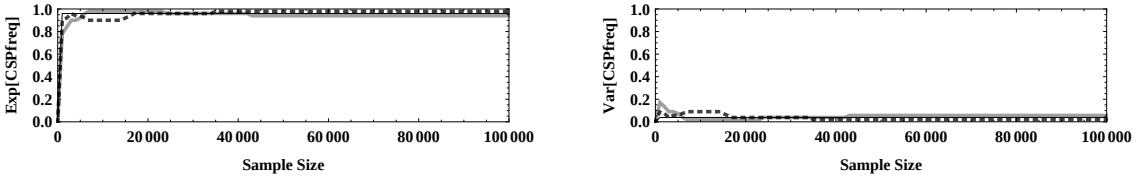
(b) Level 1.



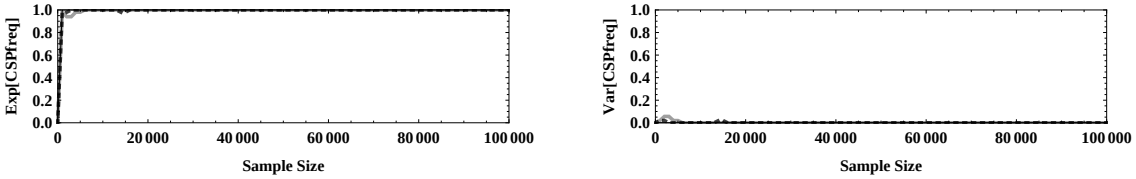
(c) Level 2.



(d) Level 3.

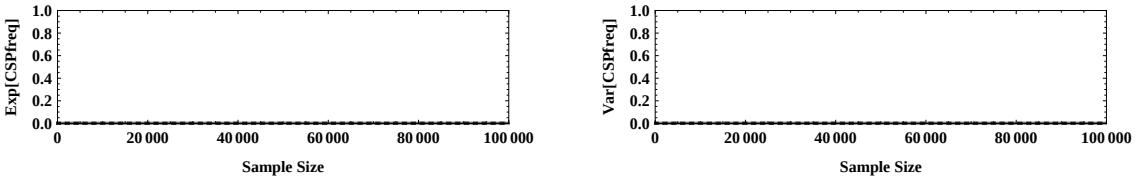


(e) Level 4.

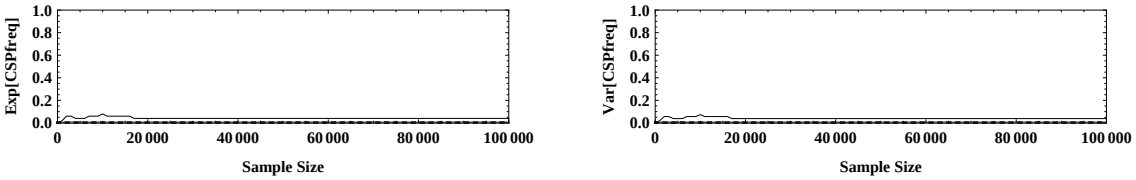


(f) Level 5.

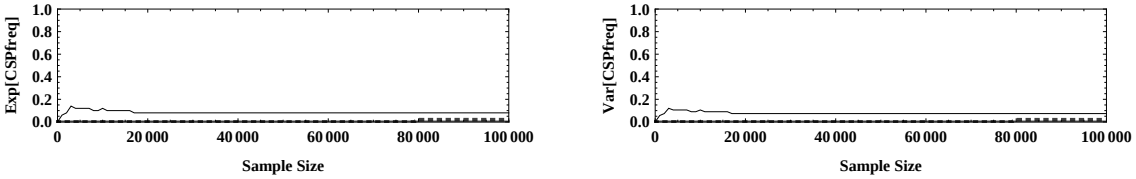
Figure E.14.: **Shape results for 5S rRNA benchmark, with LSCFG model and common strategy.** Figures show averaged CSP_{freq} values for MP predictions as functions of sample size (left) and corresponding variances (right), derived on the basis of our 5S rRNA benchmark set by considering the LSCFG model and employing the common sampling strategy. Plots correspond to those presented in Figures 9.9c and 9.9d.



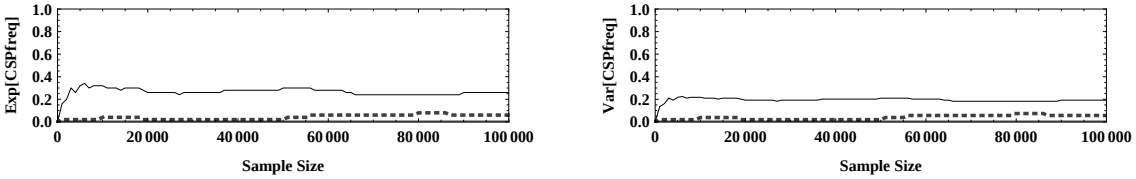
(a) Level 0.



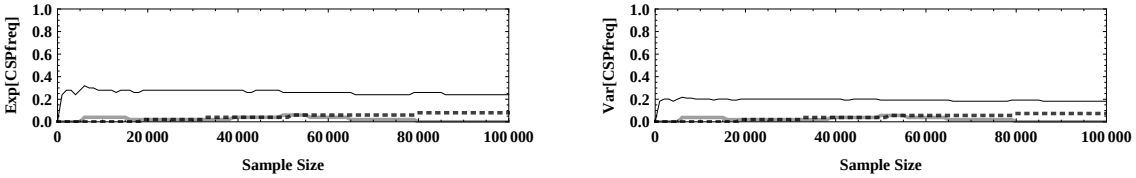
(b) Level 1.



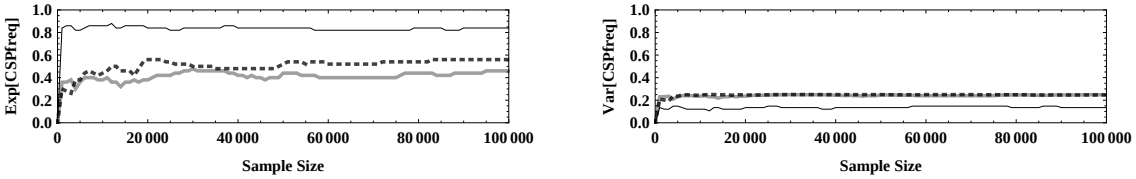
(c) Level 2.



(d) Level 3.

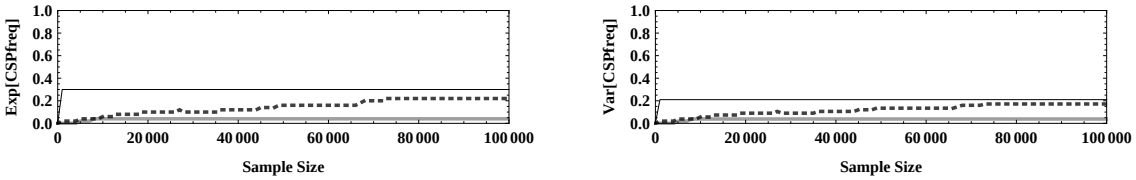


(e) Level 4.

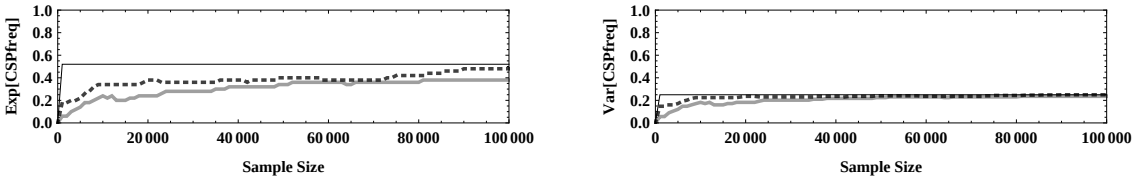


(f) Level 5.

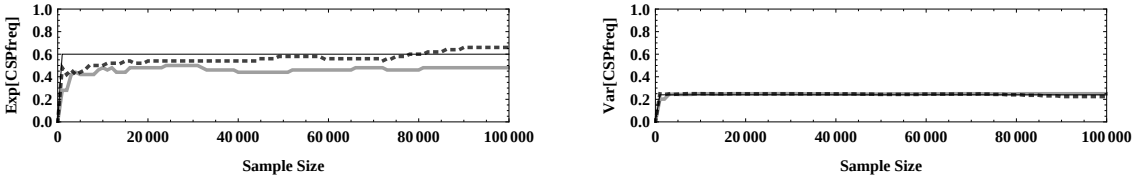
Figure E.15.: **Shape results for 5S rRNA benchmark, with SCFG model and dynamic strategy.** Figures show averaged CSP_{freq} values for MP predictions as functions of sample size (left) and corresponding variances (right), derived on the basis of our 5S rRNA benchmark set by considering the traditional SCFG model and employing the dynamic sampling strategy. Plots correspond to those presented in Figures 9.10a and 9.10b.



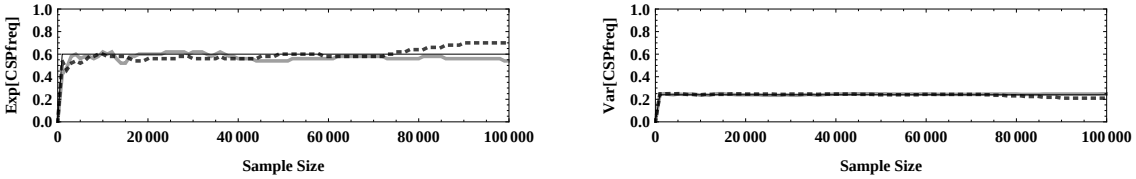
(a) Level 0.



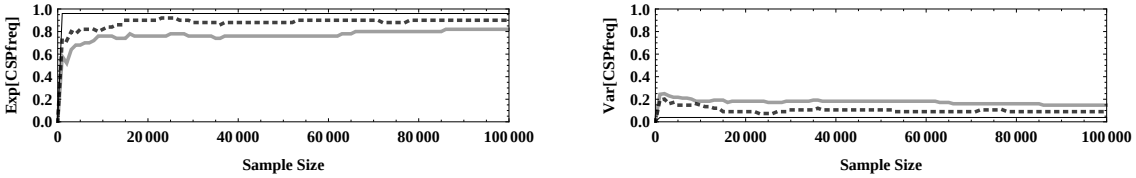
(b) Level 1.



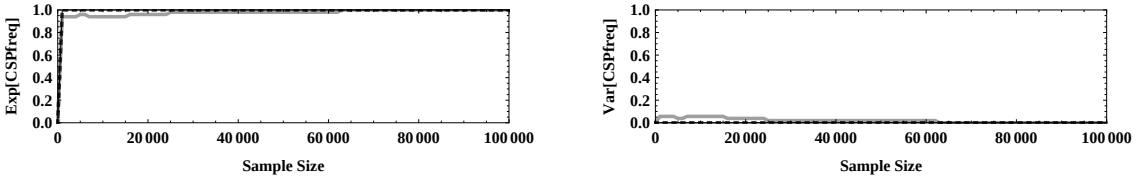
(c) Level 2.



