



# Ensemble Algorithms and Analytic Combinatorics in RNA Bioinformatics and Beyond

Yann Ponty

## ► To cite this version:

Yann Ponty. Ensemble Algorithms and Analytic Combinatorics in RNA Bioinformatics and Beyond. Bioinformatics [q-bio.QM]. Université Paris-Saclay, 2020. tel-03219977v1

**HAL Id: tel-03219977**

**<https://theses.hal.science/tel-03219977v1>**

Submitted on 6 May 2021 (v1), last revised 19 May 2021 (v2)

**HAL** is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.



Distributed under a Creative Commons Attribution 4.0 International License

# Ensemble Algorithms and Analytic Combinatorics in RNA Bioinformatics and Beyond

Yann Ponty

An *Habilitation à Diriger des Recherches* thesis of Paris-Saclay Université,  
defended on May 22<sup>nd</sup> 2020 and evaluated by:

Cédric CHAUVE, Simon Fraser University, Canada  
Léonid CHINDELEVITCH, Imperial College London, UK  
Johanne COHEN, CNRS/Université Paris-Saclay  
Philippe DUCHON, Université de Bordeaux  
Ivo HOFACKER, University of Vienna, Autriche  
Daniel MERKLE, University of South Denmark  
Cyril NICAUD, Université Paris-Est, Marne la Vallée  
Peter STADLER, University of Leipzig, Germany

## Abstract

Predictive Bioinformatics represents a major field of applications for combinatorial optimization techniques. Very often, an ensemble perspective, which not only consider the optimal solution but also fully embraces the set of suboptimal solutions, needs to be adopted. In this *Habilitation à Diriger des Recherches*, I present a series of algorithmic and combinatorial contributions, inspired by problems and questions arising in the study of RiboNucleic Acids (RNAs), in particular pertaining to their structural properties at the thermodynamic equilibrium.

I first describe a collection of generic and applied algorithmic techniques, enabling the efficient computation of statistical properties within search spaces. Such computations can be exact, or rely on unbiased estimates produced, using constrained sampling strategies, and are founded on on a combinatorial (re-)interpretation of dynamic programming schemes. I then adopt a purely combinatorial point of view over search spaces, and establish asymptotic properties of classes of discrete objects arising in Bioinformatics, showcasing the unreasonable power of (a subset of) analytic combinatorics. Finally, I conclude with a collection of algorithmic results, obtained through the application of a wide array of techniques, in the context of RNA design, a field focused on combinatorial problems that are at the same time original, difficult, and relevant to the modern goals of biology and medicine.

# Contents

<b>1</b>	<b>Introduction</b>	<b>5</b>
1.1	Predictive Bioinformatics and its foundations . . . . .	5
1.2	A crash course into RNA structure prediction . . . . .	7
1.2.1	RNA structure . . . . .	8
1.2.2	RNA folding prediction models and paradigms . . . . .	9
1.2.3	The secondary structure . . . . .	11
1.2.4	RNA 2D structure prediction from thermodynamic principles . . . .	13
1.3	Outline . . . . .	16
<b>2</b>	<b>Ensemble dynamic programming: techniques and analyses</b>	<b>17</b>
2.1	A formal framework and basic algorithms . . . . .	19
2.1.1	Dynamic programming as a rewriting system . . . . .	20
2.1.2	Search space and suitability for ensemble applications . . . . .	22
2.1.3	Classic optimization . . . . .	24
2.2	Exact computation of Ensemble properties . . . . .	25
2.2.1	Computing the partition function. . . . .	26
2.2.2	Probabilities in Boltzmann-Gibbs distributions (inside-outside) . . .	27
2.2.3	Ensemble centroid and Maximum Expected Accuracy (MEA) . . . .	29
2.2.4	General moments of additive scores [150] . . . . .	31
2.2.5	Classified DP with the Discrete Fourier Transform (DFT) [173, 174]	34
2.3	Probabilistic estimates . . . . .	37
2.3.1	Foreword: On the number of samples [165] . . . . .	38
2.3.2	Statistical sampling . . . . .	39
2.3.3	Non redundant sampling [118, 133, 165] . . . . .	41
2.3.4	Adaptive sampling of constrained sequences [18, 90, 158] . . . . .	45
2.4	Applications of ensemble dynamic programming . . . . .	48
2.4.1	(Recursive) simple type pseudoknots and kissing hairpins [150] . .	49
2.4.2	Dual partition functions and evolutionary robustness of RNAs . . .	51
2.4.3	Unambiguous tree alignments [32, 33] . . . . .	53
<b>3</b>	<b>RNA design</b>	<b>56</b>
3.1	Why do we design RNAs? . . . . .	58
3.1.1	The potential of RNA design for statistical evolutionary studies . . .	59

3.1.2	The different flavors of RNA design . . . . .	59
3.2	Exact combinatorial negative design [85, 86] . . . . .	61
3.2.1	Basic results . . . . .	62
3.2.2	Design as a tree coloring problem . . . . .	63
3.2.3	Structure-approximating design . . . . .	65
3.3	Stochastic positive design under constraints . . . . .	66
3.3.1	Sampling from the dual-partition function [117, 158, 160] . . . . .	67
3.3.2	Avoiding and forcing motifs [210] . . . . .	69
3.3.3	Multiple structures and extended design principles [88, 90] . . . . .	71
<b>4</b>	<b>Constrained random generation through rejection and beyond</b>	<b>73</b>
4.1	It's not you, it's me: The unreasonable power of rejection . . . . .	76
4.1.1	Correcting a (bounded) bias . . . . .	77
4.1.2	Complexity aspects . . . . .	78
4.2	Rejecting in dimension one and then some. . . . .	81
4.2.1	(Combinatorial) Boltzmann sampling . . . . .	81
4.2.2	Multidimensional Boltzmann sampling . . . . .	82
4.2.3	Beyond simplicity . . . . .	88
4.3	Applications of multidimensional sampling . . . . .	89
4.3.1	Controlling the GC-content [88, 90, 158, 160, 192, 193] . . . . .	89
4.3.2	Dinucleotides content of protein-coding RNAs [209]. . . . .	90
4.4	Redundancy in weighted sampling and countermeasures . . . . .	92
4.4.1	Collisions in weighted languages [58, 78] . . . . .	92
<b>5</b>	<b>Asymptotic properties of RNA secondary structures and other trees</b>	<b>96</b>
5.1	Basic tools in enumerative and analytic combinatorics . . . . .	97
5.1.1	Context-free languages. . . . .	97
5.1.2	Enumeration and generating functions. . . . .	99
5.1.3	Basic singularity analysis . . . . .	101
5.1.4	Useful extensions and shortcuts . . . . .	105
5.2	Asymptotic combinatorics of RNA secondary structures . . . . .	111
5.2.1	Expected 5'–3' distance [38] . . . . .	113
5.2.2	RNA network properties [180] . . . . .	115
5.2.3	RNA shapes: Abstract representations of structures [120] . . . . .	119
5.2.4	Enumerating designable structures [206] . . . . .	123
<b>6</b>	<b>Conclusion and perspectives</b>	<b>127</b>
	<b>Bibliography</b>	<b>129</b>

## Foreword

The French academic system has enjoyed a long and eventful history, leading to many charming peculiarities. One such specificity is the requirement of junior tenured scientists to successfully defend an *Habilitation à Diriger des Recherches* (HDR) before being allowed supervise doctoral candidates in an official capacity. As part of the requirements for such a defense, the candidate produces a manuscript summarizing the research results obtained since his PhD. This is my contribution towards applying for an HDR in Computer Science at Université Paris-Sud. The individual nature of the evaluation dictates the use of the first-person singular, but an overwhelming majority of those results hinge critically on contributions from a community of collaborators, which I have attempted to acknowledge profusely.

The structure, ambition, scope and length of an HDR vary substantially between disciplines, institutions, and follow personal aesthetics. Some will elicit to contribute a general introduction for a collection of articles, while others will interlace past results with novel research material, trying to use the opportunity to outline general theories. Some will embrace the diversity of their past contributions, while others will strive to provide a unified perspective over research projects spanning one to two decades.

My ambition for this document is, by formalizing some of the intuitions underlying my past contributions, to reveal their triviality and, more seriously, contribute firm foundations for the systematic design and analysis of algorithms in RNA Bioinformatics and beyond. My main focus is therefore on the description of techniques, either classic in other fields or contributed over the course of my young career as a scientist, showcasing their power and level of generality by a brief description of their application in the context of RNA bioinformatics.

For these reasons, and partly because compact does not necessarily mean simple, I apologize in advance to the casual reader for explanations and digressions which may appear, at times, unnecessarily technical. It is unfortunately the price to pay to achieve my intended goal of using this document to lay out stable and explicit foundations for futures contributions to Bioinformatics.

# Chapter 1

## Introduction

Bioinformatics as a field of study is the poster child of **interdisciplinarity**. It is unified by an overarching objective to automate the processing and analysis of biological data, and plays an essential part in the production of knowledge in modern **Biology**. In the context of **Molecular Biology**, it is informed by models stemming from **Biophysics** and **Chemistry**, quantitatively parameterized by computational methods in **Statistics** and **Probability theory (Machine Learning)**, or analyzed at a theoretical level using **Theoretical Physics** and **Discrete Mathematics** techniques. Such quantitative models, typically in conjunction with parsimony arguments, represent the foundations of **predictive algorithmic methods**.

### 1.1 Predictive Bioinformatics and its foundations

**Predictive Bioinformatics** strives to produce methods which, from empirical data, predict phenomena far beyond our capacity for direct observations. Examples of such limitations include events that occur at the nanometer scale, in the distant past, or whose observation significantly interferes with (e.g. kill) a living system of interest. Given an established model, consistent with current biological knowledge, **predictive methods in bioinformatics** embrace some notion of search space induced by the input data. They elect one or several element(s) of the search space by maximizing some notion of score, usually analogous to a probability, provided by the model. The produced solution constitutes a *best bet* under (possibly implicit) assumptions of the model. Such methods are typically trained on reference data sets, for which a **ground-truth** is known, to calibrate a widely-varying set of parameters, and **validated** on independent data sets as an empirical test of their accuracy and, importantly, capacity of generalization.

Once a predictive method has been deemed satisfactory, a **leap of faith** occurs, following which the model and method are treated as uniformly correct. The subsequent predictions are considered as reflective of Nature itself, and treated as primary data in

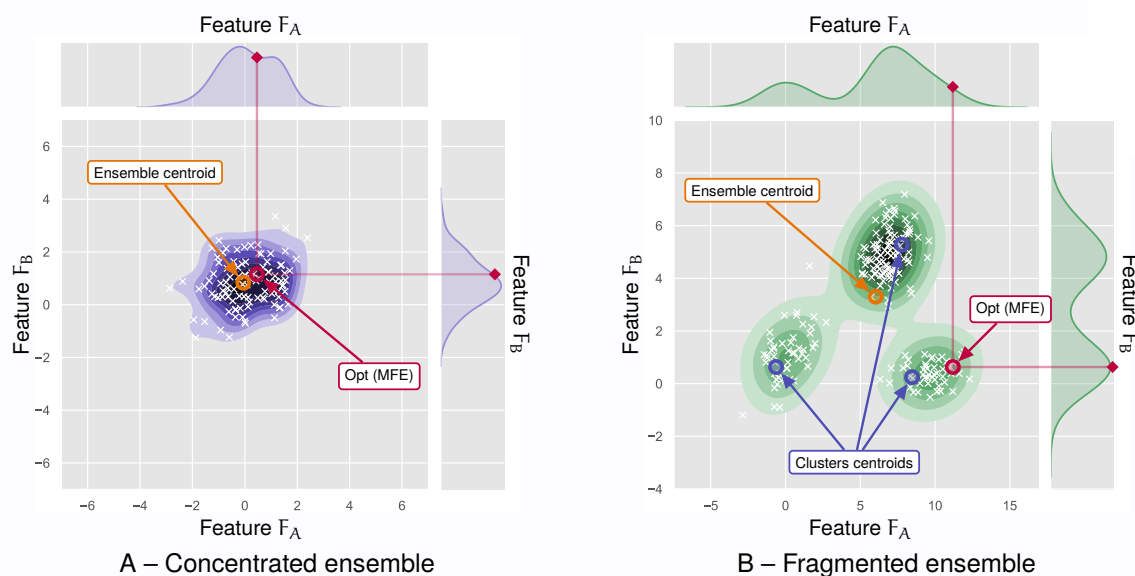


Figure 1.1: **Rationale of ensemble analyses.** In combinatorial optimization, near-optimal solutions can be **representative** of the ensemble (A), with suboptimals concentrating around the optimal. Ensembles may also be fragmented (B), i.e. near-optimal solutions regroup into clusters whose representatives may differ from the optimal with respect to some features  $F_A$  and  $F_B$  of interest. The optimal may then be an **outlier**, and should be **treated with caution**, or even **disregarded** in favor of the ensemble centroid.

the formulation of hypotheses and functional models in Biology. This abrupt change of perspective, treating a method with uttermost paranoia during the validation phase and with unabashed confidence in subsequent analyses, may appear dangerously optimistic at first sight. In fact, it is usually the root cause of the *cultural clash* experienced by scientists transitioning from exact sciences to Bioinformatics. However, Biology as field has evolved a **remarkable robustness** to misleading observations and appears, sometimes through impassioned controversies, to overcome biased observations caused by imperfect experimental devices, among which Bioinformatics methods can now be counted.

Indeed, predictions produced by Bioinformatics methods are typically falsifiable, i.e. they can be contradicted by future produced experimental data. This partly justifies the popularity of **exact algorithms** in molecular biology. Indeed, in the context of an exact optimization, failure to predict can be interpreted as a shortcoming of the underlying quantitative model, motivating and directing further refinements. By contrast, ill-defined optimization schemes (e.g. machine learning or heuristics), convolve possible modeling errors with optimization errors. Failure to predict does not necessarily point towards flaws of the model, and does not allow to refine them, as an improved model may induce worse optimization performances (and vice-versa, a broken clock being right twice a day), missing an opportunity for future developments.

Predictive optimization-based methods, however, suffer from a severe shortcoming in the

context of combinatorial search spaces. Indeed, they ultimately output a single solution, presumed to be optimally-likely in some sense. However, the dominating nature of the proposed solution may show limited robustness to, even modest, changes of the objective function, whose parameters are usually learned from data and thus subject to experimental errors. Moreover, an optimal solution may be poorly representative of the subspace of near-optimal solutions, so that it may be more advisable to consider a suboptimal – representative – solution rather than an optimal – non-representative – one.

These observations constitute the main motivations behind the development of **ensemble methods** in combinatorial Bioinformatics, treating the whole search space as an object of study rather than merely solving a *needle in a haystack* optimization problem. To that purpose, a probability distribution is postulated over the search space, and probabilistic analyses are performed. The optimal solution may remain relevant, but ensemble methods allow to go further, and assess notions of **support** for any given solution, or how distant features of the optimal solution are from the **centroid of the ensemble**. Indeed, as shown in Figure 1.1, ensembles associated with a given instance may either be concentrated around the optimal, or fragmented, leading to multiple clusters of diverse solutions, and multimodal distributions for features of interest.

## 1.2 A crash course into RNA structure prediction

The Bioinformatics of **RiboNucleic Acids** (RNAs) represent a very natural realm of application, and a source of constant inspiration, for ensemble analyses.

RNA constitutes a category of biomolecules, abstracted as **sequence of nucleotides** Adenine (A), Cytosine (C), Guanine (G) and Uracil (U), initially **transcribed** from a DNA template and further processed before reaching their cellular environment. They can form stable complexes with proteins, but also DNA and other RNAs, allowing them to **regulate genetic expression**. They can also perform **enzymatic functions**, i.e. process other molecules (or themselves), act as biosensors by undergoing conformational changes upon binding with small metabolites, and store the **entire genetic material** of certain viruses (e.g. HIV, SARS, 2019 nCoV).

Their dual capacity to store and process information is unmatched, leading current theories in evolution to consider RNA as the most likely candidate at the **origin of life** [93]. Such a versatility, illustrated in Figure 1.2, not only stems from the combinatorial nature of RNA sequences but also from its capacity to adopt one or several well-defined **structures**, driving the specificity of its interactions with other actors of the cellular world.