(d) Level 3.

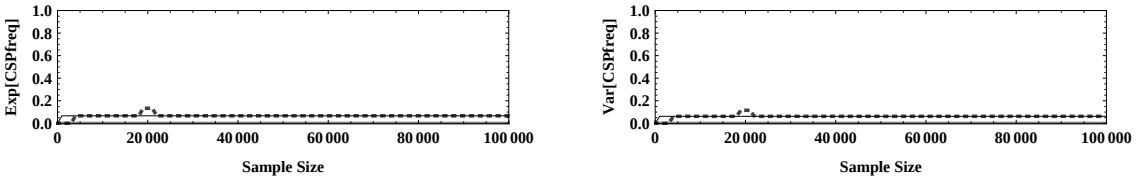


(e) Level 4.

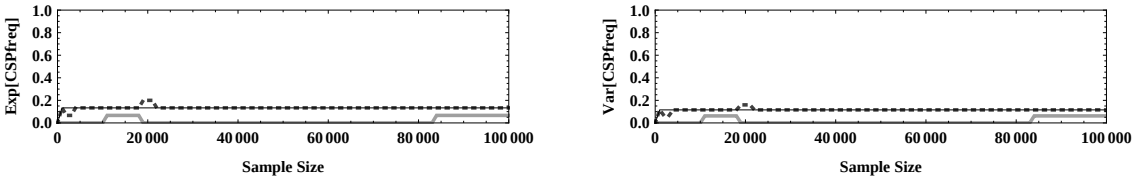


(f) Level 5.

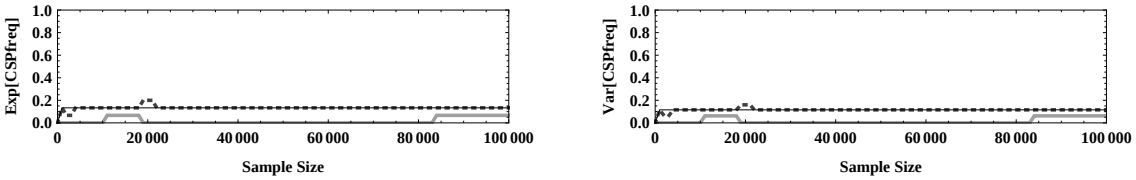
Figure E.16.: **Shape results for 5S rRNA benchmark, with LSCFG model and dynamic strategy.** Figures show averaged CSP_{freq} values for MP predictions as functions of sample size (left) and corresponding variances (right), derived on the basis of our 5S rRNA benchmark set by considering the LSCFG model and employing the dynamic sampling strategy. Plots correspond to those presented in Figures 9.10c and 9.10d.



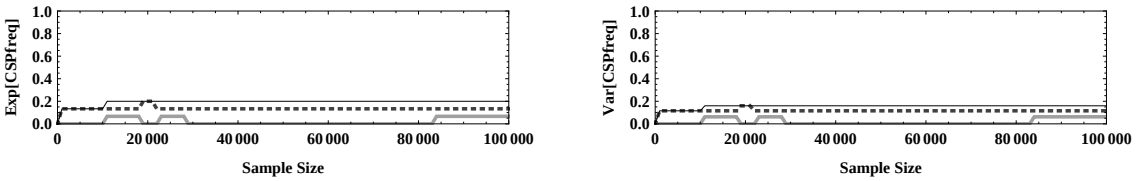
(a) Level 0.



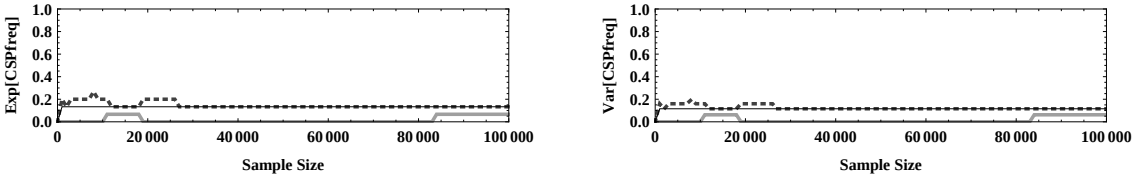
(b) Level 1.



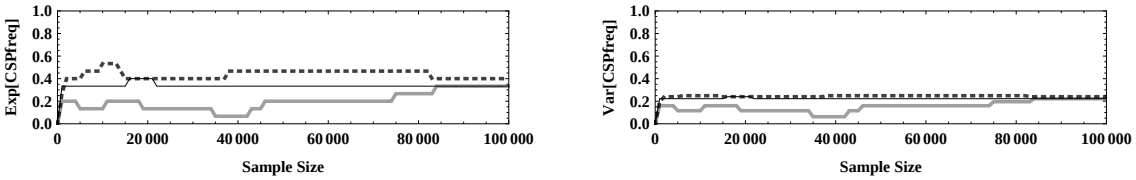
(c) Level 2.



(d) Level 3.

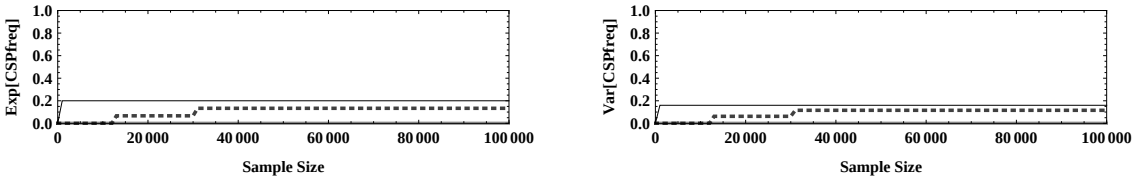


(e) Level 4.

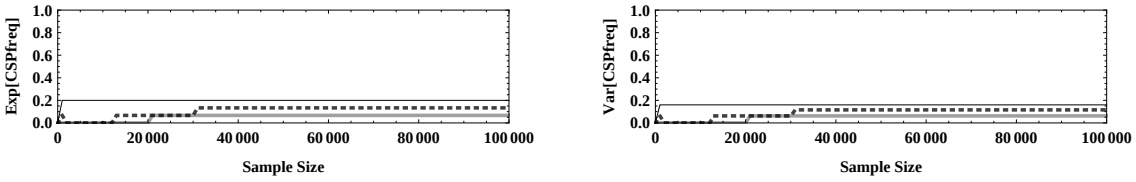


(f) Level 5.

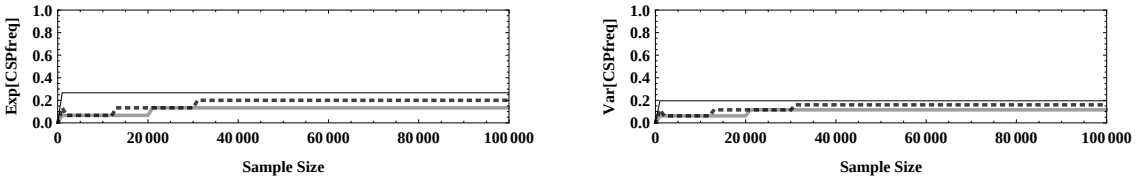
Figure E.17.: **Shape results for S-151Rfam benchmark, with SCFG model and common strategy.** Figures show averaged CSP_{freq} values for MP predictions as functions of sample size (left) and corresponding variances (right), derived on the basis of our S-151Rfam benchmark set by considering the traditional SCFG model and employing the common sampling strategy. Plots correspond to those presented in Figures 9.11a and 9.11b.



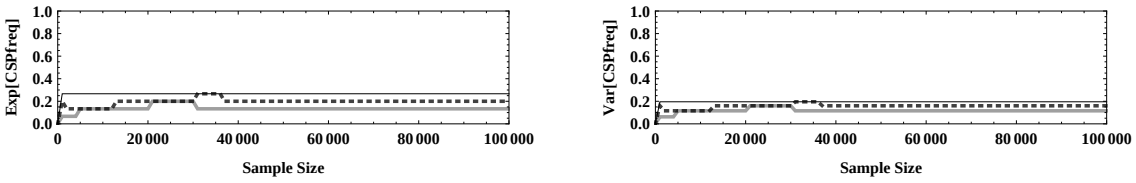
(a) Level 0.



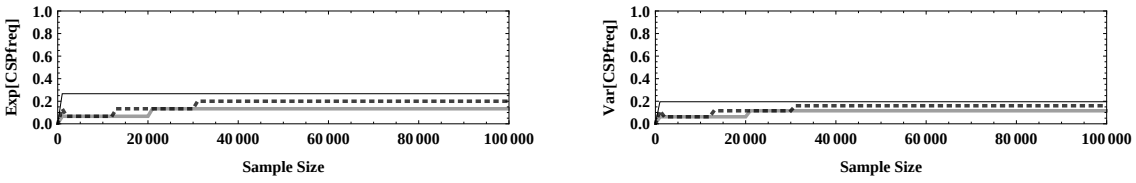
(b) Level 1.



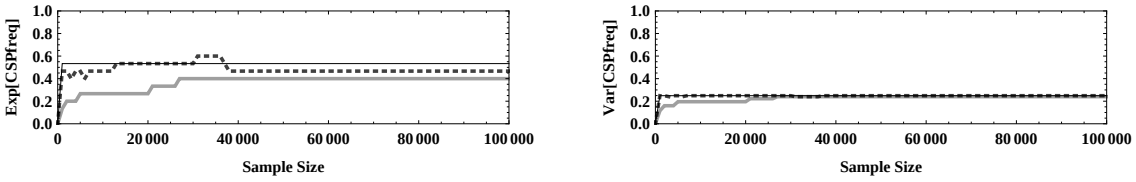
(c) Level 2.



(d) Level 3.

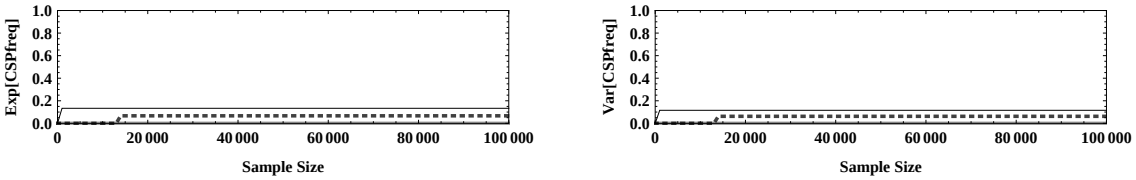


(e) Level 4.