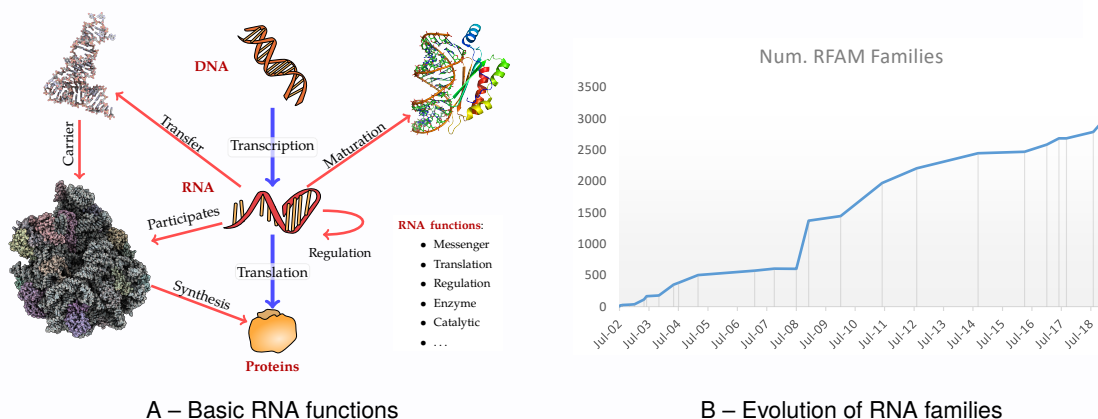


Figure 1.2: **Beyond coding for proteins, a growing list of RNA functions.** The versatility of RNA (A) is confirmed by the sustained growth (B) of the number of functional families in the Rfam database [84, 106], a majority of which are not coding for proteins, and associated with a conserved well-defined secondary structure.

### 1.2.1 RNA structure

Following Tinoco and Bustamante [184], RNA is believed to **fold** in a hierarchical fashion where **canonical base pairs**, mediated by hydrogen bonds, initially form a tree-like architecture called the **secondary structure**. Later in the folding process, the secondary structure is completed by complex topological motifs, including **pseudoknots** and **non-canonical** base pairs and motifs [114, 115]. Due to its combinatorial nature, the secondary structure is found at the core of successful computational methods for predicting the 3D structure of RNA [131]. It also supports the discovery of new functional families of **non-coding RNAs**, RNAs that do not (only) encode a protein, but (also) act directly on their environment through participation in catalytic and/or regulatory functions. A conserved **consensus secondary structure** is indeed a central object for many families in the Rfam database [84, 106], and a crucial element of the design of covariance models [62] used to find homologous RNAs across sequenced genomes.

At a 3D level, structure modeling is tackled using a mixture of comparative methods [134] and low-throughput/high resolution experimental techniques such as X-ray crystallography, Nuclear Magnetic Resonance (NMR), or Cryogenic Electron Microscopy (Cryo-EM). However, those techniques are extremely time-consuming, and require a preparation of molecules that may either not be feasible for certain RNAs, or can interfere with the folding process itself. These difficulties induce a **growing gap** between the number of functional families ( $\pm 3000$  as of Jan 2020), many of which requiring the adoption of a specific structure, and the number of families with experimentally-resolved 3D model available for at least one of their members (99 as of Jan 2020).

Recently, high throughput/low resolution alternatives have been proposed in the form

of improved **chemical probing** protocols, notoriously including SHAPE probing [47, 176, 198]. Those methods expose RNA to a chemical reagent, whose affinity towards individual nucleotides depend on the structure (and therefore partly reveal it, albeit in a stochastic and highly noisy fashion) and can be quantified using DNA and RNA sequencing technologies. The end-result of those methods are **1D reactivity** profiles that are not sufficient to fully characterize a structure, but greatly informative for further (computational) modeling.

### 1.2.2 RNA folding prediction models and paradigms

For all the aforementioned reasons, computational **structure prediction** methods are very relevant to the current objectives and challenges of RNA Bioinformatics. In the context of molecular biology, RNA structure prediction is mainly concerned with the prediction of one (or several) functional fold(s) for a given molecule. Since the influence of the cellular environment on RNA folding is difficult to fully characterize, and even harder to capture computationally, popular *in silico* methods adopt an approach inspired by **statistical mechanics**. They focus on stable conformations, having **low free-energies**, under the rationale that unstable structures are unlikely to be recognized by their partners, and play a reliable role in important phenotypic effects.

This notion can be formalized by considering the **Boltzmann-Gibbs distribution**, where any possible structure  $S$  for a given RNA  $w$  is observed with probability

$$\mathbb{P}_w(S) \propto e^{-E(w,S)/RT}, \quad (1.1)$$

where  $E(w, S)$  represents the free-energy in  $\text{kcal.mol}^{-1}$  of the  $(w, S)$  pair,  $R$  the Boltzmann constant ( $1.9872 \cdot 10^{-3} \text{ kcal.mol}^{-1}.\text{K}^{-1}$ ) and  $T$  the absolute temperature in Kelvin. This distribution can be thought as the **stationary distribution** of a **continuous time Markov chain**, illustrated in Figure 1.3 on a toy example. In this process, a transition between two conformations is chosen with probability/rate that only depend on the energy difference, thus respecting a **detailed balance**. Starting from the fully-unpaired/empty structure<sup>1</sup>, having initial probability 1, the probability mass diffuses within the space of conformations, and ultimately reaches the **thermodynamic equilibrium** when probabilities no longer evolve with time. The evolution with time of probabilities/concentrations can be determined using efficient numerical integration [201], as illustrated in Figure 1.3.B.

This vision, inspired by statistical mechanics, is at the core of **historical paradigms** in the field of RNA structure prediction:

<sup>1</sup>More sophisticated models may consider a co-transcriptional folding [111] of the nascent transcript, leading to a non-binary initial distribution at the beginning of the process.

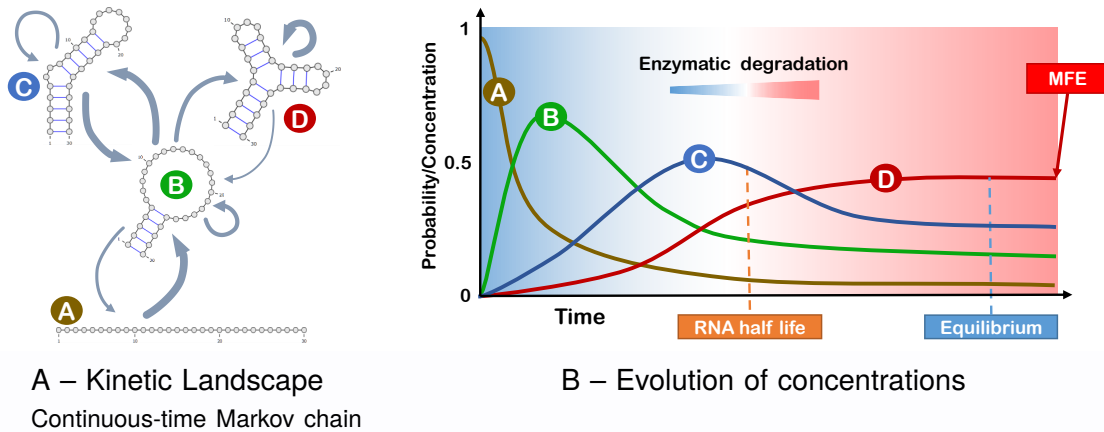


Figure 1.3: **RNA folding paradigms.** The process of RNA folding can be abstracted as a Continuous Time Markov Chain (CTMC – A), over a discrete state space consisting of (a subset of) the secondary structures adopted by an RNA. Starting from an initial distribution, typically assigning full probability to the open chain, kinetic studies attempt to characterize the evolution of the probability distribution with time (B). At the thermodynamic equilibrium, the stationary distribution of the CMTC is reached and probabilities cease to evolve with time. The most probable structure is then the most stable one, i.e. the MFE structure. However, RNA degradation may occur before the equilibrium is reached, and the dominant structures at finite time may represent more promising candidates for functional hypotheses.

**MFE paradigm.** Early computational methods for structure prediction [95, 143, 213] focused their effort on producing (one of) the optimally stable, or **Minimum Free-Energy (MFE)**, structure(s), under the rationale that the MFE structure has highest probability at the thermodynamic equilibrium;

**Boltzmann ensemble paradigm.** While optimal, the probability of the MFE structure alone can be abysmally small, and decreases exponentially with the RNA length. The MFE structure can therefore be **overwhelmed**, at the thermodynamic equilibrium, by a set of alternative structures having comparable stability and, possibly, very different characteristics. This motivates studies of the **expected properties** in the Boltzmann-Gibbs distribution, hinging on an efficient computation of the **partition function** [130]

$$\mathcal{Z} = \sum_S e^{-E(w,S)/RT}.$$

When these properties diverge from those of the MFE, it may be more relevant to consider structures that are more accurate representatives of the ensemble of structures. One such structure is the **Maximum Expected Accuracy** structure [55, 121], the structure minimizing some notion of expected distance to the rest of the ensemble. Alternatively a statistically representative, Boltzmann distributed, set of structures can be randomly generated, clustered at a structural level to identify

alternative conformations, and eliminate outliers. A **centroid structure** can then be elected for each cluster [52];

**The kinetics paradigm.** More recently, there has been a growing awareness of the importance of **out-of-equilibrium effects** in the function(s) carried out by RNA. Indeed, in a cellular context, RNA is constantly transcribed and degraded by enzymes. Depending on the precise dynamics of these concurrent processes, a population of RNAs may simply be degraded before reaching its stationary distribution, so the thermodynamic paradigm may not provide an accurate picture of the functional structure. This generally applies to RNAs whose **energy landscapes** feature substantial **barriers**, leading to a slow convergence towards the equilibrium. Kinetics analyses are thus concerned with the structural behavior of RNA *before* reaching the equilibrium.

Examples of kinetics effects are suspected to include **riboswitches**, bistables RNAs which are observed in *on* and *off* conformations in the presence/absence of a **ligand**, a small molecule, believed to have insufficient contribution to the free-energy to invert the relative stabilities of the *on* and *off* states. Current models are thus based on kinetics, and postulate that the ligand modifies an energy barrier, leading to a faster/slower convergence towards the thermodynamic equilibrium [65]. More generally, **co-transcriptional folding**, the folding of RNA during transcription, reveals the importance of kinetic effects. Indeed, this phenomenon would be without effect at the thermodynamic equilibrium, since the stationary distribution of a (ergodic) Markov chain does not depend on its initial distribution, so it would should not matter at the equilibrium.

In the rest of this document, I will mainly focus on algorithmic strategies relevant to the MFE and Boltzmann equilibrium paradigms, although certain were have designed with kinetics analysis in mind [133, 180]. Kinetics analyses are indeed much more time-consuming, and are associated with computational problems that are routinely NP-hard [126]. Efficient heuristics and methods for analyzing kinetics are, however, the object of ongoing projects within the RNA Bioinformatics community.

### 1.2.3 The secondary structure

A **RNA secondary structure** of size  $n$  represents the outcome of a folding process, and focuses on a subset of **base-pairs**, mediated by hydrogen bonds. For essentially computational reasons [3, 123, 175] this definition forbids crossing base pairs, also called **pseudoknots** due to their ability to induce complex topologies [205]. Moreover, any nucleotide can only be involved in a single base pair, since additional partners would involve non-canonical edges [114]. Finally, most definitions rule out base pairs between

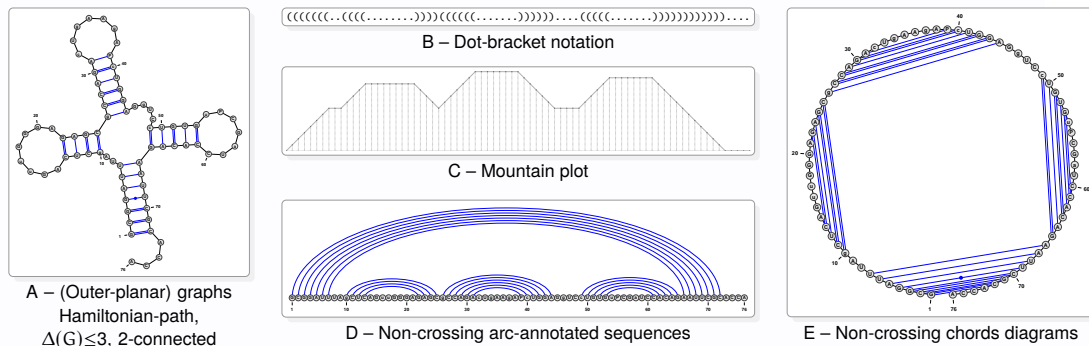


Figure 1.4: Various representations for RNA secondary structures.

proximate positions, due to **steric effects** inducing geometric constraints, leading to a **minimal number**  $\theta$  of unpaired positions between paired positions.

**Definition 1.2.1 (RNA secondary structure):** An RNA secondary structure  $S$  of length  $n$  is a set of base-pairs  $(i, j)$ ,  $1 \leq i < j \leq n$ , such that:

- Each position is **monogamous**,  $\forall (i, j) \neq (i', j') \in S : \{i, j\} \cap \{i', j'\} = \emptyset$ ;
- **Minimal distance**  $\theta$  between paired nucleotides,  $\forall (i, j) \in S : j - i > \theta$ ;
- **No pseudoknot** allowed,  $\forall (i, j), (i', j') \in S, i < i' : (j' < j) \text{ or } (j < i')$ .

The **conformational space** associated with a sequence of length  $n$  is simply  $\mathcal{S}_n$ , the set of secondary structures over  $n$  nucleotides.

In typical *ab initio* RNA structure prediction problems, the input is a sequence of nucleotides  $w \in \{A, C, U, G\}^*$ . This space of secondary structure is then usually restricted to structures consisting of **canonical base pairs**  $\mathcal{B} := \{\{G, C\}, \{A, U\}, \{G, U\}\}$ . In other words, for any **valid secondary structure**  $S \in \mathcal{S}$  one has:

$$\forall (i, j) \in S : \{w_i, w_j\} \in \mathcal{B}.$$

The secondary structure can be drawn in a variety of ways, as illustrated by Figure 1.4, many of which being supported by our popular software VARNA, developed in collaboration with Kevin Darty and Alain Denise [44].

The stability of a secondary structure is assessed using a free-energy model. The most popular such model is the **Turner nearest-neighbor free-energy model** [185], which associates experimentally-determined **free-energy** contributions to structural motifs, called **loops** (see Figure 1.5). The energy of any given secondary structure is then additively defined, i.e. obtained by summing the contributions of the various loops appearing in the structure.

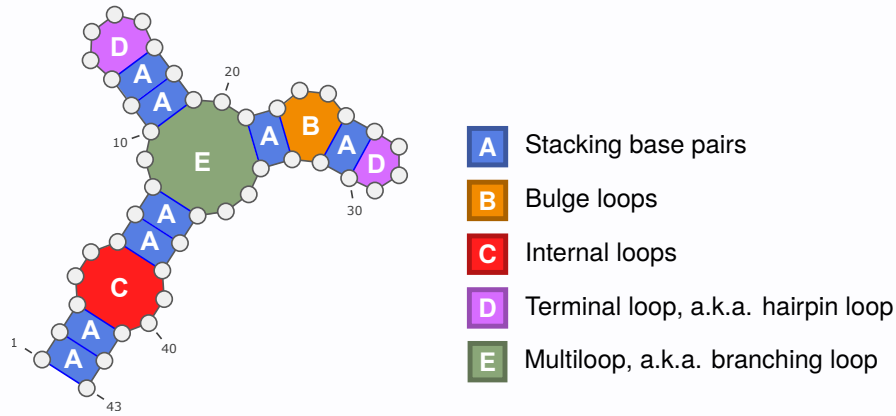


Figure 1.5: **Loop decomposition supporting the Turner nearest neighbor model [185].** Free-energies are associated to each loop type, precise topology and content in nucleotides, determined by, or extrapolations from, direct experimental measurements.

#### 1.2.4 RNA 2D structure prediction from thermodynamic principles

The secondary structure provides a convenient discretization of the conformational space, and allows a reformulation of several folding paradigms into combinatorial problems. For instance, predicting the most stable secondary structure (MFE paradigm) amounts to finding the secondary structure having minimal free-energy according to a chosen model. Since the overall and expected number of secondary structure of length  $n$  both grow exponentially on the length [194, 212], a proof being provided in Section 5.2, brute-force optimization is not realistically feasible.

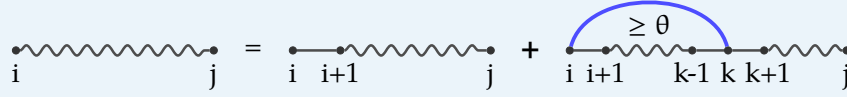
However, as illustrated in Example 1 below, the tree-like nature of secondary structure, and the independence of contributions within the energy model, make those optimizations amenable to a  $\Theta(n^3)$  **dynamic programming (DP)**, as initially shown by Nussinov and Jacobson [143] in a simplified energy model. The algorithm was later extended to capture loops and additional features of the Turner free-energy model by Zuker and Stiegler [213], leading to a  $\Theta(n^4)$  time algorithm (empirically running in  $\Theta(n^3)$  time). This algorithm was then extended by [204] to generate the exhaustive collection of secondary structures within a free-energy range  $\Delta$  of the MFE structure.

##### Example 1: RNA BP folding – Running example

A historical problem, first efficiently solved by Nussinov and Jacobson [143] is the prediction of the **MFE structure in a simple base pairing model**, i.e. having maximum number of pairs, with only nucleotides in  $\mathcal{B}$  being allowed to pair. Our goal is thus to compute some  $S^* \in \mathcal{S}_n$  such that

$$|S^*| = \max_{\substack{S \in \mathcal{S}_n \text{ such that} \\ \forall (i,j) \in S, \{w_i, w_j\} \in \mathcal{B}}} |S|.$$

A classic **dynamic programming scheme**, inspired by Waterman [194], considers each interval  $[i, j] \subset [1, n]$  and their associated optimal structure  $S_{i,j}^*$  for  $[i, j]$ :



Reasoning on the fate of the first position  $i$  within  $S_{i,j}^*$ , one only needs to consider three cases:

1.  $i + \theta \geq j$ , so no base pair can be formed, and we have  $S_{i,j}^* := \emptyset$ ;
2.  $i + \theta < j$  and  $i$  is left unpaired, so any base pair in  $S_{i,j}^*$  involves positions in  $[i + 1, j]$ , and we have  $S_{i,j}^* := S_{i+1,j}^*$ ;
3.  $i + \theta < j$  and  $i$  is paired to some  $k \in [i + 1, j]$  in  $S_{i,j}^*$ , subject to  $\{w_i, w_k\} \in \mathcal{B}$ . Beyond  $(i, k)$ , any base pairs in  $S_{i,j}^*$  involves positions in  $[i + 1, k - 1]$  in  $[k + 1, j]$  (but not both since pseudoknots are forbidden), so we get  $S_{i,j}^* := \{(i, k)\} \cup S_{i+1,k-1}^* \cup S_{k+1,j}^*$ .

This immediately suggests a **recurrence for the maximum number of base pairs**  $E_{i,j}$ , achieved by some secondary structure for the region  $[i, j] \in [1, n]$  of a given RNA  $w$ :

$$i + \theta \geq j : E_{i,j} := 0,$$

$$i + \theta < j : E_{i,j} := \max \begin{cases} E_{i+1,j} & (i \text{ unpaired}) \\ \max_{k=i+\theta+1}^j 1 + E_{i+1,k-1} + E_{k+1,j} & \text{if } \{w_i, w_k\} \in \mathcal{B} \text{ (i paired)} \end{cases}$$

Note that the computation of  $E_{i,j}$  only requires values  $E_{i',j'}$  such that  $||[i', j']|| < ||[i, j]||$ . A **dynamic programming algorithm** for computing  $|S^*| := E_{1,n}$  will consider intervals  $[i, j] \subseteq [1, n]$ , **ordered by increasing span**  $||[i, j]||$ , and compute  $E_{i,j}$  using precomputed  $E_{i',j'}$ , storing the result before proceeding to a (possibly) larger interval.

The **correctness** of the algorithm hinges on the fact that, while computing over  $E_{i,j}$ , all  $[i', j']$  such that  $||[i', j']|| < ||[i, j]||$  are already been processed, so that  $E_{i',j'}$  in the right hand side can be used. The number of different  $E_{i,j}$  terms grows in  $\Theta(n^2)$ . Moreover, each  $E_{i,j}$  can be computed in  $\mathcal{O}(n)$  arithmetic operations, assuming that all  $E_{i',j'}$  in the right-hand-side are accessible in  $\mathcal{O}(1)$  operations. The **overall complexities** of the algorithm are therefore in  $\Theta(n^3)/\Theta(n^2)$  time and space.

Once the maximum number of base pairs is computed, one can **backtrack** through these recursions, starting from  $[1, n]$ , by figuring out at each step (one of) the case(s) that contributed to the max. Concretely, one will define a recursive function **Backtrack** such that **Backtrack**( $i, j$ ) := Return $\emptyset$  when  $i + \theta \geq j$  and, for  $i + \theta < j$ :

$$\text{Backtrack}(i, j) := \begin{cases} \text{if } E_{i,j} = E_{i+1,j} : \text{return } \text{Backtrack}(i+1, j) \\ \text{if } E_{i,j} = 1 + E_{i+1,k-1} + E_{k+1,j}, k \in [i + \theta + 1, j] \text{ and } \{w_i, w_k\} \in \mathcal{B} : \\ \quad \text{return } \{(i, k)\} \cup \text{Backtrack}(i+1, k-1) \cup \text{Backtrack}(k+1, j) \end{cases}$$

Note that the two cases are not necessarily mutually-exclusive, i.e. multiple structures may achieve maximal number of base pairs, and an arbitrary-chosen optimal one will be returned.

A second breakthrough came with the advent of **comparative folding**, pioneered by the work of Sankoff [167], who automated the practices of manual modelers [134], and proposed an algorithm for the **simultaneous folding and alignment** of RNA. While associated with extreme computational demands ( $\Theta(n^6)$  for aligning two homologous sequences), its impressive gain in predictive power and reliability compared to *ab initio* techniques led it to be adapted within many subsequent methods, undergoing active



developments [128, 179, 199, 200].

A revolution came with the McCaskill [130] algorithm, which championed a transition towards the **Boltzmann ensemble** paradigm. Namely, McCaskill observed that the **partition function**

$$\mathcal{Z} = \sum_S e^{-E(w,S)/RT}$$

could be computed in essentially  $\Theta(n^3)$  time, through a simple change of algebra to the dynamic programming equations of [213]. Moreover, he showed that the same decomposition could be adapted into an instance of the inside/outside algorithm[10, 113], leading to a  $\Theta(n^4)$  algorithm for computing the **base-pair probabilities** within the Boltzmann ensemble. Such **ensemble properties** provide a notion of support for structures predicted in the MFE paradigm [127], but also allow to predict a structure that more adequately represents the ensemble than the MFE, such as the **Maximal Expected Accuracy** structure [55, 121].

The flexibility, and scope of applications, of ensemble methods was greatly extended by the contribution by Ding and Lawrence [51] of a **stochastic backtrack** algorithm, allowing to produce a random, Boltzmann distributed, secondary structure in time  $\Theta(n^2)$ . This enabled the implementation of **statistical estimators** for many features, including some that cannot be captured by dynamic programming. This also paved the way for a statistical estimation of the **dominant conformations** within the Boltzmann ensemble, using a combination of sampling and clustering [52], used by subsequent methods [177].

Finally, a strong emphasis has been recently put on the development of **integrative methods** that exploit the availability of (partial/noisy) experimental information. In the early 2000s, enzymatic and chemical probing data were integrated as **hard constraints** by Mathews et al. [129] within DP equations for RNA structure prediction. Following the development of quantitative experimental methods, such as the SHAPE technologies [176, 198], probing data are now incorporated as **soft constraints**, a.k.a. **pseudo-energies** within DP schemes for prediction. Such energy terms can be thought as *shifting* the Boltzmann ensemble towards areas of greater compatibility with the **reactivity profiles** produced by the probing experiments.



### 1.3 Outline

In this manuscript, I attempt to provide an unifying view on the algorithmic and theoretical concepts, used in a series of personal (frequently collaborative) contributions in Bioinformatics, Theoretical Computer Science and Discrete Mathematics. Those contributions are of different nature, and address a variety of objects of study, yet they share a consideration of ensembles of combinatorial objects, and are based on an interpretation of enumerations schemes as algorithmic principles.

In Chapter 2, I reformulate **ensemble dynamic programming** algorithms, contributed in the context of Bioinformatics within a unified framework. My contributions draw on a strong connection between (ensemble) dynamic programming and enumerative combinatorics, and put a strong emphasis on the design of dynamic programming schemes, whose productions are bijectively associated with an ensemble of objects of interest.

Chapter 3 focuses more specifically on algorithmic methods for RNA design, the algorithmic construction of new RNA sequences achieving a certain function, here through the adoption of a given structure. In this context, I revisit the classic inverse folding of RNA at under (essentially) a base pair maximization model, and show families for design can be approximated in an original sense. I also develop a vision inspired by random generation for the positive design problem, where one attempts to favor the affinity towards a given fold.

Chapter 4 describes rejection-based techniques and algorithms for a controlled random generation of combinatorial objects. Again, the initial focus is to present those techniques in an application-agnostic setting, later to describe some of their applications in Bioinformatics. Finally, I mention some analyses of the, arguably uninformative, redundancy within samples and the shortcomings of a rejection-based strategy to overcome it.

Finally, Chapter 5 summarizes a series of analyses focusing on the asymptotic properties of objects occurring in (RNA) bioinformatics. Those include RNA secondary structures, for which a careful application of analytic combinatorics principles allow to derive properties in the homopolymer model, but also other objects with connection to the design and analysis of algorithms.

## Chapter 2

# Ensemble dynamic programming: techniques and analyses

Following Bellman [14], a **dynamic programming algorithm** is usually stated as a system of recursive equations, relating the optimal value of an **objective function** over a certain argument, or problem, to a set of its values over other arguments, or sub-problems. Additionally, such an equation has induce an **acyclic computation**, meaning that the set of arguments that are used by following the recursive calls, can be totally ordered in a way that is consistent with the left-to-right transitions in the system. Such a definition overlooks an important aspect of a dynamic programming scheme: the semantics associated with the choice of one of the available left-to-right transitions.

In this manuscript, we adopt a perspective dually inspired by **enumerative combinatorics**, where the choice of a derivation provides partial information regarding the final solution. Such an enumerative perspective allows to define notions of unambiguity, correctness and completeness with respect to a given search space which, if fulfilled by a DP scheme, unlock a variety of algorithms to make statements regarding the **ensemble of solutions**. Such analyses go **beyond the optimization of an objective functions**, and attempt to answer selected questions involving the ensemble of candidate solutions:

- What is the support of an optimal solution?
- What is the number of near-optimal solutions?
- Are all near-optimal solutions similar? how diverse are they?
- What is the centroid solution, i.e. most similar to other near-optimal solutions?
- What are the average properties of near optimal solutions?
- Which distributions of properties are expected within near optimal solutions?

Such questions require the definition of a **probability distribution**, assigning probabilities to elements of the search space increasingly with their value for the objective function.

**Ensemble analyses**, whose first instances can probably be traced back to the origins of Natural Language Processing [10, 188], enjoy a great popularity in Bioinformatics. In particular, the early age of RNA bioinformatics saw seminal contributions from scientists with a strong combinatorial culture in Discrete Mathematics, Theoretical Physics and Computer Science, including Michael Waterman [194], Michael Zuker [212, 213], David Sankoff [167], Walter Fontana, Peter Stadler, Peter Schuster and Ivo Hofacker [74, 95, 96]. . .

The popularity of ensemble analyses in RNA Bioinformatics also stems directly from their origin in **statistical mechanics**, inspiring the seminal contribution of John McCaskill [130] who turned a popular DP scheme for energy minimization into an algorithm to compute the **partition function** of the Boltzmann ensemble. This central quantity allowed to derive expected properties of RNA at the **thermodynamic equilibrium**, starting with the **base pairing probabilities**, which have become central to modern methods for comparative folding of RNA [199, 200], based on a simultaneous folding and alignment of RNAs pioneered by David Sankoff [167].

In this context, my efforts have been focusing, on: i) the design of novel DP schemes amenable to ensemble analyses, associated with challenges of variable technicality; and ii) the development of new techniques to enable, or accelerate, ensemble analyses.

**Outline.** Section 2.1 describes a unifying framework for **ensemble dynamic programming**, used in Section 2.2 to reformulate classic and novel tools to perform an **exact computation of ensemble properties**. Section 2.3 focuses on **statistical sampling**, providing estimates for, possibly complex, Ensemble properties. Finally, Section 2.4 presents a selection of **algorithms and results** obtained in various areas of Bioinformatics using this general framework.

The following summarizes, and attempts to unify, contributions described within the following list of articles published in journals and/or presented at conferences.

#### Associated contributions

J. Waldispühl and Y. Ponty. An unbiased adaptive sampling algorithm for the exploration of RNA mutational landscapes under evolutionary pressure. *Journal of Computational Biology*, 18(11):1465–79, Nov. 2011

↑ J. Waldispühl and Y. Ponty. An unbiased adaptive sampling algorithm for the exploration of RNA mutational landscapes under evolutionary pressure. In **RECOMB 2011**, volume 6577 of *Lecture Notes in Computer Science*, pages 501–515, Vancouver, Canada, Mar. 2011. Springer Berlin / Heidelberg

Y. Ponty and C. Saule. A Combinatorial Framework for Designing (Pseudoknotted) RNA Algorithms. In **WABI 2011**, Saarbrücken, Germany, 2011

S. Sheikh, R. Backofen, and Y. Ponty. Impact Of The Energy Model On The Complexity Of RNA Folding With Pseudoknots. In **CPM 2012**, volume 7354 of *Combinatorial Pattern Matching*, pages 321–333, Helsinki, Finland, July 2012. Juha Kärkkäinen, Springer

- P. Rinaudo, Y. Ponty, D. Barth, and A. Denise. Tree decomposition and parameterized algorithms for RNA structure-sequence alignment including tertiary interactions and pseudoknots. In **WABI 2012**, tba, Ljubljana, Slovenia, Sept. 2012. University of Ljubljana
- E. Senter, S. Sheikh, I. Dotu, Y. Ponty, and P. Clote. Using the Fast Fourier Transform to Accelerate the Computational Search for RNA Conformational Switches. **PLoS ONE**, 7(12):e50506, Dec. 2012
- ↑ E. Senter, S. Sheikh, I. Dotu, Y. Ponty, and P. Clote. Using the Fast Fourier Transform to accelerate the computational search for RNA conformational switches (extended abstract). In **RECOMB 2013**, Beijing, China, Apr. 2013
- V. Reinharz, Y. Ponty, and J. Waldispühl. Using Structural and Evolutionary Information to Detect and Correct Pyrosequencing Errors in Noncoding RNAs. **Journal of Computational Biology**, 20(11):905–19, Nov. 2013
- ↑ V. Reinharz, Y. Ponty, and J. Waldispühl. A linear inside-outside algorithm for correcting sequencing errors in structured RNA sequences. In **RECOMB 2013**, Beijing, China, Apr. 2013
- C. Chauve, Y. Ponty, and J. P. P. Zanetti. Evolution of genes neighborhood within reconciled phylogenies: an ensemble approach. **BMC Bioinformatics**, 16(Suppl 19):S6, Dec. 2015
- ↑ C. Chauve, Y. Ponty, and J. P. P. Zanetti. Evolution of genes neighborhood within reconciled phylogenies: an ensemble approach. In **BSB 2014**, volume 8826 of *Advances in Bioinformatics and Computational Biology*, pages 49–56, Belo Horizonte, Brazil, Oct. 2014. Springer
- A. Rajaraman, C. Chauve, and Y. Ponty. Assessing the robustness of parsimonious predictions for gene neighborhoods from reconciled phylogenies. In **ISBRA 2015**, volume 9096, pages 260–271, Norfolk, Virginia, United States, June 2015
- E. Jacox, C. Chauve, G. J. Szöllösi, Y. Ponty, and C. Scornavacca. ecceTERA: Comprehensive gene tree-species tree reconciliation using parsimony. **Bioinformatics**, 32(13):2056–2058, July 2016
- V. Reinharz, Y. Ponty, and J. Waldispühl. Combining structure probing data on RNA mutants with evolutionary information reveals RNA-binding interfaces. **Nucleic Acids Research**, 44(11):e104 – e104, 2016
- W. Duchemin, Y. Anselmetti, M. Patterson, Y. Ponty, S. Bérard, C. Chauve, C. Scornavacca, V. Daubin, and E. Tannier. DeCoSTAR: Reconstructing the ancestral organization of genes or genomes using reconciled phylogenies. **Genome Biology and Evolution**, 9(5):1312–1319, 2017
- J. Deforges, S. De Breyne, M. Ameer, N. Ulryck, N. Chamond, A. Saaidi, Y. Ponty, T. Ohlmann, and B. Sargueil. Two ribosome recruitment sites direct multiple translation events within HIV1 Gag open reading frame. **Nucleic Acids Research**, 45(12):7382–7400, July 2017
- C. Chauve, J. Courtiel, and Y. Ponty. Counting, generating, analyzing and sampling tree alignments. **International Journal of Foundations of Computer Science**, 29(5):741–767, 2018
- ↑ C. Chauve, J. Courtiel, and Y. Ponty. Counting, generating and sampling tree alignments. In **ALCOB 2016**, volume 9702, pages 53–64, Trujillo, Spain, 2016. Springer
- S. Hammer, W. Wang, S. Will, and Y. Ponty. Fixed-parameter tractable sampling for RNA design with multiple target structures. **BMC Bioinformatics**, 20(1):209, Dec. 2019
- ↑ S. Hammer, Y. Ponty, W. Wang, and S. Will. Fixed-Parameter Tractable Sampling for RNA Design with Multiple Target Structures. In **RECOMB 2018**, Paris, France, 2018

## 2.1 A formal framework and basic algorithms

In order to refactor both classic and novel dynamic programming-based algorithms, we introduce a formal framework. Largely inspired from Juraj Michalik’s PhD [132], it can be seen as an operational version of the declarative proposal of Giegerich and Touzet [81], modeling dynamic-programming processes as inverse coupled term-rewriting systems [108]. Some of its features were previously introduced in collaboration with Cédric Saule [150], based on an oriented hypergraph framework pioneered by Finkelstein and Roytberg [66], independently pursued by Huang and Chiang in the context of natural language process-

ing [98]. It is also worth mentioning a substantial overlap with the book of Miklos [135] dedicated to the complexity of counting and sampling, of which our framework covers some of the easier (polynomially-solvable) cases.