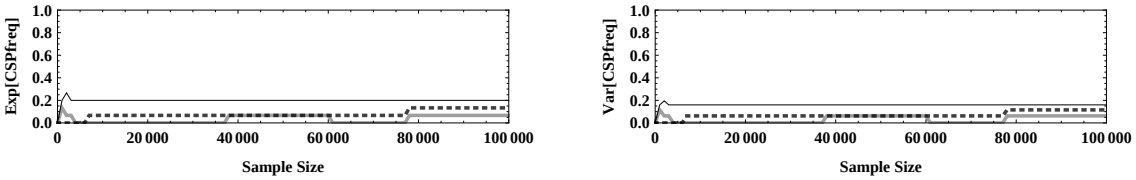


(f) Level 5.

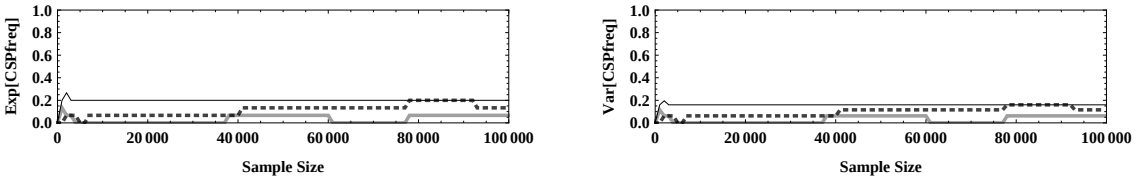
Figure E.18.: **Shape results for S-151Rfam benchmark, with LSCFG model and common strategy.** Figures show averaged CSP_{freq} values for MP predictions as functions of sample size (left) and corresponding variances (right), derived on the basis of our S-151Rfam benchmark set by considering the LSCFG model and employing the common sampling strategy. Plots correspond to those presented in Figures 9.11c and 9.11d.



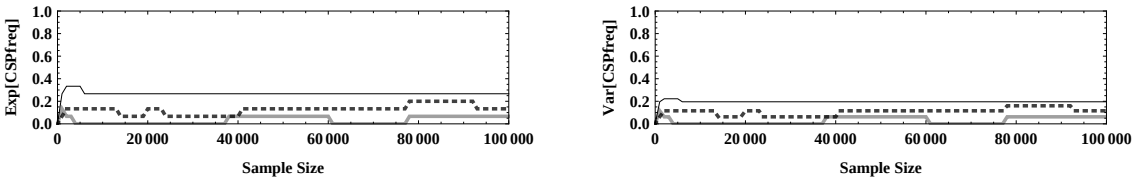
(a) Level 0.



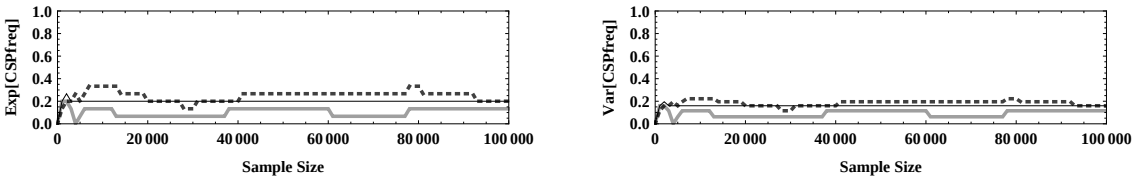
(b) Level 1.



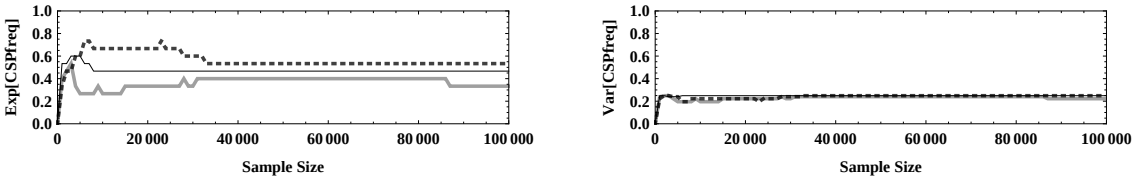
(c) Level 2.



(d) Level 3.

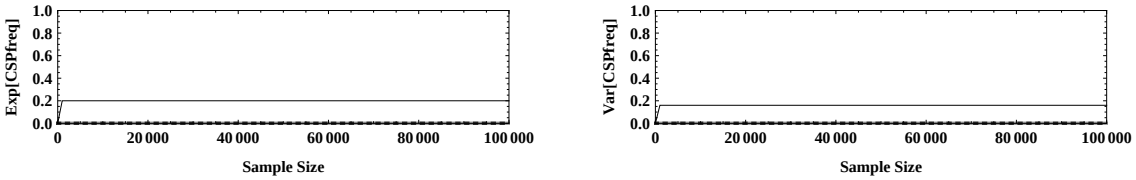


(e) Level 4.

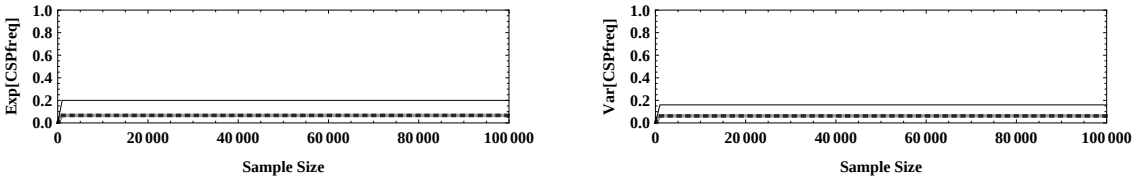


(f) Level 5.

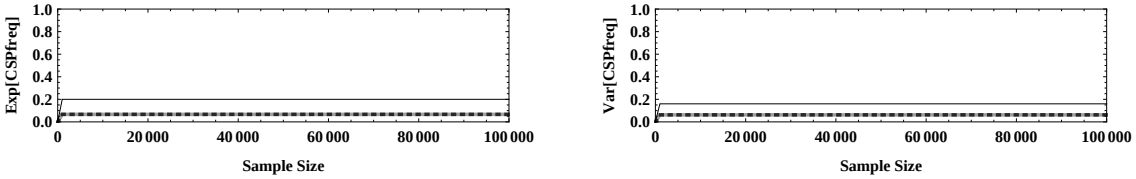
Figure E.19.: **Shape results for S-151Rfam benchmark, with SCFG model and dynamic strategy.** Figures show averaged CSP_{freq} values for MP predictions as functions of sample size (left) and corresponding variances (right), derived on the basis of our S-151Rfam benchmark set by considering the traditional SCFG model and employing the dynamic sampling strategy. Plots correspond to those presented in Figures 9.12a and 9.12b.



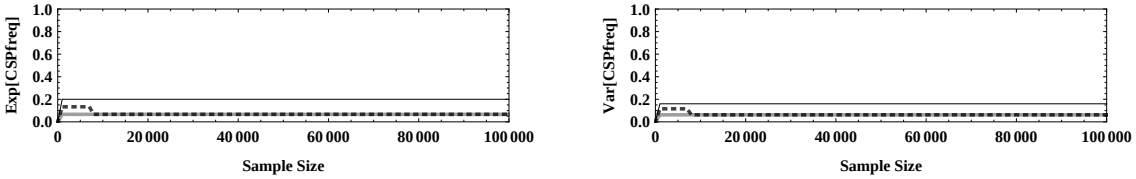
(a) Level 0.



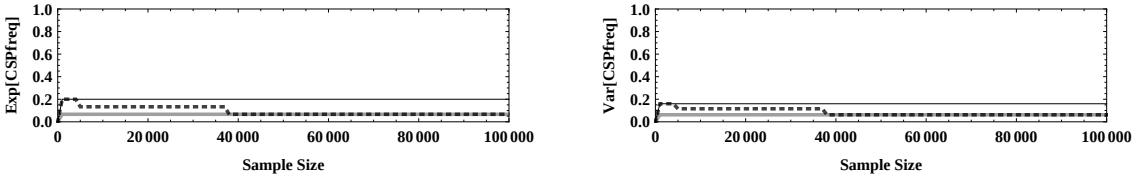
(b) Level 1.



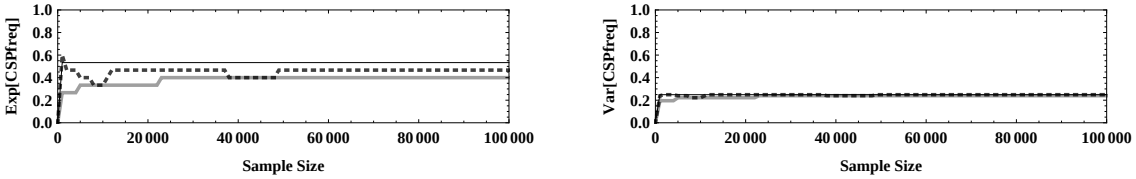
(c) Level 2.



(d) Level 3.



(e) Level 4.



(f) Level 5.

Figure E.20.: **Shape results for S-151Rfam benchmark, with LSCFG model and dynamic strategy.** Figures show averaged CSP_{freq} values for MP predictions as functions of sample size (left) and corresponding variances (right), derived on the basis of our S-151Rfam benchmark set by considering the LSCFG model and employing the dynamic sampling strategy. Plots correspond to those presented in Figures 9.12c and 9.12d.

Bibliography

[ABHC08] M. Andronescu, V. Bereg, H. H. Hoos, and A. Condon. RNA STRAND: the RNA secondary structure and statistical analysis database. *BMC Bioinformatics*, 9:340, 2008.

[ACH+07] M. Andronescu, A. Condon, H. H. Hoos, D. H. Mathews, and K. P. Murphy. Efficient parameter estimation for RNA secondary structure prediction. *Bioinformatics*, 23:i19, 2007.

[ACH+10] M. Andronescu, A. Condon, H. H. Hoos, D. H. Mathews, and K. P. Murphy. Computational approaches for RNA energy parameter estimation. *RNA*, 16:2304–2318, 2010.

[AdCC+08] J. Allali, Y. d´Aubenton Carafa, C. Chauve, A. Denise, C. Drevet, P. Ferraro, D. Gautheret, C. Herrbach, F. Leclerc, A. de Monte, A. Ouangraoua, M.-F. Sagot, C. Saule, M. Termier, C. Thermes, and H. Touzet. Benchmarking RNA secondary structures comparison algorithms. In *Actes des Journées Ouvertes de Biologie, Informatique et Mathématiques – JOBIM'08*, pages 67–68, 2008.

[ADKF70] V. L. Arlazarov, E. A. Dinic, M. A. Kronrod, and I. A. Faradzev. On economical construction of the transitive closure of a directed graph. *Soviet Mathematics—Doklady*, 11(5):1209–1210, 1970.