### 2.1.1 Dynamic programming as a rewriting system

Denote by  $w$  an **instance** (e.g. sequence, tree, graph. . . ) of a combinatorial problem. An instance implicitly defines a discrete **universe**  $\mathcal{U}_w$ , of which the **search space**  $\Omega_w \subset \mathcal{U}_w$  of a combinatorial algorithm is a subset. Next, we need to describe how elements of the search space are decomposed, or conversely generated, by recursive constructors.

**Definition 2.1.1 (Constructor):** A **constructor** of **arity**  $k$  is a function  $\lambda : \mathcal{U}^k \rightarrow \mathcal{U}$  that returns/creates a novel element the search space  $\Omega$  from  $k$  elements of  $\Omega$ .

In other words, a constructor is a function that **assembles** a candidate solution from a collection of (smaller) candidate solutions to subproblems. Constructors with zero arity, i.e. constant functions, are called **atoms** and constitute base cases in the classic recursive exposition of dynamic programming. To mark this distinction between a constructor  $\lambda$  seen as a function, e.g. used to label the nodes of terms (see definition below), and its evaluation, we use the notation  $\lambda \rightsquigarrow v$ . Denote by  $\Lambda$  the **set of all constructors** for a given DP scheme.

Prior to any definition of a dynamic programming scheme, one needs to introduce a **state space**  $Q$ . Any state  $q \in Q$  represents a (sub)problem encountered while solving the recursive DP computation. The spirit of dynamic programming is to solve a given problem by solving a number of (smaller) problems, depending of the associated state and the instance. This dependency is materialized in our formalism by **derivations**, each decorated by a combinatorial **constructor**.

**Definition 2.1.2 (Derivation):** A **derivation** is a tuple

$$(q, (q_1, \dots, q_k), \lambda) \in Q \times Q^* \times \Lambda,$$

denoted as  $q \xrightarrow{\lambda} q_1, \dots, q_k$  such that:

- $q$  is the origin;
- $(q_1, \dots, q_k) \in Q^*$  is the production, i.e. an ordered list of states (a.k.a. subproblems) that have to be solved in order to solve  $q$ ;
- $\lambda \in \Lambda$  is a constructor of arity  $k$ .

We can now define a (combinatorial) dynamic programming scheme as a system of equations, coupled with a derivation system.

**Definition 2.1.3 (Dynamic Programming Scheme):** A dynamic programming scheme  $\Delta$  is a tuple  $(Q, q_w, \delta)$ , where:

- $Q$  is the state space;
- $q_w \in Q$  is the initial state;
- $\delta \subset Q \times Q^* \times \Lambda$  is an **acyclic**, i.e. non transitively self-referential, set of derivations.

The **acyclicity condition** forbids any state to (transitively) derive into itself, and is essential for algorithmic considerations. Note that, while the instance does not explicitly appear in this definition, its precise content is at the origin of the state space and lists of derivations. To illustrate this bundle of abstract definitions, let us reformulate our running example.

**Example 2: RNA BP folding – Formalized dynamic programming**

The DP scheme  $\Delta_{BP}$  for RNA 2D structure prediction problem, introduced in Section 1.2.3 can be formalized as follows: the **instance** is  $w \in \{A, C, U, G\}^*$ , a sequence of nucleotides; the **universe** is  $\mathcal{U} := \cup_{n' \leq n} \mathcal{S}_{n'}$ ; the **states** are the intervals  $Q := \{[i, j] \mid 1 \leq i \leq j \leq n\}$ ,  $n := |w|$  of the input sequence  $w$ ; and three types of **constructors** are sufficient to generate all structures:

- $\lambda_{\emptyset}^{[i,j]} \rightsquigarrow \emptyset \quad \rightarrow \text{Atom, returning the empty secondary structure;}$
- $\lambda_{\bullet}^{[i,j]}(S) \rightsquigarrow S \quad \rightarrow \text{Leaves position } i \text{ unpaired;}$
- $\lambda_k^{[i,j]}(S, S') \rightsquigarrow S \cup S' \cup \{\{i, k\}\} \quad \rightarrow \text{Adds base pair } \{i, k\} \text{ to two substructures } S \text{ and } S'.$

The **derivations** in  $\delta$  consist of:

- Terminal derivations:

$$\forall [i, j] \subset [1, n], j \leq i + \theta \rightarrow (i, j) \xrightarrow{\lambda_{\emptyset}^{[i,j]}} \varepsilon$$

Semantics: Position  $i$  left unpaired, no further processing required (sequence too short to support base pair), empty structure  $\lambda_{\varepsilon}$  returned;

- Unpaired derivations:

$$\forall [i, j] \subset [1, n], j \leq i + \theta : (i, j) \xrightarrow{\lambda_{\bullet}^{[i,j]}} (i + 1, j)$$

Semantics: Position  $i$  is left unpaired, requires processing of interval  $[i + 1, j]$ , (optimal) structure built over  $[i + 1, j]$ ;

- Paired derivations:

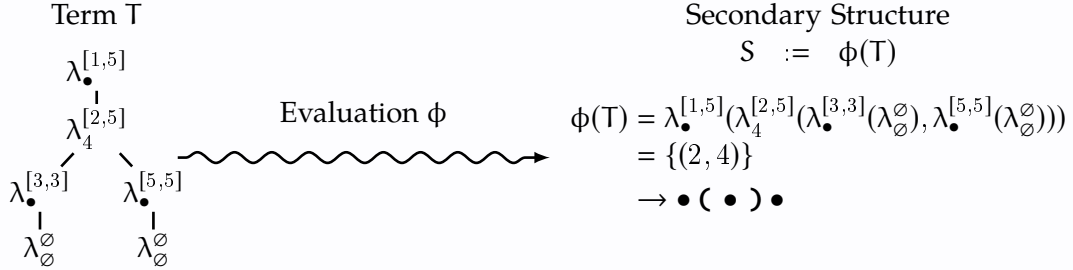
$$\forall [i, j] \subset [1, n], \forall k \in [i + \theta + 2, j], \{w_i, w_k\} \in \mathcal{B} : (i, j) \xrightarrow{\lambda_k^{[i,j]}} ((i + 1, k - 1), (k + 1, j))$$

Semantics: Position  $i$  paired with  $k$ , requires processing of intervals  $[i + 1, k - 1]$  and  $[k + 1, j]$ , (optimal) structure built from  $[i + 1, k - 1]$  and  $[k + 1, j]$ , augmented with base pair  $\{i, k\}$ ;

Derivation rules, in conjunction with constructors, define the search space explored (or, conversely, generated) by a DP scheme. To reason on the relationship between

constructors and elements of the search space, we make a subtle distinction between a **term**  $T$ , a tree-like hierarchy of constructors produced by a complete series of derivations, and its **evaluation**  $\phi(T)$  as an element of the search space.

For instance, using the constructors and semantics of Example 2, we have



### 2.1.2 Search space and suitability for ensemble applications

The terms space and search space produced by a given dynamic programming scheme can then be defined recursively as follows.

**Definition 2.1.4 (Terms of a DP scheme):** The **terms set**  $\mathcal{T}_{\Delta}$  generated by a dynamic programming scheme  $\Delta = (Q, q_w, \delta)$  is defined as  $\mathcal{T}_{\Delta} := \mathcal{T}_{q_w}$  where, for any  $q \in Q$ , one has:

$$\mathcal{T}_q = \bigcup_{q \xrightarrow{\lambda} q_1 \dots q_k \in \delta} \left\{ \lambda_{t_1 t_2 \dots t_k} \mid t_1 \in \mathcal{T}_{q_1}, t_2 \in \mathcal{T}_{q_2}, \dots, t_k \in \mathcal{T}_{q_k} \right\}.$$

Terms represent the syntactical structure of elements of the search space, which we now define.

**Definition 2.1.5 (Search space of a DP scheme):** The **search space**  $\Omega_{\Delta} := \Omega_{q_w}$  generated by a dynamic programming scheme can be similarly defined through

$$\begin{aligned}
 \Omega_q &= \{\phi(T)\}_{T \in \mathcal{T}_q} \\
 &= \bigcup_{q \xrightarrow{\lambda} q_1 \dots q_k \in \delta} \left\{ \lambda(s_1, \dots, s_k) \mid (s_1, \dots, s_k) \in \Omega_{q_1} \times \dots \times \Omega_{q_k} \right\}.
 \end{aligned}$$

Equipped with these notions, we can now define the properties of a dynamic programming scheme, that will connect it to an underlying reality.

**Definition 2.1.6 (Completeness and unambiguity of a DP scheme):** A DP scheme  $\Delta = (Q, q_w, \delta)$  is:

1. **Unambiguous** if and only if every element of the search space can be generated in only one way, i.e.  $\phi$  is bijective between  $\mathcal{T}_\Delta$  and  $\Omega_\Delta$ ;
2. **Complete** with respect to a targeted search space  $\Omega^*$  if and only if every element in  $\Omega^*$  is considered by  $\Delta$ , i.e. one has  $\Omega_\Delta = \Omega^*$ .

These two notions are crucial for ensemble applications of application. Indeed, in combination, they allow to use a given DP scheme to extract relevant properties of a preexisting search space.

Let us now distinguish elements within the search spaces, by introducing **scoring functions** that will map numerical values with each constructors and, in turn, to terms/elements of the search space

**Definition 2.1.7 (Additive scoring function):** An **additive scoring function**  $f : \Lambda \rightarrow \mathbb{R}$  associates a numerical value to each constructor, such that the **score**  $f(T)$  of a term  $T \in \mathcal{T}$  is defined as

$$f(T) := f(\phi(T)) := \sum_{\lambda \in T} f(\lambda).$$

An important property of a dynamic programming scheme lies is its ability to **emulate a given function** defined over its search space, by using a suitable scoring of its constructors/derivations. Such function could represent an **objective function** in the context of an optimization, or help induce a desired probability distribution over the search space.

**Definition 2.1.8 (Correctness of a DP scheme):** Let  $\Delta$  be a DP scheme, coupled with a scoring function  $f : \Lambda \rightarrow \mathbb{R}$ . A pair  $(\Delta, f)$  is **correct**, with respect to a given function  $F : \Omega_\Delta \rightarrow \mathbb{R}$ , if and only if  $F(\phi(T)) = f(T)$ ,  $\forall T \in \mathcal{T}$ .

By extension, we say that a DP scheme  $\Delta$  is **correct** with respect to a function  $F$  if and only if there exists a scoring function  $f$  such that  $(\Delta, f)$  is correct.

**Example 3: RNA BP folding – Unambiguity/completeness/correctness**

The **unambiguity** of  $\Delta_{BP}$  can be proven by considering two terms  $T$  and  $T'$ ,  $T \neq T'$ . Consider the first position from the root where constructors  $\lambda$  and  $\lambda'$ , such that  $\lambda \neq \lambda'$  are found in  $T$  and  $T'$  respectively. Since their paths to the root encounter the same constructors,  $\lambda$  and  $\lambda'$  are of the form  $\lambda_k^{[i,j]}$  or  $\lambda_\bullet^{[i,j]}$  (but not  $\lambda_\emptyset^{[i,j]}$ , since then both would be  $\Rightarrow \lambda = \lambda'$ ). Since two constructors irrevocably induce different partners for position  $i$ , their associated structures  $S := \phi(T)$  and  $S' := \phi(T')$  differ by at least one base



pair, and we have  $S \neq S'$ , implying the non-ambiguity of  $\Delta_{BP}$ .

The **completeness** of  $\Delta_{BP}$  requires that any structure  $S \in \mathcal{S}_n$  can be generated by the evaluation of some term in  $\mathcal{T}_q$ . This can be established by induction of the length  $n \geq \Theta + 2$  of the interval, assuming that, for all  $[i', j'] \in Q$  such that  $n' := j' - i' + 1 < n$ , one has  $\Omega_{n'} = \mathcal{S}_{n'}$ . Now, consider an interval  $[i, j]$ ,  $j - i + 1 = n$  and a structure  $S \in \mathcal{S}_n$ , and discuss the partner of  $i$  in  $S$ : if  $i$  is unpaired, then  $S = \bullet S'$  with  $|S'| < n$ , so  $S'$  is generated by  $T' \in \mathcal{T}_{[i+1, j]}$  and  $T := \lambda_{\bullet}^{[i, j]}(T') \in \mathcal{T}_{[i, j]}$  such that  $\phi(T) = S$ ; if  $i$  is paired to some  $k$ , then  $S = (S') S''$  with  $|S'| + |S''| < n$ , so  $S'$  and  $S''$  are generated by terms  $T' \in \mathcal{T}_{[i+1, k-1]}$  and  $T'' \in \mathcal{T}_{[k+1, j]}$  respectively, so  $T := \lambda_k^{[i, j]}(T', T'') \in \mathcal{T}_{[i, j]}$  such that  $\phi(T) = S$ .

The **correctness** of  $\Delta_{BP}$  requires that, for any secondary structure in the search space, the number of base-pairs is obtained by adding numerical values mapped to constructors. This is possible since, for any term  $T$  evaluated as  $\phi(T) = S$ , the number of base pairs in  $S$  coincides with the number of constructors of type  $\lambda_k^{[i, j]}$  in  $T$ , so the scoring function  $f : \lambda \rightarrow \mathbb{R}$  defined as

$$f(\lambda_{\bullet}^{[i, j]}) = f(\lambda_{\emptyset}^{[i, j]}) = 0 \quad \text{and} \quad f(\lambda_k^{[i, j]}) = \begin{cases} -1 & \text{if } \{w_i, w_k\} \in \mathcal{B} \text{ (valid base pair)} \\ +\infty & \text{otherwise} \end{cases}$$

**Remark 2.1.1:** Note that our assumption of scoring schemes that are additively-defined on constructors/transitions represents a limitations in expressivity in comparison to the more general **evaluation algebra** considered in **algebraic dynamic programming** and its extensions [80, 81, 169, 211]. However, general algebras do not allow a smooth transition from optimization to ensemble analyses, so we (slightly) limit the scope of our framework rather than burden our proofs and theorems. Moreover, the current framework captures, without any complexity overhead, every applications of dynamic programming known to this author in Bioinformatics.

### 2.1.3 Classic optimization

With this final notion of correctness being defined, we can finally turn to a more algorithmic dimension of dynamic programming, initially focusing on optimization problems, an historical focus of dynamic programming since its initial pioneering by Bellman [14].