[AE85] S. F. Altschul and B. W. Erickson. Significance of nucleotide sequence alignments: a method for random sequence permutation that preserves dinucleotide and codon usage. *Molecular Biology and Evolution*, 2(6):256–538, 1985.

[Aho68] A. Aho. Indexed grammars – an extension of context-free grammars. *Journal of the ACM*, 15(4):647–671, 1968.

[AJL+02] B. Alberts, A. Johnson, J. Lewis, M. Raff, K. Roberts, and P. Walter. *Molecular Biology of the Cell*. Garland Publishing, fourth edition, 2002.

[Aku99] T. Akutsu. Approximation and exact algorithms for RNA secondary structure prediction and recognition of stochastic context-free languages. *Journal of Combinatorial Optimization*, 3(2–3):321–336, 1999.

[Aku00] T. Akutsu. Dynamic programming algorithms for RNA secondary structure prediction with pseudoknots. *Discrete Applied Mathematics*, 104(1-3):45–62, 2000.

[Ald97] J. Aldrich. R. A. Fisher and the making of maximum likelihood 1912–1922. *Statistical Science*, 12(3):162–176, 1997.

[And08] M. Andronescu. *Computational approaches for RNA energy parameter estimation*. PhD thesis, University of British Columbia, Vancouver, Canada, 2008.

[ARL+06] E. S. Andersen, M. A. Rosenblad, N. Larsen, J. C. Westergaard, J. Burks, I. K. Wower, J. Wower, J. Gorodkin, T. Samuelsson, and C. Zwieb. The tmRDB and SRPDB resources. *Nucleic Acids Research*, 34:163–168, 2006.

[AvdBvBP90] J. P. Abrahams, M. van den Berg, E. van Batenburg, and C. W. Pleij. Prediction of RNA secondary structure, including pseudoknotting, by computer simulation. *Nucleic Acids Research*, 18(10):3035–3044, 1990.

[BBC+00] P. Baldi, S. Brunak, Y. Chauvin, C. A. Andersen, and H. Nielsen. Assessing the accuracy of prediction algorithms for classification: an overview. *Bioinformatics*, 16(5):412–424, 2000.

[BCW+04] J. Barrick, K. Corbino, W. Winkler, A. Nahvi, M. Mandal, J. Collins, M. Lee, A. Roth, N. Sudarsan, I. Jona, J. Wickiser, and R. Breaker. New RNA motifs suggest an expanded scope for riboswitches in bacterial genetic control. *Proceedings of the National Academy of Sciences of the United States of America*, 101(17):6421–6426, 2004.

[BDTU74] P. N. Borer, B. Dengler, I. Tinoco, Jr., and O. C. Uhlenbeck. Stability of ribonucleic acid double-stranded helices. *Journal of Molecular Biology*, 86:843–853, 1974.

[BHS06] S. Bernhart, I. Hofacker, and P. Stadler. Local RNA base pairing probabilities in large sequences. *Bioinformatics*, 22:614–615, 2006.

[Bis07] C. M. Bishop. *Pattern Recognition and Machine Learning*. Springer, 2007.

[BKML+11] D. A. Benson, I. Karsch-Mizrachi, D. J. Lipman, J. Ostell, and E. W. Sayers. Genbank. *Nucleic Acids Research*, 39:D32–D37, 2011.

[BOB+92] H. M. Berman, W. K. Olson, D. L. Beveridge, J. Westbrook, A. Gelbin, T. Demeny, S. H. Hsieh, A. R. Srinivasan, and B. Schneider. The nucleic acid database. A comprehensive relational database of three-dimensional structures of nucleic acids. *Biophysical Journal*, 63(3):751–759, 1992.

[BR04] M. A. Beaumont and B. Rannala. The Bayesian revolution in genetics. *Nature Reviews Genetics*, 5:251–261, 2004.

[Bro99] J. W. Brown. The ribonuclease P database. *Nucleic Acids Research*, 27(1):314, 1999.

[BS07] J. Benedi and J. Sanchez. Fast stochastic context-free parsing: A stochastic version of the Valiant algorithm. *Lecture Notes in Computer Science*, 4477:80–88, 2007.

[BT92] G. E. P. Box and G. C. Tiao. *Bayesian Inference in Statistical Analysis*. Wiley-Interscience, 1992.

[BTM+06] S. H. Bernhart, H. Tafer, U. Mückstein, C. Flamm, P. F. Stadler, and I. L. Hofacker. Partition function and base pairing probabilities of RNA heterodimers. *Algorithms for Molecular Biology*, 1(3), 2006.

[BTZZU11] R. Backofen, D. Tsur, S. Zakov, and M. Ziv-Ukelson. Sparse RNA folding: Time and space efficient algorithms. *Journal of Discrete Algorithms*, 9:12–31, 2011.

[BW97] P. Brion and E. Westhof. Hierarchy and dynamics of RNA folding. *Annual Review of Biophysics and Biomolecular Structure*, 26:113–137, 1997.

[CB00] P. Clote and R. Backofen. *Computational Molecular Biology: An Introduction*. Mathematical and Computational Biology. Jon Wiley & Sons, Chichester, August 2000.

[CDK+06] K. Crammer, O. Dekel, J. Keshet, S. Shalev-Shwartz, and Y. Singer. Online passive-aggressive algorithms. *Journal of Machine Learning Research*, 7:551–585, 2006.

[CDR+04] A. Condon, B. Davy, B. Rastegari, S. Zhao, and F. Tarrant. Classifying RNA pseudoknotted structures. *Theoretical Computer Science*, 320:35–50, 2004.

[CG98] T. Chi and S. Geman. Estimation of probabilistic context-free grammars. *Computational Linguistics*, 24(2):299–305, 1998.

[Cho56] N. Chomsky. Three models for the description of language. *Information Theory, IEEE Transactions*, 2(3):113–124, September 1956.

[CL08] L. E. Carvalho and C. E. Lawrence. Centroid estimation in discrete high-dimensional spaces with applications in biology. *Proceedings of the National Academy of Sciences of the United States of America*, 105:3209–3214, 2008.

[CLD05] C. Y. Chan, C. E. Lawrence, and Y. Ding. Structure clustering features on the Sfold web server. *Bioinformatics*, 21:3926–3928, 2005.

[CNHP98] J.-L. Chen, J. M. Nolan, M. E. Harris, and N. R. Pace. Comparative photocrosslinking analysis of the tertiary structures of Escherichia coli and Bacillus subtilis RNase P RNAs. *The EMBO Journal*, 17:1515–1525, 1998.

[Col02] M. Collins. Discriminative training methods for hidden Markov models: Theory and experiments with perceptron algorithms. In *Proceedings of the ACL-02 conference on Empirical methods in natural language processing*, pages 1–8, Stroudsburg, PA, USA, 2002. Association for Computational Linguistics.

[CS70] J. Cocke and J. T. Schwartz. Programming languages and their compilers: Preliminary notes. Technical report, Courant Institute of Mathematical Sciences, New York, 1970.

[CS06] A. Corazza and G. Satta. Cross-entropy and estimation of probabilistic context-free grammars. In *Proceedings of HLT-NAACL'06*, pages 335–342, New York, USA, 2006.

[CSS+02] J. J. Cannone, S. Subramanian, M. N. Schnare, J. R. Collett, L. M. D´Souza, Y. Du, B. Feng, N. Lin, L. V. Madabusi, K. M. Müller, N. Pande, Z. Shang, N. Yu, and R. R. Gutell. The comparative RNA web (CRW) site: an online database of comparative sequence and structure information for ribosomal, intron, and other RNAs. *BMC Bioinformatics*, 3(1):15, 2002.

[CZJT04] G. Chen, B. M. Znosko, X. Jiao, and D. H. Turner. Factors affecting thermodynamic stabilities of RNA 3x3 internal loops. *Biochemistry*, 43:12865–12876, 2004.

[DB12] S. Dimitrieva and P. Bucher. Practicality and time complexity of a sparsified RNA folding algorithm. *Journal of Bioinformatics and Computational Biology*, 10(2), 2012.

[DCCG04] K. J. Doshi, J. J. Cannone, C. W. Cobaugh, and R. R. Gutell. Evaluation of the suitability of free-energy minimization using nearest-neighbor energy parameters for RNA secondary structure prediction. *BMC Bioinformatics*, 5:105, 2004.

[DCL04] Y. Ding, C. Y. Chan, and C. E. Lawrence. Sfold web server for statistical folding and rational design of nucleic acids. *Nucleic Acids Research*, 32:W135–W141, 2004.

[DCL05] Y. Ding, C. Y. Chan, and C. E. Lawrence. RNA secondary structure prediction by centroids in a Boltzmann weighted ensemble. *RNA*, 11:1157–1166, 2005.

[DE04] R. D. Dowell and S. R. Eddy. Evaluation of several lightweight stochastic context-free grammars for RNA secondary structure prediction. *BMC Bioinformatics*, 5:71, 2004.

[DE06] R. D. Dowell and S. R. Eddy. Efficient pairwise RNA structure prediction and alignment using sequence alignment constraints. *BMC Bioinformatics*, 7(1):400, 2006.

[DEKM98] R. Durbin, S. Eddy, A. Krogh, and G. Mitchison. *Biological Sequence Analysis: Probabilistic Models of Proteins and Nucleic Acids*. Cambridge University Press, Cambridge, UK, 1998.

[DFLS04] P. Duchon, P. Flajolet, G. Louchard, and G. Schaeffer. Boltzmann samplers for the random generation of combinatorial structures. *Combinatorics, Probability, and Computing*, 13:577–625, 2004.

[DFN07] C. B. Do, C.-S. Foo, and A. Y. Ng. Efficient multiple hyperparameter learning for log-linear models. In *Neural Information Processing Systems*, volume 21, 2007.

[DG94] S. H. Damberger and R. R. Gutell. A comparative database of group I intron structures. *Nucleic Acids Research*, 22:3508–3510, 1994.

[Din02] Y. Ding. Rational statistical design of antisense oligonucleotides for high throughput functional genomics and drug target validation. *Statistica Sinica*, 12:273–296, 2002.

[Din06] Y. Ding. Statistical and bayesian approaches to RNA secondary structure prediction. *RNA*, 12:323–331, 2006.

[DL99] Y. Ding and C. E. Lawrence. A Bayesian statistical algorithm for RNA secondary structure prediction. *Computers & Chemistry*, 23:387–400, 1999.

[DL03] Y. Ding and C. E. Lawrence. A statistical sampling algorithm for RNA secondary structure prediction. *Nucleic Acids Research*, 31(24):7280–7301, 2003.

[DP03] R. M. Dirks and N. A. Pierce. A partition function algorithm for nucleic acid secondary structure including pseudoknots. *Journal of Computational Chemistry*, 24:1664–1677, 2003.