#### **Problem 1 (DP-based optimization):**

**Input:** A dynamic programming scheme  $\Delta$  and a scoring function  $f$ , such that  $(\Delta, f)$  is correct with respect to an **objective function**  $F : \Omega_{\Delta} \rightarrow \mathbb{R}$

**Output:** Some element  $s^* \in \mathcal{S}_{\Delta}$  such that  $F(s^*) = \max_{s \in \mathcal{S}_{\Delta}} F(s)$

Unsurprisingly, this problem can be solved using dynamic programming, using an algorithm consisting of the following steps:

1. Matrix filling: For all state  $q \in Q$ , traversed in preorder, compute

$$m_q := \max_{q \xrightarrow{\lambda} q_1, \dots, q_k \in \delta} f(\lambda) + \sum_{i=1}^k m_{q_i} \quad (2.1)$$

2. Backtracking: Return  $\mathfrak{B}(q_w)$ , recursively defined for any  $q \in Q$  as:

$$\mathfrak{B}(q) := \lambda(\mathfrak{B}(q_1), \dots, \mathfrak{B}(q_k)), \text{ if } m_q = f(\lambda) + \sum_{i=1}^k m_{q_i}, q \xrightarrow{\lambda} q_1, \dots, q_k \in \delta$$

Note that the preorder in Step 1 always exists due to the acyclicity of derivations. The time complexity of this step is in  $\mathcal{O}(|Q| + \alpha^* \times |\delta|)$ ,  $\alpha^* := \max_{d \in \delta} \alpha(d)$  for  $\alpha(d)$  the arity of a derivation  $d$ , i.e. the number of states in its production. The memoization of computed values  $m_q$  requires  $\Theta(|Q|)$  memory. These complexities hold for the whole algorithm, since Step 2 involves recursing over at most  $|Q|$  states (due to acyclicity), and its complexity is typically orders of magnitude below the requirements of Step 1.

#### Example 4: RNA BP folding – Energy minimization

Let us illustrate Equation (2.1) in the context of  $\Delta_{BP}$ , the DP scheme of RNA BP folding. For base-pair maximization, the **objective function** is  $F(S) := |S|$ , achieved by a scoring function  $f$  such that  $f(\lambda_k^{[i,j]}) = \{1 \text{ if } (w_i, w_k) \in \mathcal{B}; -\infty \text{ otherwise}\}$ , and  $f(\lambda_{\bullet}^{[i,j]}) = f(\lambda_{\emptyset}^{[i,j]}) = 0$ . We get

$$m_{[i,j]} = \max \begin{cases} f(\lambda_{\emptyset}^{[i,j]}) & \text{if } i + \theta \geq j \quad \triangleright [i, j] \xrightarrow{\lambda_{\emptyset}^{[i,j]}} \emptyset \\ f(\lambda_{\bullet}^{[i,j]}) + m_{[i+1,j]} & \text{if } i + \theta < j \quad \triangleright [i, j] \xrightarrow{\lambda_{\bullet}^{[i,j]}} [i+1, j] \\ \max_k f(\lambda_k^{[i,j]}) + m_{[i+1,j]} + m_{[k+1,j]} & \text{if } i + \theta < j \quad \triangleright [i, j] \xrightarrow{\lambda_k^{[i,j]}} [i+1, k-1], [k+1, j] \end{cases}$$

$$= \max \begin{cases} 0 & \text{if } i + \theta \geq j \\ m_{[i+1,j]} & \text{if } i + \theta < j \\ \max_{k=i+\theta+1}^j 1 + m_{[i+1,j]} + m_{[k+1,j]} & \text{if } i + \theta < j \wedge (w_i, w_k) \in \mathcal{B} \end{cases}$$

in which one recognizes the classic DP equation reminded in Section 1.2.3.

## 2.2 Exact computation of Ensemble properties

In many applications of ensemble dynamic programming, one attempts to analyze a specific subset of the search space. Examples abound in RNA bioinformatics where an integer-valued feature function, additively defined with respect to the dynamic programming scheme, partition of the secondary structures with respect to their free-energy, base-pair distance to one or several references structure(s)...

### 2.2.1 Computing the partition function.

A ubiquitous quantity of interest is the **partition function**, whose definition requires integrating over the whole search space, and is used as a normalization term within ensemble studies. As observed by McCaskill [130], the optimization algorithm of any unambiguous DP scheme can be adapted to compute the partition function, through a simple algebraic substitution. Namely, it suffices to substitute  $(\min/\max, +) \rightarrow (+, \times)$ , coupled with a suitable exponentiation of energy contributions to compute the partition function from the MFE recursions. This observation has given rise to systematic studies decorrelating the DP scheme from its algebra [80, 97] with a specific focus on semi-ring algebras [135, 138].

#### Problem 2 (Partition function):

**Input:** An unambiguous DP scheme  $\Delta$  and a scoring  $f$ , such that  $(\Delta, f)$  is correct with respect to an **energy function**  $E : \Omega_\Delta \rightarrow \mathbb{R}$ ;  $\beta \in \mathbb{R}$  a constant

**Output:** The **partition function**  $Z_\Delta$  of  $\Omega_\Delta$ , defined as

$$Z_\Delta = \sum_{s \in \Omega_\Delta} e^{-\beta \cdot E(s)}$$

The above problem can be solved by returning  $Z_\Delta := Z_{q_w}$ , following the computation, for all state  $q \in Q$ , of

$$Z_q = \sum_{s \in \Omega_q} e^{-\beta \cdot E(s)}.$$

Those quantities can be computed recursively (in preorder), using

$$Z_q := \sum_{q \xrightarrow{\lambda} q_1, \dots, q_k \in \delta} e^{-\beta \cdot E(\lambda)} \times \prod_{i=1}^k Z_{q_i}. \quad (2.2)$$

The time complexity of this computation is in  $\mathcal{O}(|Q| + \alpha^* \times |\delta|)$ ,  $\alpha^*$  being the max arity of a constructor, and requires  $\Theta(|Q|)$  memory.

#### Example 5: RNA BP folding – Partition function

A reasonable **energy function** is defined as  $E(S) := -|S|$ , and implicitly used in Nussinov-Jacobson [143] scheme. It is achieved by an eponymous scoring function  $E$  such that  $E(\lambda_k^{[i,j]}) = \{-1 \text{ if } (w_i, w_k) \in \mathcal{B}; +\infty \text{ otherwise}\}$ , and  $E(\lambda_\bullet^{[i,j]}) = f(\lambda_\bullet^{[i,j]}) = 0$ . We get

$$Z_{[i,j]} := \sum \begin{cases} 1 & \text{if } i + \theta \geq j \\ Z_{[i+1,j]} & \text{if } i + \theta < j \\ \sum_{k=i+\theta+1}^j e^\beta \times Z_{[i+1,j]} \times Z_{[k+1,j]} & \text{if } i + \theta < j \wedge (w_i, w_k) \in \mathcal{B}. \end{cases}$$

The **validity** of the final result, i.e. the fact that  $Z_{[1,n]} = \sum_{S \in \mathcal{S}_n} e^{-\beta \cdot E(S)}$ , is a direct consequence of the unambiguity, completeness and correctness properties of  $\Delta_{BP}$ . The time and space complexities are in  $\Theta(n^3)$  and  $\Theta(n^2)$  respectively.

### 2.2.2 Probabilities in Boltzmann-Gibbs distributions (inside-outside)

In many relevant contexts, the elements of a search space  $\Omega$  can be assumed to follow a **Boltzmann-Gibbs distribution**, where the probability of any  $s \in \Omega$  is such that

$$\mathbb{P}(s) = \frac{e^{-\beta \cdot E(s)}}{\mathcal{Z}} \quad (2.3)$$

where  $E$  represents an energy score, and  $\beta$  a constant (analogous to a temperature), and  $\mathcal{Z}$  is the partition function. In such a context, the probabilities of individual elements induce average properties (e.g. base-pair probabilities) that are extremely relevant to ensemble analyses.

**Definition 2.2.1 (Unicity of a constructor):** A constructor  $\lambda \in \Lambda$  is **unique** within a DP scheme  $\Delta$  if it occurs at most once in each term  $T \in \mathcal{T}_\Delta$ .

Under the unicity condition, the probability of a constructor  $\lambda$  can be obtained by summing the individual probabilities of search space elements that result from its application (or, equivalently, the terms that contain  $\lambda$ ). Fortunately, this property can be computed efficiently using a suitable dynamic programming scheme, as stated below.

**Problem 3 (Boltzmann probability of constructor(s)):**

**Input:** Unambiguous DP scheme  $\Delta$  + scoring  $f$ , correct w.r.t. energy function  $E$ ;  $\beta \in \mathbb{R}$  a constant; and a **set**  $C \subset \Lambda$  of **unique constructors**

**Output:** The Boltzmann probabilities of constructors in  $C$ :

$$\forall \lambda \in C : \mathbb{P}(\lambda \in T) = \frac{\sum_{\substack{T \in \mathcal{T}_\Delta \\ \text{s.t. } \lambda \in T} e^{-\beta \cdot E(\Phi(T))}}{\mathcal{Z}_\Delta} \quad (2.4)$$

This problem, which generalizes the computation of production probabilities in probabilistic context-free grammars, is tackled by a variant of the **inside-outside algorithm** [10, 113]. The algorithm is based on the observation that, for any monitored constructor  $\lambda^*$ , any term  $T^* \in \mathcal{T}_{\lambda^*} := \{T \in \mathcal{T}_\Delta \mid \lambda^* \in T\}$  can be decomposed into:

1. a derivation  $d := q \xrightarrow{\lambda^*} q_1, \dots, q_k$  labeled by an occurrence of  $\lambda^*$ ;
2. an **outside** part, a partial term in  $T_q \in \mathcal{T}_\Delta$ , **truncated** on an occurrence of  $q$  (left underived). Let us denote by  $\mathfrak{D}_q$  the set of outside parts leading to  $q$ ;

3. several **inside** parts, i.e. individual **continuations** of the derivation process, starting from  $q_1, \dots, q_k$ . Denote as  $\mathfrak{I}_{q_1}, \dots, \mathfrak{I}_{q_k}$  the set of inside parts generated from  $q_1, \dots, q_k$  respectively;

Under the unicity condition, this decomposition is unambiguous. Moreover, the respective energy contributions of the three parts are independently contributing to the energy, and it follows that  $\mathcal{Z}_{\lambda^*}$ , the partition function restricted to  $\mathcal{T}_{\lambda^*}$ , obeys

$$\begin{aligned}
\mathcal{Z}_{\lambda^*} &:= \sum_{T \in \mathcal{T}_{\lambda^*}} e^{-\beta \cdot E(T)} \\
&= \sum_{q \xrightarrow{\lambda^*} q_1, \dots, q_k \in \delta} e^{-\beta \cdot E(\lambda^*)} \times \left( \sum_{T_q \in \mathfrak{D}_q} e^{-\beta \cdot E(T_q)} \right) \times \prod_{i=1}^k \left( \sum_{T_i \in \mathfrak{I}_{q_i}} e^{-\beta \cdot E(T_i)} \right) \\
&= \sum_{q \xrightarrow{\lambda^*} q_1, \dots, q_k \in \delta} e^{-\beta \cdot E(\lambda^*)} \times \mathcal{Y}_q \times \prod_{i=1}^k \mathcal{Z}_{q_i}
\end{aligned} \tag{2.5}$$

where  $\mathcal{Y}_q := \sum_{T_q \in \mathfrak{D}_q} e^{-\beta \cdot E(T_q)}$  is the **outside partition function**. Note that, when  $\mathcal{Y}_q$  is known, the above equation allows to simultaneously compute the partition functions  $\mathcal{Z}_{\lambda}$  for all monitored unique constructors, through a **single pass over the derivations**.

The computation of  $\mathcal{Y}_q$  itself can also be performed by **inverting the dynamic programming scheme**, going from a given state back to the root while allowing the further derivation of siblings found along the way. The outside partition function  $\mathcal{Y}_q$  of a node can be computed using dynamic programming using infix order, i.e. starting from the root  $q_w$  and processing the ancestors of a node before itself, through

$$\mathcal{Y}_q := \begin{cases} 1 & \text{if } q = q_w \text{ (root)} \\ \sum_{\substack{q_p \xrightarrow{\lambda} q \in \delta \\ \text{s.t. } q \in \mathfrak{q}}} e^{-\beta \cdot E(\lambda)} \times \mathcal{Y}_{q_p} \times \prod_{\substack{q' \in \mathfrak{q} \\ q' \neq q}}^k \mathcal{Z}_{q'} & \text{otherwise.} \end{cases} \tag{2.6}$$

Overall, the **inside-outside algorithm** solving Problem 3 can be stated as:

- Using Equation (2.2), compute the inside partition function  $\mathcal{Z}_q$  for all state  $q \in Q$  in preorder;  $\rightarrow \mathcal{O}(|Q| + \alpha^* \times |\delta|)$  time
- Using Equation (2.6), compute the outside partition function  $\mathcal{Y}_q$  for all state  $q \in Q$  in infix order;  $\rightarrow \mathcal{O}(|Q| + \alpha^* \times |\delta|)$  time
- Iterate over derivations  $q \xrightarrow{\lambda} q_1 \dots q_k \in \delta$  to compute  $(\mathcal{Z}_{\lambda})_{\lambda \in C}$ , initially set to 0. If  $\lambda \in C$ , update  $\mathcal{Z}_{\lambda} \leftarrow \mathcal{Z}_{\lambda} + e^{-\beta \cdot E(\lambda^*)} \mathcal{Y}_q \prod_{i=1}^k \mathcal{Z}_{q_i}$ ;  $\rightarrow \mathcal{O}(\alpha^* \times |\delta| + |C|)$  time
- Finally, the algorithm returns  $\mathbb{P}(\lambda \in T) := \mathcal{Z}_{\lambda} / \mathcal{Z}_{q_w}, \forall \lambda \in C$ .

The algorithm runs in time  $\mathcal{O}(|Q| + |C| + \alpha^* \times |\delta|)$ ,  $\alpha^*$  being the maximum arity of a constructor, and requires storage for  $\mathcal{O}(|Q| + |C|)$  numbers.

#### Example 6: RNA BP folding – BP probabilities

In the context of RNA, the inside/outside algorithm can be used to compute **base-pair probabilities**, as done by McCaskill [130]. Inside contributions/partition functions are computed as detailed in Example 5, using  $E(\lambda_k^{[i,j]}) = \{-1 \text{ if } (w_i, w_k) \in \mathcal{B}; +\infty \text{ otherwise}\}$  and  $E(\lambda_{\bullet}^{[i,j]}) = f(\lambda_{\emptyset}^{[i,j]}) = 0$ . The monitored constructors are all  $\lambda_k^{[i,j]}$ , each **unique** as a position  $i$  cannot be assigned twice.

The outside contributions  $Y_q$  are then computed through a specialization of Equation (2.6):

$$Y_{[i,j]} := \sum \begin{cases} 1 & \text{if } [i,j] = [1,n] \\ Y_{[i-1,j]} & \text{if } i-1 \geq 1 \\ \sum_{i'=1}^{i-\Theta-1} e^{\beta} \times Y_{[i',j]} \times Z_{[i'+1,i-2]} & \text{if } (w_{i'}, w_{i-1}) \in \mathcal{B}; \\ \sum_{j'=j+1}^n e^{\beta} \times Y_{[i,j']} \times Z_{[i+1,j']} & \text{if } (w_{i-1}, w_{j+1}) \in \mathcal{B}. \end{cases}$$

We finally obtain the probabilities of constructors through a specialization of Equation (2.5)

$$\mathbb{P}(\lambda_k^{[i,j]} \in T) := \begin{cases} \frac{e^{\beta} \times Y_{[i,j]} \times Z_{[i+1,k-1]} \times Z_{[k+1,j]}}{Z_{[1,n]}} & \text{if } (w_i, w_k) \in \mathcal{B} \\ 0 & \text{otherwise} \end{cases}$$

Summing over all values of  $j$ , we get the probability of a base-pairs  $(i, k)$ ,  $k - i > \theta$

$$\mathbb{P}((i, k) \in S) := \sum_{j \geq k}^n \mathbb{P}(\lambda_k^{[i,j]} \in T)$$

The probabilities of all base pairs can then be computed the inside/outside, in  $\Theta(n^3)$  time, and  $\Theta(n^2)$  space by computing the probabilities of constructors *on the fly* within the above sum.

**Remark 2.2.1 (Beyond unique features):** Remark that, in the case where  $\lambda$  is not unique, i.e. it occurs more than once in a term, the output of the above algorithm is no longer the probability of occurrence, but the **expected number of occurrences** of  $\lambda$  in a random, Boltzmann-distributed, term. This quantity may be of interest, for instance when trying to assess expected properties of the Boltzmann ensembles, since it allows the simultaneous computation of many expected features in a single pass.