[DP04] R. M. Dirks and N. A. Pierce. An algorithm for computing nucleic acid base-pairing probabilities including pseudoknots. *Journal of Computational Chemistry*, 25:1295–1304, 2004.

[DPD92] E. Dam, K. Pleij, and D. Draper. Structural and functional aspects of RNA pseudoknots. *Biochemistry*, 31:11665–11676, 1992.

[DPT03] A. Denise, Y. Ponty, and M. Termier. Random generation of structured genomic sequences. In *Proceedings of RECOMB'03*, page 3 pages (poster), 2003.

[DWB06] C. B. Do, D. A. Woods, and S. Batzoglou. CONTRAfold: RNA secondary structure prediction without physics-based models. *Bioinformatics*, 22(14):e90–e98, 2006.

[Ear70] J. Earley. An efficient context-free parsing algorithm. *Communications of the ACM*, 13(2):94–102, 1970.

[ED94] S. R. Eddy and R. Durbin. RNA sequence analysis using covariance models. *Nucleic Acids Research*, 2(11):2079–2088, 1994.

[Edd01] S. R. Eddy. Non-coding RNA genes and the modern RNA world. *Nature Reviews Genetics*, 2(12):919–929, 2001.

[Edd04] S. R. Eddy. How do RNA folding algorithms work? *Nature Biotechnology*, 22(11):1457–1458, 2004.

[EGGI92a] D. Eppstein, Z. Galil, R. Giancarlo, and G. F. Italiano. Sparse dynamic-programming. 1. linear cost-functions. *Journal of the Association for Computing Machinery*, 39(3):519–545, 1992.

[EGGI92b] D. Eppstein, Z. Galil, R. Giancarlo, and G. F. Italiano. Sparse dynamic-programming. 2. convex and concave cost-functions. *Journal of the Association for Computing Machinery*, 39(3):546–567, 1992.

[Fel07] B. Felden. RNA structure: experimental analysis. *Current Opinion in Microbiology*, 10:286–291, 2007.

[FFP07] P. Flajolet, E. Fusy, and C. Pivoteau. Boltzmann sampling of unlabelled structures. In *Proceedings of ANALCO'07 (Analytic Combinatorics and Algorithms) Conference*, pages 201–211. SIAM Press, January 2007.

[FG10a] Y. Frid and D. Gusfield. A simple, practical and complete $\mathcal{O}(n^3/\log(n))$-time algorithm for RNA folding using the Four-Russians speedup. *Algorithms for Molecular Biology*, 5(1):5–13, 2010.

[FG10b] Y. Frid and D. Gusfield. A worst-case and practical speedup for the RNA co-folding problem using the Four-Russians idea. In *Proceedings of the 10th international workshop on Algorithms in Bioinformatics*, WABI'10, pages 1–12, 2010.

[FH72] K. S. Fu and T. Huang. Stochastic grammars and languages. *International Journal of Computer and Information Sciences*, 1(2):135–170, 1972.

[Fit83] W. M. Fitch. Random sequences. *Journal of Molecular Biology*, 163:171–176, 1983.

[FS09] P. Flajolet and R. Sedgewick. *Analytic Combinatorics*. Cambridge University Press, January 2009.

[FZC94] P. Flajolet, P. Zimmermann, and B. Van Cutsem. A calculus for the random generation of combinatorial structures. *Theoretical Computer Science*, 132(2):1–35, 1994.

[Gaz88] G. Gazdar. Applicability of indexed grammars to natural languages. In U. Reyle and C. Rohrer, editors, *Natural Language Parsing and Linguistic Theories*, page 69Ð94. Reidel, Dordrecht, 1988.

[GC73] J. Gralla and D. M. Crothers. Free energy of imperfect nucleic acid helices : II. small hairpin loops. *Journal of Molecular Biology*, 73:497–511, 1973.

[GCSR97] A. Gelman, J. B. Carlin, H. S. Stern, and D. B. Rubin. *Bayesian data analysis*. Chapman & Hall/CRC, New York, 1997.

[GG04] P. P. Gardner and R. Giegerich. A comprehensive comparison of comparative RNA structure prediction approaches. *BMC Bioinformatics*, 5(140), 2004.

[GHS97] J. Gorodkin, L. Heyer, and G. Stormo. Finding the most significant common sequence and structure motifs in a set of RNA sequences. *Nucleic Acids Research*, 25(18):3724–3732, 1997.

[Gie00] R. Giegerich. Explaining and controlling ambiguity in dynamic programming. In *Proceedings of the 11th Annual Symposium on Combinatorial Pattern Matching*, pages 46–59, London, UK, 2000. Springer-Verlag.

[GJ08] D. Grune and C. J. H. Jacobs. *Parsing Techniques – A Practical Guide*. Springer, 2nd edition, 2008.

[GJBM+03] S. Griffiths-Jones, A. Bateman, M. Marshall, A. Khanna, and S. R. Eddy. Rfam: an RNA family database. *Nucleic Acids Research*, 31(1):439–441, 2003.

[GJMM+05] S. Griffiths-Jones, S. Moxon, M. Marshall, A. Khanna, S. R. Eddy, and A. Bateman. Rfam: annotating non-coding RNAs in complete genomes. *Nucleic Acids Research*, 33:D121–D124, 2005.

[GLC02] R. R. Gutell, J. C. Lee, and J. J. Cannone. The accuracy of ribosomal RNA comparative structure models. *Current Opinion in Structural Biology*, 12:301–310, 2002.

[Goo98] J. T. Goodman. *Parsing Inside-Out*. PhD thesis, Harvard University, Cambridge, Massachusetts, May 1998.

[Goo99] J. Goodman. Semiring parsing. *Computational Linguistics*, 25(4):573–605, 1999.

[GPH+92] R. R. Gutell, A. Power, G. Z. Hertz, E. J. Putz, and G. D. Stormo. Identifying constraints on the higher-order structure of RNA: Continued development and application of comparative sequence analysis methods. *Nucleic Acids Research*, 20:5785–5795, 1992.

[GvH06] T. Gesell and A. von Haeseler. In silico sequence evolution with site-specific interactions along phylogenetic trees. *Bioinformatics*, 22:716–722, 2006.

[GVR04] R. Giegerich, B. Voß, and M. Rehmsmeier. Abstract shapes of RNA. *Nucleic Acids Research*, 32(16):4843–4851, 2004.

[GW90] R. R. Gutell and C. R. Woese. Higher order structural elements in ribosomal RNAs: Pseudo-knots and the use of noncanonical pairs. *Proceedings of the National Academy of Sciences of the United States of America*, 87:663–667, 1990.

[GWR80] S. L. Graham, L. Walter, and M. H. Ruzzo. An improved context-free recognizer. *ACM Transactions on Programming Languages and Systems*, 2(3):415–462, 1980.

[GzS11] R. Giegerich and C. Höner zu Siederdissen. Semantics and ambiguity of stochastic RNA family models. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, 8:499–516, 2011.

[Har78] M. A. Harrison. *Introduction to Formal Language Theory*. Addison-Wesley, 1978.

[HBS04] I. L. Hofacker, S. Bernhart, and P. Stadler. Alignment of RNA base pairing probability matrices. *Bioinformatics*, 20:2222–2227, 2004.

[HF71] T. Huang and K. S. Fu. On stochastic context-free languages. *Information Sciences*, 3:201–224, 1971.

[HFS+94] I. L. Hofacker, W. Fontana, P. F. Stadler, S. Bonhoeffer, M. Tacker, and P. Schuster. Fast folding and comparison of RNA secondary structures (the Vienna RNA package). *Monatshefte für Chemie*, 125(2):167–188, 1994.

[HFS02] I. L. Hofacker, M. Fekete, and P. F. Stadler. Secondary structure prediction for aligned RNA sequences. *Journal of Molecular Biology*, 319:1059–1066, 2002.

[HHW+01] J. K. Harris, E. S. Haas, D. Williams, D. N. F., and James W. Brown. New insight into RNase P RNA structure from comparative analysis of the archaeal RNA. *RNA*, 7:220–232, 2001.

[HKS+09] M. Hamada, H. Kiryu, K. Sato, T. Mituyama, and K. Asai. Prediction of RNA secondary structure using generalized centroid estimators. *Bioinformatics*, 25(4):465–473, 2009.

[Hof03] I. L. Hofacker. The Vienna RNA secondary structure server. *Nucleic Acids Research*, 31(13):3429–3431, 2003.

[Hol05] I. Holmes. Accelerated probabilistic inference of RNA structure evolution. *BMC Bioinformatics*, 24(6):73, 2005.

[HPS04] I. Hofacker, B. Priwitzer, and P. Stadler. Prediction of locally stable RNA secondary structures for genome-wide surveys. *Bioinformatics*, 20:186–190, 2004.

[HR] F. W. D. Huang and C. M. Reidys. On the combinatorics of sparsification. Submitted.

[HSS98] I.ÊL. Hofacker, P. Schuster, and P.ÊF. Stadler. Combinatorics of RNA secondary structures. *Discrete Applied Mathematics*, 88(1–3):207–237, 1998.

[HU79] J. E. Hopcroft and J. D. Ullman. *Introduction to Automata Theory, Languages, and Computation*. Addison-Wesley, 1979.

[IMN05] I. M. Meyer I. Miklos and B. Nagy. Moments of the Boltzmann distribution for RNA secondary structures. *Bulletin of Mathematical Biology*, 67:1031–1047, 2005.

[INN+03] R. Ishitani, O. Nureki, N. Nameki, N. Okada, S. Nishimura, and S. Yokoyama. Alternative tertiary structure of tRNA for recognition of a post-transcriptional modification enzyme. *Cell*, 113:383–394, 2003.

[Jay03] E. T. Jaynes. *Probability Theory: The Logic of Science*. Cambridge University Press, 2003.

[JMH+09] F. Jühling, M. Mörl, R. K. Hartmann, M. Sprinzl, P. F. Stadler, and J. Pütz. tRNAdb 2009: compilation of tRNA sequences and tRNA genes. *Nucleic Acids Research*, 37:D159–D162, 2009.

[JR91] B. H. Juang and L. R. Rabiner. Hidden Markov models for speech recognition. *Technometrics*, 33:251–272, 1991.

[JRG08] S. Janssen, J. Reeder, and R. Giegerich. Shape based indexing for faster search of RNA family databases. *BMC Bioinformatics*, 9(131), 2008.

[JTZ89] J. A. Jaeger, D. H. Turner, and M. Zuker. Improved predictions of secondary structures for RNA. *Proceedings of the National Academy of Sciences of the United States of America*, 86:7706–7710, 1989.