### 2.2.3 Ensemble centroid and Maximum Expected Accuracy (MEA)

The probabilities computed in Problem 3 allow to assess a notion of **support** for the individual **features** (e.g. base pairs, helices...) of a solution within the Boltzmann-Gibbs distribution. Thus, they can be used to assess how **representative** a given solution is of the Boltzmann-Gibbs ensemble. In particular, when  $s^*$  is the minimum free-energy solution, we know that  $s^*$  achieves maximal probability in the Boltzmann-Gibbs distribution.

However, in absolute terms, the probability of  $s^*$  may be (and usually is) abysmally small, and does not allow *in itself* to distinguish between two very different situations:

1. The solution  $s^*$  is surrounded by a family of **similar suboptimal solutions**  $\{s'_1, s'_2 \dots\}$ , having very similar features (e.g.  $|s^*, s'_i| \leq \eta$  for some notion of distance), **overtak-**

ing the probability distribution ( $\mathbb{P}(s^*) + \sum_i \mathbb{P}(s'_i) = 1 - \epsilon$ );

2. The solution  $s^*$ , even supplemented by similar suboptimals, is **highly dominated** by **dissimilar** solutions in the Boltzmann ensemble ( $\mathbb{P}(s^*) + \sum_i \mathbb{P}(s'_i) = \epsilon$ ).

A possible way to distinguish between those two worlds, consists in computing an **expected distance** of  $s^*$  to a random solution in the ensemble.

**Definition 2.2.2 (Weighted distance between solutions):** Given two solutions  $s, s' \in \Omega$  resulting from the applications of sets of unique constructors  $C = \{\lambda_1, \dots, \lambda_k\}$  and  $C' = \{\lambda'_1, \dots, \lambda'_{k'}\}$  respectively. Then the **distance**  $|s, s'|$  between  $s$  and  $s'$  is defined as

$$|s, s'|_\mu = \sum_{\lambda \in \Lambda} \pi_\lambda \times (\mathbf{1}_{\lambda \in C} - \mathbf{1}_{\lambda \in C'})^2$$

where  $\pi : \Lambda \rightarrow \mathbb{R}^+$  is a collection of weights.

Since constructors represent atomic operations that build a given element of the search space (e.g. adding a base pair, declaring a nucleotide unpaired. . . ), this notion of distance represents a natural way to represent popular distance metrics.

Equipped with a notion of distance, we can now define the **centroid** of the Boltzmann-Gibbs ensemble [55, 87] as its most central element, i.e. the solution having **minimum expected distance** to a, Boltzmann-distributed, elements of the search space.

**Problem 4 (Centroid solution):**

**Input:** Unambiguous DP scheme  $\Delta$  + scoring  $f$ , correct w.r.t. function  $E$ ;  $\beta \in \mathbb{R}$  a constant; and a weighted distance  $|\star, \star|_\mu$

**Output:** Solution  $s^* \in \Omega_\Delta$  minimizing the **expected distance to the ensemble**:

$$s^* = \operatorname{argmin}_{s \in \Omega_\Delta} \sum_{s' \in \Omega_\Delta} \mathbb{P}(s') \times |s, s'|_\mu \quad (2.7)$$

Fortunately, the expected distance to the ensemble of any given candidate solution  $s \in \Omega$  can be reexpressed as a simple sum over the Boltzmann probabilities of constructors.

Indeed, one has

$$\begin{aligned}
\sum_{s' \in \Omega_\Delta} \mathbb{P}(s') \times |s, s'|_\mu &= \sum_{s' \in \Omega_\Delta} \mathbb{P}(s') \times \sum_{\lambda \in \Lambda} \pi_\lambda \times (\mathbf{1}_{\lambda \in s} - \mathbf{1}_{\lambda \in s'})^2 \\
&= \sum_{\substack{\lambda \in \Lambda \\ \lambda \in s}} \pi_\lambda \times \mathbb{P}(\lambda \notin T) + \sum_{\substack{\lambda \in \Lambda \\ \lambda \notin s}} \pi_\lambda \times \mathbb{P}(\lambda \in T) \\
&= \sum_{\substack{\lambda \in \Lambda \\ \lambda \in s}} \pi_\lambda \times (\mathbb{P}(\lambda \notin T) - \mathbb{P}(\lambda \in T)) + \sum_{\lambda \in \Lambda} \pi_\lambda \times \mathbb{P}(\lambda \in T) \quad (2.8)
\end{aligned}$$

Since the rightmost sum no longer depends on  $s$ , finding the solution that minimizes the expected distance is equivalent to finding a solution that optimizes the leftmost sum. Following this observation, one can solve Problem 4 by executing the following algorithm:

1. Compute the Boltzmann probabilities of constructors used by the weighted distance, i.e. solve Problem 3 with  $C := \{\lambda \in \Lambda \mid \pi_\lambda \neq 0\}$ ;
2. Find  $s \in \Omega_\Delta$  that minimizes the leftmost term of Equation 2.8, i.e. solve Problem 1, maximizing the objective function  $F : \Lambda \rightarrow \mathbb{R}$  such that

$$F(\lambda) := \pi_\lambda \times (\mathbb{P}(\lambda \in T) - \mathbb{P}(\lambda \notin T)) = \pi_\lambda \times (2 \mathbb{P}(\lambda \in T) - 1) \quad (2.9)$$

#### Example 7: RNA BP folding – Centroid computation

We consider the classic **base-pair distance** as the distance to be minimized, and accordingly replace all constructors  $\lambda_k^{[i,j]}$  with new **simplified constructors**  $\lambda_{(i,k)}$  and  $\lambda_{(i)}$  which respectively represent occurrences of a base pair  $(i, k)$  and an unpaired position  $i$ , irrespectively of their context  $[i, j]$  of creation (since the context of a base pair should not contribute to the distance). We set the weight of all constructors to 0 except for  $\pi(\lambda_{(i,k)}) = 1$  in the distance definition, and compute the base-pair probabilities  $p_{i,j} := \mathbb{P}((i, j) \in S)$  as shown in Section 2.2.2.

Then, we solve Problem 1 in this new setting, i.e. compute the recurrence

$$c_{[i,j]} = \max \begin{cases} 0 & \text{if } i + \theta \geq j \\ c_{[i+1,j]} & \text{if } i + \theta < j \\ \max_{k=i+\theta+1}^j (2 p_{i,k} - 1) + c_{[i+1,j]} + c_{[k+1,j]} & \text{if } i + \theta < j \wedge (w_i, w_k) \in \mathcal{B} \end{cases}$$

to get the least distance of a structure to the ensemble (up to a constant, i.e. the rightmost term in (2.9))

A classic backtrack allows to recover the **centroid secondary structure**.

A **Maximum Expected Accuracy (MEA)** solution [121] can be obtained in a very similar fashion by simplifying the objective function of Equation (2.9) to  $F(\lambda) := \pi_\lambda \times \mathbb{P}(\lambda \in T)$ , with  $\pi_{\lambda_{BP}} := 2\gamma$  for base-pairing constructors, and  $\pi_{\lambda_{Unp.}} := 1$  for unpaired constructors.

### 2.2.4 General moments of additive scores (150)

Given an additive scoring function  $F$ , it is a natural question to ask for the induced distribution of  $F$  under a Boltzmann Gibbs distribution. Since most such distributions are typically Gaussian, a first task is to compute the expected value  $\mu_\Delta(F)$  and variance



$\sigma_\Delta(F)$  of  $F$ , respectively defined as  $\mu_\Delta(F) := \mathbb{E}(F(T))$  and

$$\sigma_\Delta(F) := \sqrt{\sum_{T \in \mathcal{T}_\Delta} \mathbb{P}(T) \cdot (F(T) - \mu_\Delta(F))^2} = \sqrt{\mathbb{E}(F(T)^2) - \mathbb{E}(F(T))^2}.$$

In order to capture characteristics of more general distributions, one may consider the **moments** of the distribution, defined as  $\mathbb{E}(F(T)), \mathbb{E}(F(T)^2), \dots$ , previously considered by Miklos, Meyer and Borbala [136] specifically for the free-energy. The **cross-moments**  $\mathbb{E}(F_1(T)^{n_1} \cdot F_2(T)^{n_2} \dots)$  of multiple functions are also of potential interest, as their computation allows to derive the **Pearson correlation**  $\rho_\Delta(F_1, F_2)$  of two functions  $F_1$  and  $F_2$  through

$$\begin{aligned} \rho_\Delta(F_1, F_2) &= \frac{\mathbb{E}((F_1(T) - \mu_\Delta(F_1)) \times (F_2(T) - \mu_\Delta(F_2)))}{\sigma_\Delta(F_1) \times \sigma_\Delta(F_2)} \\ &= \frac{\mathbb{E}(F_1(T) \times F_2(T)) - \mathbb{E}(F_1(T)) \times \mathbb{E}(F_2(T))}{\sqrt{\mathbb{E}(F_1(T)^2) - \mathbb{E}(F_1(T))^2} \times \sqrt{\mathbb{E}(F_2(T)^2) - \mathbb{E}(F_2(T))^2}} \end{aligned}$$

so the correlation can be computed from the evaluation of  $\mathbb{E}(F_1(T)^{n_1} \times F_2(T)^{n_2})$  for all values of  $(n_1, n_2) \in \{(1, 0), (0, 1), (2, 0), (0, 2), (1, 1)\}$ .

In a collaboration with Cédric Saule [150], we have considered the computation of general cross-moments within dynamic programming schemes.

**Problem 5 ((Cross) moments of a DP scheme):**

**Input:** Unambiguous DP scheme  $\Delta$  + scoring  $f$ , correct w.r.t. function  $E$ ;  $\beta \in \mathbb{R}$ ; and **scoring functions**  $(F_1, \dots, F_p)$  with associated **degrees**  $(\gamma_1^*, \dots, \gamma_p^*)$

**Output:** The (cross) moment  $\gamma_1^*, \dots, \gamma_p^*$  for  $F_1, \dots, F_p$  :

$$m^{\{\gamma_1^* \dots \gamma_p^*\}} := \mathbb{E}(F_1(T)^{\gamma_1^*} \times \dots \times F_p(T)^{\gamma_p^*}) = \sum_{T \in \mathcal{T}_\Delta} \mathbb{P}(T) \prod_{i=1}^p F_i(T)^{\gamma_i^*} \quad (2.10)$$

A reexpression of our result [150] states that  $m^{\gamma_1, \dots, \gamma_p}$  can be computed as

$$m_q^{\{\gamma_1 \dots \gamma_p\}} := \sum_{q \xrightarrow{\lambda} q_1 \dots q_k \in \delta} e^{-\beta \cdot E(\lambda)} \sum_{\substack{\lambda_1 + \tau_{1,1} \dots \tau_{1,k} = \gamma_1 \\ \dots \\ \lambda_p + \tau_{p,1} \dots \tau_{p,k} = \gamma_p}} \prod_{i=1}^p F_i(\lambda)^{\lambda_i} \binom{\gamma_i}{\tau_{i,1} \dots \tau_{i,k}} \prod_{j=1}^k m_{q_j}^{\{\tau_{1,j} \dots \tau_{p,j}\}}. \quad (2.11)$$

This expression, whose underlying intuition is probably hard to decipher at first sight, is in fact strongly inspired by the **partial pointing operator** in enumerative combinatorics [50, 69]. Its main underlying idea is to modify the DP scheme  $\Delta$ , introducing of a **controlled ambiguity**, into a DP scheme  $\Delta^{\{\gamma_1, \dots, \gamma_p\}}$  designed to generate pointed / weighted versions of the original terms.

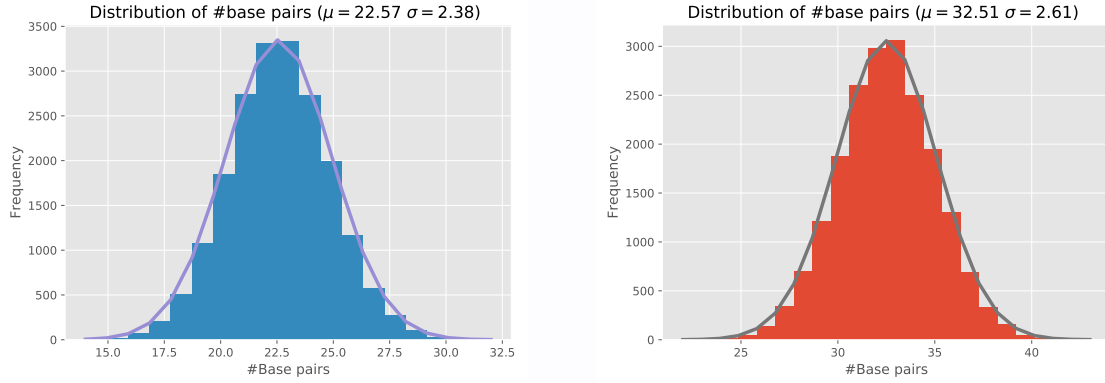


Figure 2.1: **Distribution of base pairs induced by a uniform (left) and Boltzmann distribution.** Empirical distributions (histograms) measured on 20 000 stochastic backtracks, approximated by a Normal distribution with exact means and variances, computed from the first two moments. Boltzmann Sampling performed at 37°C with  $E(S) = -|S|$  (right) for a toy RNA  $w := (ACGU)^{25}$ .

By generating each term of  $\Delta$  with suitable multiplicity, and weighting them with a carefully chosen monomial in  $F_1, F_2, \dots, \Delta^{\{\gamma_1, \dots, \gamma_p\}}$  we ensure that the **multinomial formula** applies and, for all  $T \in \Delta$ , the overall weight contributed by the equivalence class  $\{T' \in \mathcal{T}_{\Delta^{\{\gamma_1, \dots, \gamma_p\}}} \mid S' \Leftrightarrow S\}$  simplifies into  $F_1^{\gamma_1}(T) \times \dots \times F_p^{\gamma_p}(T)$ .

An algorithm for Problem 5 then consists in using Equation (2.11) to compute  $m_q^{\gamma_1 \dots \gamma_p}$  for all  $(\gamma_1 \dots \gamma_p) \leq (\gamma_1^* \dots \gamma_p^*)$  in increasing lexicographic order and all  $q \in Q$  in preorder for each degree vector, and return  $m_{q_w}^{\{\gamma_1^* \dots \gamma_p^*\}}$ . Its time complexity is loosely bounded by  $\mathcal{O}(|\delta| \times \prod_{i=1}^p \gamma_i^{*1+\alpha^*} \times P(k, p))$  where  $P$  is a polynomial of bounded degree, and the space complexity is in  $\Theta(|Q| \times \prod_{i=1}^p \gamma_i^*)$ . In practice, degree vectors are typically small values, so the complexity of the algorithm is equivalent, up to a constant factor, to the computation of the partition function.

#### Example 8: RNA BP folding – Distribution of base pairs

The homopolymer model predicts that the distribution of base-pairs in random, Boltzmann-distributed secondary structure asymptotically follows a normal distribution, and is thus characterized by its first two moments  $m^{\{1\}}/\mathcal{Z}$  and  $m^{\{2\}}/\mathcal{Z}$ . Those can be computed by introducing a scoring function  $F$ , defined as  $F(\lambda_{\bullet}^{[i,j]}) = F(\lambda_{\emptyset}^{[i,j]}) = 0$  and  $F(\lambda_k^{[i,j]}) = 1$ , leading to

$$m_{[i,j]}^{\{1\}} = \begin{cases} F(\lambda_{\emptyset}^{[i,j]}) & \text{if } i + \theta \geq 1 \\ F(\lambda_{\bullet}^{[i,j]}) \times m_{[i,j]}^{\{0\}} + m_{[i,j]}^{\{1\}} & \text{if } i + \theta < 1 \\ \sum_{k=i+\theta+1}^j F(\lambda_k^{[i,j]}) \times e^{\beta} \times m_{[i+1,j]}^{\{0\}} \times m_{[k+1,j]}^{\{0\}} & \text{if } i + \theta < j \wedge (w_i, w_k) \in \mathcal{B} \\ \sum_{k=i+\theta+1}^j e^{\beta} \times m_{[i+1,j]}^{\{1\}} \times m_{[k+1,j]}^{\{0\}} & \text{if } i + \theta < j \wedge (w_i, w_k) \in \mathcal{B} \\ \sum_{k=i+\theta+1}^j e^{\beta} \times m_{[i+1,j]}^{\{0\}} \times m_{[k+1,j]}^{\{1\}} & \text{if } i + \theta < j \wedge (w_i, w_k) \in \mathcal{B} \end{cases}$$

with  $m_{[i,j]}^{\{0\}} := \mathcal{Z}_{[i,j]}$ , leading to the exact value for the expected number of base pairs  $\mu_{BP} :=$

$m_{[1,n]}^{\{1\}} / \mathcal{Z}_{[1,n]}$ . A similar equation can be derived for the second moment

$$m_{[i,j]}^2 = \begin{cases} 0 & \text{if } i + \theta \geq 1 \\ m_{[i,j]}^2 & \text{if } i + \theta < 1 \\ \sum_{k=i+\theta+1}^j e^\beta \times F(\lambda_k^{[i,j]})^2 \times m_{[i+1,j]}^0 \times m_{[k+1,j]}^0 & \text{if } i + \theta < j \wedge (w_i, w_k) \in \mathcal{B} \\ \sum_{k=i+\theta+1}^j e^\beta \times 2 \times F(\lambda_k^{[i,j]})^1 \times m_{[i+1,j]}^1 \times m_{[k+1,j]}^0 & \text{if } i + \theta < j \wedge (w_i, w_k) \in \mathcal{B} \\ \sum_{k=i+\theta+1}^j e^\beta \times 2 \times F(\lambda_k^{[i,j]})^1 \times m_{[i+1,j]}^0 \times m_{[k+1,j]}^1 & \text{if } i + \theta < j \wedge (w_i, w_k) \in \mathcal{B} \\ \sum_{k=i+\theta+1}^j e^\beta \times 2 \times m_{[i+1,j]}^1 \times m_{[k+1,j]}^1 & \text{if } i + \theta < j \wedge (w_i, w_k) \in \mathcal{B} \\ \sum_{k=i+\theta+1}^j e^\beta \times m_{[i+1,j]}^2 \times m_{[k+1,j]}^0 & \text{if } i + \theta < j \wedge (w_i, w_k) \in \mathcal{B} \\ \sum_{k=i+\theta+1}^j e^\beta \times m_{[i+1,j]}^0 \times m_{[k+1,j]}^2 & \text{if } i + \theta < j \wedge (w_i, w_k) \in \mathcal{B} \end{cases}$$

from which we get the standard deviation  $\sigma_{BP} := \sqrt{m_{[1,n]}^2 / \mathcal{Z}_{[1,n]} - \mu_{BP}^2}$

### 2.2.5 Classified DP with the Discrete Fourier Transform (DFT) (173, 174)

In some situations, precise aspects of the distribution are of interest, and cannot be easily captured by summary statistics. For instance, Freyhult *et al* [75] partition the Boltzmann Ensemble according to the distance to a reference secondary structure. Considering the distribution of distance, they observe bimodal (an even trimodal) distributions that they interpret as potential evidences for the presence of multistable RNAs. Other examples in RNA bioinformatics include the classification of sequences/structures with respect to their Hamming distance to a wild-type sequence, to assess the mutational robustness of concrete RNAs [191], the computation of the density of states, i.e. the distribution of free-energies within an RNA [43], or the projection of energy landscapes with respect to the unfolded and native states [119].

Those instances can be expressed as a specialized version of **classified dynamic programming**, where additional parameters are used to partition the search space.

#### Problem 6 (Classified partition function):

**Input:** Unambiguous DP scheme  $\Delta$  + scoring  $f$ , correct w.r.t. function  $E$ ;  $\beta \in \mathbb{R}$ ; and an additive **scoring functions**  $F : \Lambda \rightarrow \mathbb{N}^+$  with  $V := \max_{T \in \mathcal{T}_\Delta} F(T)$

**Output:** The **classified partition function**  $\mathcal{Z}^{[v]}$ ,  $0 \leq v \leq V$ , defined as

$$\mathcal{Z}^{[v]} := \mathbb{P}(F(T) = v) \times \mathcal{Z}_\Delta = \sum_{\substack{T \in \mathcal{T} \\ \text{s.t. } F(T)=v}} e^{-\beta \cdot E(T)}$$

A classic approach, used in many works including the above references, would consist in adapting the dynamic programming scheme through **explicit convolution products**,

leading to the following recurrence:

$$\mathcal{Z}_q^{[v]} := \sum_{q \xrightarrow{\lambda} q_1, \dots, q_k \in \delta} \sum_{v_1 + \dots + v_k + F(\lambda) = v} e^{-\beta \cdot E(\lambda)} \prod_{i=1}^k \mathcal{Z}_{q_i}^{[v_i]}. \quad (2.12)$$

While conceptually simple, this approach induces extreme computational demands, with a time complexity in  $\Theta(|\delta| \cdot V^{\alpha^*})$ ,  $\alpha^*$  being the max arity of a derivation, and using  $\Theta(|Q| \cdot V)$  memory. For instance, in our running DP scheme example for RNA secondary structures (see Section 1.2.3), classifying according to the number of base pairs and applying this strategy induces an algorithm in  $\Theta(n^5)$  time and  $\Theta(n^3)$  memory complexities.

In collaboration with Evan Senter, Ivan Dotu, Peter Clote and Saad Sheikh, we have proposed an alternative strategy based on the Discrete Fourier Transform (DFT) to avoid costly convolution products [173, 174], following observations made in an earlier collaborative work with Jérôme Waldspühl [192, 193].

Its core idea is to consider a generalized version of the partition function, seen as a polynomial in a formal variable  $x$ , which tracks the increments of the scoring function, such that

$$\mathcal{Z}(x) = \sum_{T \in \mathcal{T}_\Delta} e^{-\beta \cdot E(T)} x^{F(T)} = \sum_{v=0}^V \mathcal{Z}^{[v]} x^v.$$

Such a polynomial can be evaluated at any point without having to previously determine its coefficients, using the following DP equation

$$\mathcal{Z}_q(x) := \sum_{q \xrightarrow{\lambda} q_1, \dots, q_k \in \delta} e^{-\beta \cdot E(\lambda)} x^{F(\lambda)} \prod_{i=1}^k \mathcal{Z}_{q_i}(x). \quad (2.13)$$

From a well-chosen set of evaluations of  $\mathcal{Z}(x) := \mathcal{Z}_{q_w}(x)$ , it is then possible to use **polynomial interpolation** to recover the coefficients of the polynomial, i.e. the classified partition functions  $\mathcal{Z}^{[v]}$ .

For the sake of illustration, one may perform a preliminary evaluation of  $\mathcal{Z}$  at  $(1 + V)$  distinct points  $x_0, x_1, \dots, x_V$ , and then use **Gaussian elimination** to solve the linear system

$$\begin{bmatrix} x_0^V & \dots & x_0^1 & 1 \\ x_1^V & \dots & x_1^1 & 1 \\ \vdots & & \vdots & \vdots \\ x_V^V & \dots & x_V^1 & 1 \end{bmatrix} \times \begin{bmatrix} \mathcal{Z}^{[0]} \\ \mathcal{Z}^{[1]} \\ \vdots \\ \mathcal{Z}^{[V]} \end{bmatrix} = \begin{bmatrix} \mathcal{Z}(x_0) \\ \mathcal{Z}(x_1) \\ \vdots \\ \mathcal{Z}(x_V) \end{bmatrix}. \quad (2.14)$$

in its unknowns  $\mathcal{Z}^{[0]}, \mathcal{Z}^{[1]}, \dots$  thus solving Problem 6 in  $\Theta(V|\delta|\alpha^* + V^3)$  time. However, using Gaussian elimination would be highly impractical due to its many high **numerically instability**. Moreover, the  $\Theta(V^3)$  time complexity of elimination alone would become

**Algorithm 1: Fast Fourier Transform**

**Result:** The Fourier transform of  $\mathbf{x}$ , a vector  $[y_j = \sum_{k=0}^{m-1} x_k \omega_m^{jk} \mid j \in [0, m-1]]$

**Function** FFT(Coefficients  $\mathbf{x}$ ,  $m = |\mathbf{x}| = 2^d$ ):

```

if  $m = 1$  then return  $\mathbf{y}$ ;
 $\omega \leftarrow 1$ ;
 $(\mathbf{x}^{\text{even}}, \mathbf{x}^{\text{odd}}) \leftarrow ([x_0, \dots, x_{m-2}], [x_1, \dots, x_{m-1}]);$ 
 $(\mathbf{y}^{\text{even}}, \mathbf{y}^{\text{odd}}) \leftarrow (\text{FFT}(\mathbf{x}^{\text{even}}), \text{FFT}(\mathbf{x}^{\text{odd}}));$ 
for  $k \leftarrow 1$  to  $m/2 - 1$  do
     $y_k \leftarrow y_k^{\text{even}} + \omega \times y_k^{\text{odd}};$ 
     $y_{k+m/2} \leftarrow y_k^{\text{even}} - \omega \times y_k^{\text{odd}};$ 
     $\omega \leftarrow \omega \times \omega_n;$ 
return  $[y_0, y_1, \dots, y_{m-1}]$ 

```

unreasonable for larger ranges of values for  $F$ , needed to emulate several scoring functions as pointed out in Remark 2.2.2 below.

Enters the **(Inverse) Discrete Fourier Transform (IDFT)** [41, 42] which allows, in  $\mathcal{O}(m \log m)$  time, to recover the coefficients  $\mathbf{x} := (x_0, \dots, x_{m-1})$  of a polynomial  $P(x)$  of degree  $m-1$ ,  $m = 2^d$ , from its evaluation at the  $m$ -th **roots of the unity**. Indeed, let  $\omega_m := e^{2\pi i/m}$ , then the relationship between  $\mathbf{x}$  and  $\mathbf{y} := \{P(\omega_m^k)\}_{k=0}^{m-1}$  remarkably simplifies

$$\forall 0 \leq k < m : x_k = \frac{1}{m} \sum_{j=0}^{m-1} y_j \times \omega_m^{-j \cdot k}. \quad (2.15)$$

Moreover, evaluating all coefficients in  $\mathbf{x}$  can be achieved much faster than the  $\Theta(n^2)$  algorithm suggested by the above equation, using the **Fast Fourier Transform** to speed up the evaluation to  $\Theta(n \log n)$ .

To use the inverse DFT and interpolate  $\mathcal{Z}(x)$ , thus efficiently solving Problem 6, we first trivially extend Equation (2.13) to support complex arguments. Then we proceed to the main algorithm:

1. Round  $V+1$  upwards to  $m = 2^d$ ,  $d \in \mathbb{N}$ , the closest power of 2;
2. Evaluate  $\mathcal{Z}(\omega_m^0), \mathcal{Z}(\omega_m^1), \dots, \mathcal{Z}(\omega_m^{m-1}) \rightarrow y_0, y_1, \dots, y_{m-1}$  with  $\omega_m = e^{2\pi i/m}$ ;
3. Apply the FFT Algorithm 1 to  $[y_0, y_1, \dots, y_{m-1}]$  to obtain

$$\left[ x'_k := \sum_{j=0}^{m-1} y_j \omega_m^{jk} \right]_{k=0}^{m-1}.$$

Notice from Equation (2.15) that  $x'_{m-k} = m x_k$ ,  $\forall k \in [1, m-1]$  and  $x'_0 = m x_0$ ;

4. Return  $\left[ \mathcal{Z}^{[v]} := \frac{x'_{\Phi(v)}}{n} \mid v \in [0, V], \phi(0) = 0 \text{ and } \phi(v > 0) = V+1-v \right]$ .

This algorithm has time complexity in  $\Theta(V|\delta|\alpha^* + V \log V)$  and space complexity in  $\Theta(|Q| + V)$ . Moreover, being dominated by the independent evaluations of the  $\mathcal{Z}(\omega_m^i)$  terms, it can be largely accelerated by a trivial parallel execution, e.g. leading to a runtime in  $\Theta(|\delta|\alpha^* + V \log V)$  on  $V$  processors.

**Remark 2.2.2 (Multidimensional classified DP):** The DFT-based algorithm can easily be extended to support multiple scoring functions  $F_1, \dots, F_p$ , associated to maximum values  $V_1, \dots, V_p$ , to ultimately compute  $\mathcal{Z}^{[v_1, \dots, v_p]}$  the partition function classified by values  $(v_1, \dots, v_p)$  of the scoring functions.

This use-case can indeed be emulated using only a **single function**  $F$  defined on constructors as  $F(\lambda) = \sum_{i=1}^p F_i(\lambda) \prod_{j=1}^{i-1} (1 + V_j)$ , and extended additively into

$$F(T) = \sum_{i=1}^p F_i(T) \prod_{j=1}^{i-1} (1 + V_j) = F_1(T) + (1 + V_1) \times F_2(T) + (1 + V_1) \times (1 + V_2) \times F_3(T) \dots$$

In this setting, any value  $v$  for  $F$  **encodes** a vector  $(v_1, \dots, v_p)$ , such that the value of  $\mathcal{Z}^{[v_1, \dots, v_p]}$  can be read in  $\mathcal{Z}^{[v]}$  after the FFT determination of  $1 + V = \prod_i (1 + V_i)$  coefficients.

## 2.3 Probabilistic estimates

**Sampling methods** provide an alternative to exact computation, allowing the estimation of statistical ensemble properties for arbitrary features. Namely, general properties of Boltzmann ensembles can also be **estimated** from a **statistically representative sample** of candidate solutions, following the approach introduced by Ding and Lawrence [51] in the context of RNA folding prediction.

Considering a general **feature function**  $F : \Omega \rightarrow \mathbb{R}$ , one first **generates** a sequence  $s = (s_1, s_2, \dots, s_M) \in \Omega^*$  of  $M$  random elements from the search space, each drawn from the Boltzmann distribution, and then returns the **empirical mean**

$$\hat{F}(s) = \frac{\sum_{s \in s} F(s)}{M} \text{ such that } \lim_{|s| \rightarrow \infty} \hat{F}(s) = \mathbb{E}(F(S)). \quad (2.16)$$

For instance, the Boltzmann probability of a base-pair  $(i, j)$  can be estimated by considering a **Boolean feature function**  $F_{(i,j)}(s) = 1$  if  $(i, j) \in s$ ; 0 otherwise. More generally, this strategy allows to capture feature functions of arbitrary complexity, including those that are not additive with respect to any dynamic programming scheme.

A first algorithmic difficulty lies in the generation itself and is usually tackled through the simulation of a well-calibrated Markov process in classic Bayesian inference [5, 76]. However, complex distributions may induce high mixing times, before the process converges to its steady-state. Moreover, to ensure that the targeted distribution is indeed sampled (i.e. that the simulation has been executed sufficient long to guarantee converge)

requires highly-technical analyses [152] that do not appear to lend themselves to generic algorithmic design.

This section focuses on an alternative, where an unambiguous dynamic programming scheme for the partition function (but not necessarily with respect to the feature function) is used for the generation. Probabilities for each of the transitions are precomputed, so that generated objects are independent and follow the targeted distribution. This can be seen as an instance of the **recursive method** in random generation [71, 197].

### 2.3.1 Foreword: On the number of samples (165)

The choice of the **number of samples** is critical when using sampling to estimate statistical properties. This number should be large enough to yield accurate estimates, but low enough to preserve efficiency. Historically, and in many subsequent works, a sample size of 1 000 structures was proposed [51], somewhat irrespectively of the precise context. However, such a *one size fits all* approach may not yield accurate results, motivating the following discussion and recommendations.

The **empirical mean** estimator represents a sum of independent variables, meaning that classic concentration inequalities apply with minimal modifications. In particular, the **Hoeffding inequality** implies that, for any feature  $F$ :

$$\mathbb{P} \left( |\widehat{F}(S) - \mathbb{E}(F(S))| \geq \varepsilon \right) \leq 2 \exp \left( \frac{-2m\varepsilon^2}{c} \right), \quad (2.17)$$

where  $\varepsilon$  is a tolerated **absolute error** level,  $S$  is a random sample of size  $m$  and  $c := (\max_S(F(S)) - \min_S(F(S)))^2$  is a trivial upper bound of the variance of the feature. Note that when a feature function takes binary values 0/1, e.g. when estimating a probability, then one has  $c = 1$ . Equation (2.17) can be used to build a **confidence interval** at level  $(1 - \alpha)$ , for any value  $\alpha \in [0, 1]$ , and we get:

$$\left[ \widehat{F}(S) - \sqrt{\frac{c}{2m} \log \left( \frac{2}{\alpha} \right)}, \widehat{F}(S) + \sqrt{\frac{c}{2m} \log \left( \frac{2}{\alpha} \right)} \right].$$

This means that, over multiple estimations from sampled elements, at least a fraction  $(1 - \alpha)$  of the runs will produce errors smaller than  $\sqrt{\frac{c}{2m} \log \left( \frac{2}{\alpha} \right)}$ . This function can be inverted (numerically) to estimate the number  $m$  of samples that achieve an **absolute error** bounded by  $\varepsilon$  at least  $(1 - \alpha)$  of the times.