[JTZ90] J. A. Jaeger, D. H. Turner, and M. Zuker. Predicting optimal and suboptimal secondary structure for RNA. In R. F. Doolittle, editor, *Molecular Evolution: Computer Analysis of Protein and Nucleic Acid Sequences*, volume 183 of *Methods in Enzymology*, pages 281–306. Academic Press, San Diego, 1990.

[Kas65] T. Kasami. An efficient recognition and syntax analysis algorithm for context-free languages. Technical Report AFCRL-65-758, Air Force Cambridge Research Laboratory, 1965.

[KAS09] Y. Kato, T. Akutsu, and H. Seki. A grammatical approach to RNA-RNA interaction prediction. *Pattern Recognition*, 42(4):531–538, 2009.

[Kat07] Y. Kato. *Formal grammars for describing RNA pseudoknotted structure and their application to structure analysis*. PhD thesis, Department of Information Processing, Graduate School of Information Science, Nara Institute of Science and Technology, 2007.

[KBT99] R. Kierzek, M. Burkard, and D. Turner. Thermodynamics of single mismatches in RNA duplexes. *Biochemistry*, 38:14214–14223, 1999.

[KH99] B. Knudsen and J. Hein. RNA secondary structure prediction using stochastic context-free grammars and evolutionary history. *Bioinformatics*, 15(6):446–454, 1999.

[KH03] B. Knudsen and J. Hein. Pfold: RNA secondary structure prediction using stochastic context-free grammars. *Nucleic Acids Research*, 31(13):3423–3428, 2003.

[KKA08] H. Kiryu, T. Kin, and K. Asai. Rfold: an exact algorithm for computing local base pairing probabilities. *Bioinformatics*, 24, 2008.

[KMP00] Y. Kafri, D. Mukamel, and L. Peliti. Why is the DNA denaturation transition first order? *Physical Review Letters*, 85:4988–4991, 2000.

[Kro94] A. Krogh. Hidden Markov models for labeled sequences. In *Proceedings of the 12th IAPR International Conference on Pattern Recognition*, pages 140–144. IEEE Computer Society Press, 1994.

[Kro98] A. Krogh. An introduction to hidden Markov models for biological sequences. In *Computational Biology: Pattern Analysis and Machine Learning Methods*, chapter 4, pages 45–63. Elsevier, 1998.

[KS05] A. Kabakcioglu and A. L. Stella. A scale-free network hidden in the collapsing polymer. *Physical Review E*, 72:055102, 2005.

[KSK06] Y. Kato, H. Seki, and T. Kasami. RNA pseudoknotted structure prediction using stochastic multiple context-free grammar. *IPSJ Transactions on Bioinformatics*, 47:12–21, 2006.

[LB05] D. M. Layton and R. Bundschuh. A statistical analysis of RNA folding algorithms through thermodynamic parameter perturbation. *Nucleic Acids Research*, 33(2):519–524, 2005.

[LCZ91] N. Larsen, C., and Zwieb. SRP-RNA sequence alignment and secondary structure. *Nucleic Acids Research*, 19:209–215, 1991.

[Les74] A. M. Lesk. A combinatorial study of the effects of admitting non-Watson-Crick base pairings and of base compositions on the helix-forming potential of polynucleotides of random sequences. *Journal of Theoretical Biology*, 44:7–17, 1974.

[LGM09] Z. J. Lu, J. W. Gloor, and D. H. Mathews. Improved RNA secondary structure prediction by maximizing expected pair accuracy. *RNA*, 15:1805–1813, 2009.

[LHP05] C. K. Liu, A. Hertzmann, and Z. Popovic. Learning physics-based motion style with nonlinear inverse optimization. In *Proceedings of ACM SIGGRAPH 2005*, volume 24, pages 1071–1081, 2005.

[LKT90] C. E. Longfellow, R. Kierzek, and D. H. Turner. Thermodynamic and spectroscopic study of bulge loops in oligoribonucleotides. *Biochemistry*, 29:278–285, 1990.

[LMM+12] S. J. Lange, D. Maticzka, M. Möhl, N. J. Gagnon, C. M. Brown, and R. Backofen. Global or local? predicting secondary structure and accessibility in mRNAs. *Nucleic Acids Research*, 2012.

[LMP01] J. Lafferty, A. McCallum, and F. Pereira. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *18th Proceedings of the International Conference in Machine Learning*, pages 282–289, 2001.

[LP00] R. B. Lyngso and C. N. S. Pedersen. RNA pseudoknot prediction in energy-based models. *Journal of Computational Biology*, 7(3/4):409–427, 2000.

[LW01] N. Leontis and E. Westhof. Geometric nomenclature and classification of RNA base pairs. *RNA*, 7:499–512, 2001.

[LW06] A. Lescoute and E. Westhof. Topology of three-way junctions in folded RNAs. *RNA*, 12(1):83–93, 2006.

[LY90] K. Lari and S. J. Young. The estimation of stochastic context-free grammars using the inside-outside algorithm. *Computer Speech and Language*, 4:35–56, 1990.

[LY91] K. Lari and S. J. Young. Applications of stochastic context-free grammars using the inside-outside algorithm. *Computer Speech and Language*, 5:237–257, 1991.

[Mac03] D. J. C. MacKay. *Information Theory, Inference, and Learning Algorithms*. Cambridge University Press, 2003.

[Mai07] R. S. Maier. Parametrized stochastic grammars for RNA secondary structure prediction. *Information Theory and Applications Workshop*, pages 256–260, 2007.

[Mat04] D. H. Mathews. Using an RNA secondary structure partition function to determine confidence in base pairs predicted by free energy minimization. *RNA*, 10:1178–1190, 2004.

[McC90] J. S. McCaskill. The equilibrium partition function and base pair binding probabilities for RNA secondary structure. *Biopolymers*, 29:1105–1119, 1990.

[MDC+04] D. H. Mathews, M. D. Disney, J. L. Childs, S. J. Schroeder, M. Zuker, and D. H. Turner. Incorporating chemical modification constraints into a dynamic programming algorithm for prediction of RNA secondary structure. *Proceedings of the National Academy of Science of the United States of America*, 101(19):7287–7292, 2004.

[MJW98] C. Massire, L. Jaeger, and E. Westhof. Derivation of the three-dimensional architecture of bacterial ribonuclease P RNAs from comparative sequence analysis. *Journal of Molecular Biology*, 279:773–793, 1998.

[MM01] C. Martinez and X. Molinero. A generic approach for the unranking of labeled combinatorial classes. *Random Structures & Algorithms*, 19(3-4):472–497, 2001.

[MM04] I. Meyer and I. Miklos. Co-transcriptional folding is encoded within RNA genes. *BMC Molecular Biology*, 5:10, 2004.

[MN08] D. Metzler and M. E. Nebel. Predicting RNA secondary structures with pseudoknots by MCMC sampling. *Journal of Mathematical Biology*, 56:161–181, 2008.

[Mol05] X. Molinero. *Ordered Generation of Classes of Combinatorial Structures*. PhD thesis, Universitat Politècnica de Catalunya, 2005.

[MR03] V. L. Murthy and G. D. Rose. RNABase: an annotated database of RNA structures. *Nucleic Acids Research*, 31:502–504, 2003.

[MRB04] M. Mandal, R. R., and Breaker. Gene regulation by riboswitches. *Nature Reviews Molecular Cell Biology*, 5:451–463, 2004.

[MSS05] H. Matsui, K. Sato, , and Y. Sakakibara. Pair stochastic tree adjoining grammars for aligning and predicting pseudoknot RNA structures. *Bioinformatics*, 21:2611–2617, 2005.

[MSW+10] M. Möhl, R. Salari, S. Will, R. Backofen, and S. C. Sahinalp. Sparsification of RNA structure prediction including pseudoknots. *Algorithms for Molecular Biology*, 5(39), 2010.

[MSZT99] D. H. Mathews, J. Sabina, M. Zuker, and D. H. Turner. Expanded sequence dependence of thermodynamic parameters improves prediction of RNA secondary structure. *Journal of Molecular Biology*, 288:911–940, 1999.

[MT02a] D. Mathews and D. Turner. Dynalign: An algorithm for finding the secondary structure common to two RNA sequences. *Journal of Molecular Biology*, 317(2):191–203, 2002.

[MT02b] D. H. Mathews and D. H. Turner. Experimentally derived nearest-neighbor parameters for the stability of RNA three- and four-way multibranch loops. *Biochemistry*, 41:869–880, 2002.

[MT06] D. H. Mathews and D. H. Turner. Prediction of RNA secondary structure by free energy minimization. *Current Opinion in Structural Biology*, 16:270–278, 2006.

[MW47] H. Mann and D. Whitney. On a test of whether one of two random variables is stochastically larger than the other. *Annals of Mathematical Statistics*, 18:50–60, 1947.

[MW90] F. Michel and E. Westhof. Modeling of the three-dimensional architecture of group I catalytic introns based on comparative sequence analysis. *Journal of Molecular Biology*, 216:585–610, 1990.

[MZ08] N. R. Markham and M. Zuker. UNAFold: Software for nucleic acid folding and hybridization. In J. Keith, editor, *Data, Sequence Analysis, and Evolution*, volume 2 of *Bioinformatics*, chapter 1, pages 3–31. Humana Press Inc., 2008.

[NE07] E. P. Nawrocki and S. R. Eddy. Query-dependent banding (QDB) for faster RNA similarity searches. *PLoS Computational Biology*, 3(3):e56, 2007.

[Neb02a] M. E. Nebel. Combinatorial properties of RNA secondary structures. *Journal of Computational Biology*, 9(3):541–574, 2002.

[Neb02b] M. E. Nebel. On a statistical filter for RNA secondary structures. Technical report, Frankfurter Informatik-Berichte, 2002.

[Neb04a] M. E. Nebel. Identifying good predictions of RNA secondary structure. *Proceedings of the Pacific Symposium on Biocomputing*, pages 423–434, 2004.

[Neb04b] M. E. Nebel. Investigation of the Bernoulli-model of RNA secondary structures. *Bulletin of Mathematical Biology*, 66:925–964, 2004.

[NJ80] R. Nussinov and A. B. Jacobson. Fast algorithms for predicting the secondary structure of single-stranded RNA. *Proceedings of the National Academy of Science of the United States of America*, 77(11):6309–6313, 1980.

[NPGK78] R. Nussinov, G. Pieczenik, J. R. Griggs, and D. J. Kleitman. Algorithms for loop matchings. *SIAM Journal on Applied Mathematics*, 35:68–82, 1978.

[NS] M. E. Nebel and A. Scheid. Fast RNA secondary structure prediction using fuzzy stochastic models. *Communications in Computer and Information Science (CCIS)*. In press.

[NS03] M.-J. Nederhof and G. Satta. Probabilistic parsing as intersection. In *Proceedings of the 8th IWPT*, pages 137–148, Nancy, France, 2003.

[NS06] M.-J. Nederhof and G. Satta. Estimation of consistent probabilistic context-free grammars. In *Proceedings of HLT-NAACL'06*, pages 343–350, New York, USA, 2006.

[NS09] M. E. Nebel and A. Scheid. On quantitative effects of RNA shape abstraction. *Theory in Biosciences*, 128(4):211–225, 2009.

[NS11a] M. E. Nebel and A. Scheid. Analysis of the free energy in a stochastic RNA secondary structure model. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, 8(6):1468–1482, 2011.

[NS11b] M. E. Nebel and A. Scheid. Evaluation of a sophisticated SCFG design for RNA secondary structure prediction. *Theory in Biosciences*, 130(4):313–336, 2011.

[NS12] M. E. Nebel and A. Scheid. A n^2 RNA secondary structure prediction algorithm. In *Proceedings of Bioinformatics 2012 – International Conference on Bioinformatics Models, Methods and Algorithms*, pages 66–75, Vilamoura, Portugal, 2012.

[NSW11] M. E. Nebel, A. Scheid, and F. Weinberg. Random generation of RNA secondary structures according to native distributions. *Algorithms for Molecular Biology*, 6(24), 2011.

[NW] M. E. Nebel and F. Weinberg. Algebraic and combinatorial properties of common RNA pseudoknot classes with applications. *Journal of Computational Biology*. To appear.

[NW78] A. Nijenhuis and H. S. Wilf. *Combinatorial Algorithms*. Academic Press, second edition, 1978.

[NYY95] A. Nakaya, K. Yamamoto, and A. Yonezawa. RNA secondary structure prediction using highly parallel computers. *Computer Applications in the Biosciences*, 11:685–692, 1995.

[PB89] C. W. Pleij and L. Bosch. RNA pseudoknots: structure, detection, and prediction. *Methods in Enzymology*, 180:289–303, 1989.

[PBS+06] J. S. Pederson, G. Bejerano, A. Siepel, K. Rosenbloom, K. Lindblad-Toh, E. S. Lander, J. Kent, W. Miller, and D. Haussler. Identification and classification of conserved RNA secondary structures in the human genome. *PLoS Computational Biology*, 2:e33, 2006.

[PFMH04] J. S. Pedersen, R. Forsberg, I. M. Meyer, and J. Hein. An evolutionary model for protein-coding regions with conserved RNA structure. *Molecular Biology and Evolution*, 21:1913–1922, 2004.

[Ple94] C. W. Pleij. RNA pseudoknots. *Current Opinion in Structural Biology*, 4:337–344, 1994.

[PMF+04] J. S. Pedersen, I. M. Meyer, R. Forsberg, P. Simmonds, and J. Hein. A comparative method for finding and folding RNA secondary structures in protein-coding regions. *Nucleic Acids Research*, 32:4925–4936, 2004.

[Pon08] Y. Ponty. Efficient sampling of RNA secondary structures from the Boltzmann ensemble of low-energy: The boustrophedon method. *Journal of Mathematical Biology*, 56:107–127, 2008.

[PRB85] C. W. Pleij, K. Rietveld, and L. Bosch. A new principle of RNA folding based on pseudoknotting. *Nucleic Acids Research*, 13(5):1717–1731, 1985.

[Pre03] D. Prescher. A tutorial on the expectation-maximization algorithm including maximum-likelihood estimation and EM training of probabilistic context-free grammars, 2003.

[PSOJ89] N. R. Pace, D. K. Smith, G. J. Olsen, and B. D. James. Phylogenetic comparative analysis and the secondary structure of ribonuclease P RNA – a review. *Gene*, 82:65–75, 1989.

[PTW99] N. R. Pace, B. C. Thomas, and C. R. Woese. Probing RNA structure, function, and history by comparative analysis. In *The RNA World*, pages 113–141. Cold Spring Harbor Laboratory Press, Cold Spring Harbor, NY, 1999.

[RCM99] J. Rozenski, P. F. Crain, and J. A. McCloskey. The RNA modification database. *Nucleic Acids Research*, 27:196–197, 1999.

[RE99] E. Rivas and S. R. Eddy. A dynamic programming algorithm for RNA structure prediction including pseudoknots. *Journal of Molecular Biology*, 285:2053–2068, 1999.

[RE00] E. Rivas and S. R. Eddy. Secondary structure alone is generally not statistically significant for the detection of noncoding RNAs. *Bioinformatics*, 16(7):583–605, 2000.

[RG04] J. Reeder and R. Giegerich. Design, implementation and evaluation of a practical pseudoknot folding algorithm based on thermodynamics. *BMC Bioinformatics*, 5:104, 2004.

[RGK+03] M. A. Rosenblad, J. Gorodkin, B. Knudsen, C. Zwieb, and T. Samuelsson. SRPDB: Signal recognition particle database. *Nucleic Acids Research*, 31:363–364, 2003.

[RHA+11] C. M. Reidys, F. W. D. Huang, J. E. Andersen, R. C. Penner, P. F. Stadler, and M. E. Nebel. Topology and prediction of RNA pseudoknots. *Bioinformatics*, 27(8):1076–1085, 2011.

[RJ86] L. R. Rabiner and B. H. Juang. An introduction to hidden Markov models. *IEEE ASSP Magazine*, 3:4–16, 1986.

[RLE12] E. Rivas, R. Lang, and S. R. Eddy. A range of complex probabilistic models for RNA secondary structure prediction that include the nearest neighbor model and more. *RNA*, 18(2):193–212, 2012.

[Rot07] P. Rotkiewicz. imol molecular visualization program, 2007.

[RSZ04] J. Ruan, G. D. Stormo, and W. Zhang. An iterated loop matching approach to the prediction of RNA secondary structures with pseudoknots. *Bioinformatics*, 20(1):58–66, 2004.

[San85] D. Sankoff. Simultaneous solution of the RNA folding, alignment and protosequence problems. *SIAM Journal on Applied Mathematics*, 45:810–825, 1985.

[SBEB02] M. Szymanski, M. Z. Barciszewska, V. A. Erdmann, and J. Barciszewski. 5S ribosomal RNA database. *Nucleic Acids Research*, 30:176–178, 2002.

[SBH+94] Y. Sakakibara, M. Brown, R. Hughey, I. S. Mian, K. Sjölander, R. C. Underwood, and D. Haussler. Stochastic context-free grammars for tRNA modeling. *Nucleic Acids Research*, 22:5112–5120, 1994.

[SBT99] S. J. Schroeder, M. E. Burkard, and D. H. Turner. The energetics of small internal loops in RNA. *Biopolymers*, 52:157–167, 1999.

[Sch07] A. Scheid. On moments of the free energy of random RNA secondary structures. Diploma Thesis, University of Kaiserslautern, 2007.

[SHB+98] M. Sprinzl, C. Horn, M. Brown, A. Ioudovitch, and S. Steinberg. Compilation of tRNA sequences and sequences of tRNA genes. *Nucleic Acids Research*, 26:148–153, 1998.

[SHF+84] G. Steger, H. Hofmann, J. Förtsch, H. J. Gross, J. W. Randles, H. L. Sänger, and D. Riesner. Conformational transitions in viroids and virusoids: comparison of results from energy minimization algorithm and from experimental data. *Journal of Biomolecular Structure & Dynamics*, 2(3):543–571, 1984.

[Sip96] M. Sipser. *Introduction to the Theory of Computation*. PWS Publishing, 1996.

[SKMC83] D. Sankoff, J. B. Kruskal, S. Mainville, and R. J. Cedergren. Fast algorithms to determine RNA secondary structures containing multiple loops. In *Time warps, string edits, and macromolecules: the theory and practice of sequence comparison*, chapter 3, pages 93–120. Addison-Wesley, Reading, MA, 1983.

[SKV+11] Z. Sukosd, B. Knudsen, M. Vaerum, J. Kjems, and E.S. Andersen. Multithreaded comparative RNA secondary structure prediction using stochastic context-free grammars. *BMC Bioinformatics*, 12(103), 2011.

[SM07] C. Sutton and A. McCallum. An introduction to conditional random fields for relational learning. In L. Getoor and B. Taskar, editors, *Introduction to Statistical Relational Learning*. MIT Press, 2007.

[SMFK91] H. Seki, T. Matsumura, M. Fujii, and T. Kasami. On multiple context-free grammars. *Theoretical Computer Science*, 88:191–229, 1991.

[SMW+10] R. Salari, M. Möhl, S. Will, S. C. Sahinalp, and R. Backofen. Time and space efficient RNA-RNA interaction prediction via sparse folding. In *Proceedings of RECOMB'10*, pages 473–490, 2010.

[SN08] A. Scheid and M. E. Nebel. On abstract shapes of RNA. Technical report, University of Kaiserslautern, 2008.

[SN12a] A. Scheid and M. E. Nebel. Evaluating the effect of disturbed ensemble distributions on SCFG based statistical sampling of RNA secondary structures. *BMC Bioinformatics*, 13(159), 2012.

[SN12b] A. Scheid and M. E. Nebel. Statistical RNA secondary structure sampling based on a length-dependent SCFG model. Technical report, University of Kaiserslautern, 2012.

[SS78] A. Salomaa and M. Soittola. *Automata-theoretic aspects of formal power series*. Springer, 1978.

[ST95] M. J. Serra and D. H. Turner. Predicting thermodynamic properties of RNA. *Methods in Enzymology*, 259:242–261, 1995.

[Sto95] A. Stolcke. An efficient probabilistic context-free parsing algorithm that computes prefix probabilities. *Computational Linguistics*, 21(2):165–201, 1995.

[SV05] M. Sprinzl and K. Vassilenko. Compilation of tRNA sequences and sequences of tRNA genes. *Nucleic Acids Research*, 33:139–140, 2005.

[SVR+06a] P. Steffen, B. Voß, M. Rehmsmeier, J. Reeder, and R. Giegerich. RNAshapes 2.1.1 manual, February 2006.

[SVR+06b] P. Steffen, B. Voß, M. Rehmsmeier, J. Reeder, and R. Giegerich. RNAshapes: an integrated RNA analysis package based on abstract shapes. *Bioinformatics*, 22(4):500–503, 2006.

[SW78] P. R. Stein and M. S. Waterman. On some new sequences generalizing the Catalan and Motzkin numbers. *Discrete Mathematics*, 26:216–272, 1978.

[Tas05] B. Taskar. *Learning structured prediction models: a large margin approach*. PhD thesis, Stanford University, Stanford, CA, USA, 2005.

[TB99] I. Tinoco, Jr. and C. Bustamante. How RNA folds. *Journal of Molecular Biology*, 293:271–281, 1999.

[TB05] B. J. Tucker and R. R. Breaker. Riboswitches as versatile gene control elements. *Current Opinion in Structural Biology*, 15:342–348, 2005.

[TCKG05] B. Taskar, V. Chatalbashev, D. Koller, and C. Guestrin. Learning structured prediction models: A large margin approach. In *Proceedings of the 22nd International Conference on Machine Learning*, 2005.

[THG07] E. Torarinsson, J. H. Havgaard, and J. Gorodkin. Multiple structural alignment and clustering of RNA sequences. *Bioinformatics*, 23:926–932, 2007.

[THK+04] M. Tamura, D. K. Hendrix, P. S. Klosterman, N. R. Schimmelman, S. E. Brenner, and S. R. Holbrook. SCOR: Structural classification of RNA, version 2.0. *Nucleic Acids Research*, 32:D182–4, 2004.

[TLA+08] M. Taufer, A. Licon, R. Araiza, B. Mireles, F. H. D. van Batenburg, A. Gultyaev, and M.-Y. Leung. PseudoBase++: an extension of PseudoBase for easy searching, formatting and visualization of pseudoknots. *Nucleic Acids Research*, 37:D127–D135, 2008.

[TM07] R. Tyagi and D. H. Mathews. Predicting helical coaxial stacking in RNA multibranch loops. *RNA*, 13(7):939–951, 2007.

[TUL71] I. Tinoco, O. Uhlenbeck, and M. D. Levine. Estimation of secondary structure in ribonucleic acids. *Nature*, 230:362–367, 1971.

[Val75] L. Valiant. General context-free recognition in less than cubic time. *Journal of Computer and System Sciences*, 10:308–315, 1975.

[vBGP+00] F. H. D. van Batenburg, A. P. Gultyaev, C. W. A. Pleij, J. Ng, and J. Oliehoek. PseudoBase: a database with RNA pseudoknots. *Nucleic Acids Research*, 28(1):201–204, 2000.

[vBGP01] F. H. D. van Batenburg, A. P. Gultyaev, and C. W. A. Pleij. PseudoBase: structural information on RNA pseudoknots. *Nucleic Acids Research*, 29(1):194–195, 2001.

[VC85] G. Viennot and M. Vauchaussade De Chaumont. Enumeration of RNA secondary structures by complexity. *Mathematics in medicine and biology, Lecture Notes in Biomathematics*, 57:360–365, 1985.

[VGR06] B. Voß, R. Giegerich, and M. Rehmsmeier. Complete probabilistic analysis of RNA shapes. *BMC Biology*, 4(5), 2006.

[vR79] C. J. van Rijsbergen. *Information Retrieval*. London Butterworths, London, 1979.

[Wat78] M. S. Waterman. Secondary structure of single-stranded nucleic acids. *Advances in Mathematics Supplementary Studies*, 1:167–212, 1978.

[WdPWW02] J. Wuyts, Y. Van de Peer, T. Winkelmans, and R. De Wachter. The european database on small subunit ribosomal RNA. *Nucleic Acids Research*, 30(1):183–185, 2002.

[WFHS99] S. Wuchty, W. Fontana, I. Hofacker, and P. Schuster. Complete suboptimal folding of RNA and the stability of secondary structures. *Biopolymers*, 49:145–165, 1999.

[WH04] S. Washietl and I. L. Hofacker. Consensus folding of aligned sequences as a new measure for the detection of functional RNAs by comparative genomics. *Journal of Molecular Biology*, 342:19–30, 2004.

[Wil45] F. Wilcoxon. Individual comparisons by ranking methods. *Biometrics Bulletin*, 1:80–83, 1945.

[Wil10] S. Wild. An Earley-style parser for solving the RNA-RNA interaction problem. Bachelor Thesis, University of Kaiserslautern, 2010.

[WM10] N. J. P. Wiebe and I. M. Meyer. Transat – A method for detecting the conserved helices of functional RNA structures, including transient, pseudo-knotted and alternative structures. *PLoS Computational Biology*, 6(6):e1000823, 2010.

[WMJ11] E. Westhof, B. Masquida, and F. Jossinet. Predicting and modeling RNA architecture. *Cold Spring Harbor Perspectives in Biology*, 2011.

[WN10] F. Weinberg and M. E. Nebel. Non uniform generation of combinatorial objects. Technical report, University of Kaiserslautern, 2010.

[WN11] F. Weinberg and M. E. Nebel. Applying length-dependent stochastic context-free grammars to RNA secondary structure prediction. *Algorithms*, 4(4):223–238, 2011.

[Woo10] S. A. Woodson. Compact intermediates in RNA folding. *Annual Review of Biophysics*, 39:61–77, 2010.

[Woo11] S. A. Woodson. RNA folding pathways and the self-assembly of ribosomes. *Accounts of Chemical Research*, 44(12):1312–1319, 2011.

[WRdP+01] J. Wuyts, P. De Rijk, Y. Van de Peer, T. Winkelmans, and R. De Wachter. The european large subunit ribosomal RNA database. *Nucleic Acids Research*, 29(1):175–177, 2001.

[WS86] M. S. Waterman and T. F. Smith. Rapid dynamic programming algorithms for RNA secondary structure. *Advances in Applied Mathematics*, 7:455–464, 1986.

[WT86] A. L. Williams, Jr. and I. Tinoco, Jr. A dynamic programming algorithm for finding alternative RNA secondary structures. *Nucleic Acids Research*, 14:299–315, 1986.

[WT94] A. Walter and D. Turner. Sequence dependence of stability for coaxial stacking of RNA helixes with watson-crick base paired interfaces. *Biochemistry*, 33:12715–12719, 1994.

[WZZU07] Y. Wexler, C. Zilberstein, and M. Ziv-Ukelson. A study of accessible motifs and RNA folding complexity. *Journal of Computational Biology*, 14(6):856–872, 2007.

[XSB+98] T. Xia, J. SantaLucia, Jr., M. E. Burkard, R. Kierzek, S. J. Schroeder, X. Jiao, C. Cox, and D. H. Turner. Thermodynamic parameters for an expanded nearest-neighbor model for formation of RNA duplexes with watson-crick base pairs. *Biochemistry*, 37:14719–14735, 1998.

[You67] D. H. Younger. Recognition and parsing of context-free languages in time n^3. *Information and Control*, 10(2):372–375, 1967.

[ZGEZU11] S. Zakov, Y. Goldberg, M. Elhadad, and M. Ziv-Ukelson. Rich parameterization improves RNA structure prediction. *Journal of Computational Biology*, 18(11):1525–1542, 2011.

[ZGK+03] C. Zwieb, J. Gorodkin, B. Knudsen, J. Burks, and J. Wower. tmRDB (tmRNA database). *Nucleic Acids Research*, 31:446–447, 2003.

[ZMT99] M. Zuker, D. H. Mathews, and D. H. Turner. Algorithms and thermodynamics for RNA secondary structure prediction: A practical guide. In J. Barciszewski and B. F. C. Clark, editors, *RNA Biochemistry and Biotechnology*, NATO ASI Series, pages 11–43. Kluwer Academic Publishers, Dordrecht, NL, 1999.

[ZS81] M. Zuker and P. Stiegler. Optimal computer folding of large RNA sequences using thermodynamics and auxiliary information. *Nucleic Acids Research*, 9:133–148, 1981.

[ZS84] M. Zuker and D. Sankoff. RNA secondary structures and their prediction. *Bulletin of Mathematical Biology*, 46:591–621, 1984.

[ZTZU11] S. Zakov, D. Tsur, and M. Ziv-Ukelson. Reducing the worst case running times of a family of RNA and CFG problems, using Valiant's approach. *Algorithms for Molecular Biology*, 6(20), 2011.

[ZUGVWS08] M. Ziv-Ukelson, I. Gat-Viks, Y. Wexler, and R. Shamir. A faster algorithm for RNA co-folding. In *Proceedings of the 8th international workshop on Algorithms in Bioinformatics*, WABI'08, pages 174–185, 2008.

[ZUGVWS10] M. Ziv-Ukelson, I. Gat-Viks, Y. Wexler, and R. Shamir. A faster algorithm for simultaneous alignment and folding of RNA. *Journal of Computational Biology*, 17(8):1051–1065, 2010.

[Zuk86] M. Zuker. RNA folding prediction: The continued need for interaction between biologists and mathematicians. *Lectures on Mathematics in the Life Sciences*, 17:87–124, 1986.

[Zuk89a] M. Zuker. Computer prediction of RNA structure. In J. E. Dahlberg and J. N. Abelson, editors, *RNA Processing*, volume 180 of *Methods in Enzymology*, pages 262–288. Academic Press, San Diego, 1989.

[Zuk89b] M. Zuker. On finding all suboptimal foldings of an RNA molecule. *Science*, 244:48–52, 1989.

[Zuk94] M. Zuker. Prediction of RNA secondary structure by energy minimization. In A. M. Griffin and H. G. Griffin, editors, *Computer Analysis of Sequence Data*, Methods in Molecular Biology, pages 267–294. Humana Press Inc., 1994.

[Zuk00] M. Zuker. Calculating nucleic acid secondary structure. *Current Opinion in Structural Biology*, 10:303–310, 2000.

[Zuk03] M. Zuker. Mfold web server for nucleic acid folding and hybridization prediction. *Nucleic Acids Research*, 31(13):3406–3415, 2003.

List of Tables

List of Figures

List of Algorithms

Curriculum Vitæ

Education

08/1989 – 07/1993	Elementary school: Grundschule Scheuern, Germany
08/1993 – 06/2002	Secondary school: Hochwald-Gymnasium Wadern, Germany Certificate: High-school diploma (*Abitur*)
10/2002 – 01/2008	Study of Computer Science, University of Kaiserslautern, Germany Degree: Dipl.-Inf. (Diplom-Informatikerin) Diploma thesis: *On Moments of the Free Energy of Random RNA Secondary Structures*
04/2006 – 06/2012	PhD student in the Computer Science Department Graduate School (*Promotionsprogramm Informatik*), University of Kaiserslautern, Germany
since 10/2008	Doctoral candidate of the Computer Science Department, University of Kaiserslautern, Germany
since 10/2010	Scholarship of *Nachwuchsförderprogramm* (program for supporting young researchers), Carl Zeiss Foundation, Germany

Experience

2004	Teaching assistant for *Grundlagen der Programmierung* (foundations of programming)
2004 – 2005, 2008 – 2009	Teaching assistant for *Entwurf und Analyse von Algorithmen* (design and analysis of algorithms)
2009 – 2010	Mentoring of freshman in the *Lerntechniken* (learning techniques) program
2009 – 2010	Supervision of seminar *Algorithmen der Bioinformatik* (algorithms in bioinformatics)

Kaiserslautern, October 16, 2012