We report in Table 2.1 typical sample sizes required to achieve a given precision with reasonable probability when estimating probabilities (*i.e.* expectations of 0/1-valued features). For instance, to reach a **90% chance** of estimating a base pair probability **within 0.5% of its true value**, a total of **59 915 structures** should be generated. In

Tolerated Error	Frequency within tolerance		
	90%	95%	99%
$\varepsilon = 20\%$	37	46	66
$\varepsilon = 10\%$	150	184	265
$\varepsilon = 5\%$	599	738	1 060
$\varepsilon = 2.5\%$	2 397	2 951	4 239
$\varepsilon = 1\%$	14 979	18 444	26 492
$\varepsilon = 5\%$	59 915	73 778	105 966
$\varepsilon = 1\%$	1 497 866	1 844 440	2 649 159

Table 2.1: **Recommended number of samples to estimate probabilities (boolean features).** For instance, to ensure that the estimate falls within 1% of the true value for 95% of the runs, a large number of  $m = 18\,444$  structures should be generated.

particular, **1 000 structures** norm, usually considered in the literature, will guarantee a value **within 3%** of the true probability **only 2/3 of the times**, although this sample size will **almost always** (99%) return estimates **within 5%** of the correct value.

### 2.3.2 Statistical sampling

Let us now turn to the problem of sampling from the Boltzmann-Gibbs distribution, which we formalize as follows.

#### Problem 7 (Statistical sampling):

**Input:** Unambiguous DP scheme  $\Delta$  + scoring  $f$ , correct w.r.t. function  $E$ ;  $\beta \in \mathbb{R}$  a constant;  $M$  a **number of samples**

**Output:** A sequence  $s \in \Omega_{\Delta}^*$ ,  $|s| = M$  of elements of the search space, independently generated in the Boltzmann distribution

$$\mathbb{P}(s) = \frac{e^{-\beta \cdot E(s)}}{\mathcal{Z}_{\Delta}}, \forall s \in \Omega_{\Delta}^*.$$

In the context of an unambiguous/complete DP scheme, one can easily adapt the partition function recurrences into a **stochastic backtrack algorithm** to solve the statistical sampling problem, as done by Ding and Lawrence [51] in the context of RNA. First, the algorithm performs a preliminary computation of the partition function  $\mathcal{Z}_q$  for all states  $q \in Q$ , using the DP algorithm introduced for Problem 2. The algorithm then generates  $M$  independent random terms/solutions using a **stochastic backtrack** procedure  $\tilde{\mathcal{B}}$  described in Algorithm 2.

The correctness of the algorithm can be established by a simple induction. Indeed, consider a state  $q^* \in Q$  and assume that, invoked on any state accessible from  $q$ , the



**Algorithm 2:** Stochastic backtrack**Function**  $\tilde{\mathfrak{B}}(q \in Q)$ :
 $r \leftarrow \text{UnifRand}(\mathcal{Z}_q)$   $\triangleright$  Random  $r \in [0, \mathcal{Z}_q]$ 
**for**  $d := (q \xrightarrow{\lambda} q_1, \dots, q_k) \in \delta$  **do**
 $r \leftarrow r - e^{-\beta \cdot E(\lambda)} \times \prod_{i=1}^k \mathcal{Z}_{q_i};$ 
**if**  $r < 0$  **then**
 $\mathbf{return} \lambda(\tilde{\mathfrak{B}}(q_1), \dots, \tilde{\mathfrak{B}}(q_k))$   $\triangleright$   $d$  chosen with prob.  $\frac{e^{-\beta \cdot E(\lambda)} \times \prod_{i=1}^k \mathcal{Z}_{q_i}}{\mathcal{Z}_q}$ 

algorithm generates a term  $T \in \mathcal{T}_q$  with probability

$$\mathbb{P}(T \mid q) = \frac{e^{-\beta \cdot E(T)}}{\mathcal{Z}_q}$$

Executing Algorithm 2 on  $q^*$ , a derivation  $d := q^* \xrightarrow{\lambda} q_1, \dots, q_k$  is chosen with probability  $e^{-\beta \cdot E(\lambda)} \prod_{i=1}^k \mathcal{Z}_{q_i} / \mathcal{Z}_{q^*}$ , followed by the independent recursive generation of terms  $T_1, \dots, T_k$  from each of the states  $q_1, \dots, q_k$ . Due to the unambiguity of  $\Delta$ , the resulting term  $T^* := \lambda(T_1, \dots, T_k)$  cannot be generated from any other derivation of  $q^*$ , and its **emission probability** is thus

$$\begin{aligned} \mathbb{P}(T^* \mid q^*) &= \frac{e^{-\beta \cdot E(\lambda)} \prod_{i=1}^k \mathcal{Z}_{q_i}}{\mathcal{Z}_{q^*}} \times \frac{e^{-\beta \cdot E(T_1)}}{\mathcal{Z}_{q_1}} \times \dots \times \frac{e^{-\beta \cdot E(T_k)}}{\mathcal{Z}_{q_k}} \\ &= \frac{e^{-\beta \cdot (E(\lambda) + E(T_1) + \dots + E(T_k))}}{\mathcal{Z}_{q^*}} \\ &= \frac{e^{-\beta \cdot E(\lambda(T_1, \dots, T_k))}}{\mathcal{Z}_{q^*}} \end{aligned}$$

in which one recognizes the targeted distribution. The induction is completed by an inspection of leaf states, whose derivations do not require further recursive calls, to verify that they indeed locally induce a Boltzmann distribution.

The time complexity of the algorithm can be trivially bounded by  $\mathcal{O}(M \cdot |\delta| \cdot \alpha^*)$ , with  $\alpha^*$  being the max arity of a derivation, but is usually substantially lower since this upper bound would unrealistically require all states to be traversed by all backtracks. A more refined analysis could introduce upper bounds respectively on the numbers of states traversed during a generation ( $c^+$ ), and derivations available from a state ( $d^+$ ). Since one typically has  $c^+ \cdot d^+ \ll |\delta| \cdot \alpha^*$ , the worst-case complexity is then in  $\mathcal{O}(|Q| + |\delta| \cdot \alpha^* + M \cdot c^+ \cdot d^+)$ . A more precise analysis requires the definition of a notion of length associated with terms, opening the way for further optimizations, e.g. as described by Example 9 in the context of RNA folding.

#### Example 9: RNA BP folding – Statistical sampling

In the context of RNA conformational sampling, Algorithm 2 specializes into the following backtrack, preceded by a computation of the partition function  $\mathcal{Z}_{[i,j]}$  for all  $[i, j] \subseteq [1, n]$ , as described in Example 5.

```

Function SamplerRNA2D ( $[i, j] \subseteq [1, n]$ ,  $w \in \{A, C, G, U\}^n$ ,  $\mathcal{Z}$ ):
  if  $j - i \leq \theta$  then return  $\emptyset$ ;                                 $\triangleright$  Interval too short to support base pair
   $r \leftarrow \text{UnifRand}(\mathcal{Z}_{[i,j]})$                                  $\triangleright$  Random  $r \in [0, \mathcal{Z}_{[i,j]}[$ 
   $r \leftarrow r - \mathcal{Z}_{[i+1,j]}$ ;
  if  $r < 0$  then
    | return SamplerRNA2D ( $[i + 1, j]$ ,  $w$ )
  for  $k \leftarrow i + \theta + 1$  to  $j$  do
    | if  $w_i, w_k \in \mathcal{B}$  then
      | |  $r \leftarrow r - e^\beta \times \mathcal{Z}_{[i+1,k-1]} \times \mathcal{Z}_{[k+1,j]}$ ;           $\triangleright$  Defining  $\mathcal{Z}_{[j+1,j]} \equiv 1$ 
      | | if  $r < 0$  then
      | | | return
      | | |  $\{[i, k]\} \cup \text{SamplerRNA2D}([i + 1, k - 1], w) \cup \text{SamplerRNA2D}([k + 1, j], w)$ 

```

This function is a strict specialization of Algorithm 2, and randomly generates a secondary structure from the Boltzmann distribution when invoked on  $[1, n]$  with a sequence of length  $n$ .

As previously analyzed [148], its execution requires  $\Theta(n^2)$  operations in the worst-case scenario due to the possible  $\Theta(n)$  iterations of the **for** loop, followed by a recursion on a subinterval only marginally smaller ( $n - 2$ ). Its average-case behavior in the homopolymer model (all bases allowed to pair), or on a random RNA sequence of length  $n$ , was shown to be in  $\Theta(n\sqrt{n})$ , a behavior that holds for a large class of combinatorial classes [71].

The time complexity can be dramatically lowered to  $\mathcal{O}(n \log n)$  through a minor modification of the algorithm. Indeed, it suffices to substitute a **Boustrophedon order**

$$i + \theta + 1 \rightsquigarrow j \rightsquigarrow i + \theta + 2 \rightsquigarrow j - 1 \rightsquigarrow i + \theta + 3 \rightsquigarrow \dots \rightsquigarrow \left\lceil \frac{i + \theta + 1 + j}{2} \right\rceil$$

instead of the (implicit) sequential order  $i + \theta + 1 \rightsquigarrow i + \theta + 2 \rightsquigarrow \dots \rightsquigarrow j$  within the **for** loop to instantly obtain a  $\mathcal{O}(n \log n)$  time complexity for the backtracks in the worst-case scenario.

### 2.3.3 Non redundant sampling (118, 133, 165)

As shown in Section ??, the level of redundancy can be overwhelming within a Boltzmann-Gibbs sample. On the one hand, redundancy is instrumental to the **consistency** of the estimator (2.16), since convergence towards the correct expectation requires that the frequency of any given element converges towards its probability. On the other hand, redundancy appears to be **non-informative**, and even **wasteful**, while sampling from a Boltzmann-Gibbs distribution, or any distribution known *a priori*. Indeed, the exact emission probability of a generated sample can be derived exactly from the partition function and, in principle, should not need to be estimated from the frequency. Moreover, sampling can be used to recover diverse dominant solutions, with no further involvement in statistical estimates, e.g. in RNA kinetics studies [110, 133] or in automated software

testing [189].

One possible way to avoid redundancy is to perform **sampling without replacement**, forbidding the generation at the  $m$ -th step of any previously generated element  $[s_1, \dots, s_{m-1}]$ . The targeted emission probability  $\mathbb{P}(s_m \mid s_1, \dots, s_{m-1})$  is then

$$\mathbb{P}(s_m \mid s_1, \dots, s_{m-1}) = \frac{\mathbb{P}(s_m)}{1 - \sum_{i=1}^{m-1} \mathbb{P}(s_i)} \quad (2.18)$$

where  $\mathbb{P}(s)$  denotes the classic (redundant) Boltzmann-Gibbs probability. Note that successive generations are now dependent, so classic estimators such as the empirical mean of Equation (2.16) become biased and should not be used.

However, as shown by Rovetta *et al* [165], the expectation  $\mathbb{E}(F(S))$  of any given function  $F : \Omega \rightarrow \mathbb{R}$  in the Boltzmann distribution (or any known distribution) can still be estimated from a non-redundant sample  $[s_1, \dots, s_m]$  by

$$\tilde{F}([s_1, \dots, s_m]) = \frac{1}{m} \sum_{i=1}^m F(s_i) \left( 1 + (m-i) \times \mathbb{P}(s_i) - \sum_{j=1}^{i-1} \mathbb{P}(s_j) \right).$$

This estimator is provably unbiased, consistent and always yields lower expected variance than the empirical mean, motivating the following algorithmic problem.

**Problem 8 (Non-redundant sampling):**

**Input:** Unambiguous DP scheme  $\Delta$  + scoring  $f$ , correct w.r.t. energy function  $E$ ;  $\beta \in \mathbb{R}$  a constant;  $M$  a **number of samples**

**Output:** A **non-redundant sequence**  $[s_1, \dots, s_M] \in \Omega_{\Delta}^M$ , of elements:

$$\mathbb{P}(s_m \mid s_1, \dots, s_{m-1}) = \frac{\mathbb{P}(s_m)}{1 - \sum_{i=1}^{m-1} \mathbb{P}(s_i)}$$

In collaboration with Andy Lorenz [118], we considered a restriction of the above problem to the language of a weighted context-free grammar. Its main idea is to reinterpret derivation as a **purely sequential process**, where derivations involved in the production of a term are transformed into a **sequence of derivations** through preorder traversal of the derivation tree. Below is an example of a derivation tree  $t$  and its linear representation  $\omega_t$ :

$$t = \begin{array}{c} d_1 \\ / \quad | \quad \backslash \\ d_2 \quad d_3 \quad d_6 \\ / \quad \backslash \\ d_4 \quad d_5 \end{array} \quad \Leftrightarrow \quad \omega_t = d_1 \cdot d_2 \cdot d_3 \cdot d_4 \cdot d_5 \cdot d_6$$

Note that the derivation tree  $t$  can be unambiguously recovered from a sequence  $\omega_t$  through a prefix evaluation, and reinterpreted as a term  $T_{\omega_t}$ .

Such sequences are the ultimate product of a **sequential derivation process** over **immature words** in  $Q^* \times \delta^*$ , which starts from  $\omega := q_w$  and, at each step, considers the leftmost state  $q_L$  occurring in  $\omega = \omega_1 \cdot q_L \cdot \omega_2$  (if any), picks a derivation  $d = q_L \xrightarrow{\lambda} q_1 \cdots q_k$  and replace the occurrence  $q_L$  in  $\omega$  by  $d \cdot q_1 \cdots q_k$ . In the above example, the production of  $\omega_t$ , using this process, would be

$$\begin{aligned} q_w &\xrightarrow{d_1} d_1 \cdot q_2 \cdot q_3 \cdot q_6 \xrightarrow{d_2} d_1 \cdot d_2 \cdot q_3 \cdot q_6 \xrightarrow{d_3} d_1 \cdot d_2 \cdot d_3 \cdot q_4 \cdot q_5 \cdot q_6 \xrightarrow{d_4} d_1 \cdot d_2 \cdot d_3 \cdot d_4 \cdot q_5 \cdot q_6 \\ &\xrightarrow{d_5} d_1 \cdot d_2 \cdot d_3 \cdot d_4 \cdot d_5 \cdot q_6 \xrightarrow{d_6} d_1 \cdot d_2 \cdot d_3 \cdot d_4 \cdot d_5 \cdot d_6 \end{aligned}$$

Given an immature word  $\omega$ , the language  $\mathcal{L}(\omega)$  is the set of terms (transitively) derived from its states, and the partition function  $\mathcal{Z}(\omega)$  can be defined as

$$\mathcal{Z}(\omega) = \sum_{T \in \mathcal{L}(\omega)} e^{-\beta \cdot E(T)} = \prod_{q \in \omega \cap Q} \mathcal{Z}_q \prod_{q \xrightarrow{\lambda} q_1 \cdots q_k \in \omega \cap \delta} e^{-\beta \cdot E(\lambda)}. \quad (2.19)$$

In particular, given a set  $\mathcal{F}$  of forbidden terms, consider the random process which starts from  $\omega := q_w$  and, at each step, rewrites the leftmost  $q_L$  in  $\omega$  using some derivation  $d = q_L \xrightarrow{\lambda} q_1 \cdots q_k$ , chosen at random with probability

$$\mathbb{P}(d \mid \omega = \omega_1 \cdot q_L \cdot \omega_2) = \frac{\mathcal{Z}(\omega' := \omega_1 \cdot d \cdot q_1 \cdots q_k \cdot \omega_2) - \overline{\mathcal{Z}}(\omega')}{\mathcal{Z}(\omega) - \overline{\mathcal{Z}}(\omega)}, \quad (2.20)$$

with  $\overline{\mathcal{Z}}(\omega) := \sum_{T \in \mathcal{L}(\omega) \cap \mathcal{F}} e^{-\beta \cdot E(T)}$ . This process ultimately generates a derivation tree  $t^*$ , associated with a term  $T^*$ , resulting from a (unique) sequence

$$\omega_0 = q_w \xrightarrow{d_1} \omega_1 \xrightarrow{d_2} \omega_2 \rightarrow \cdots \rightarrow \omega_{p-1} \xrightarrow{d_p} \omega_p \xrightarrow{d_{p+1}} \omega_{t^*}$$

with probability

$$\begin{aligned} \mathbb{P}(t^*) &= \mathbb{P}(d_1 \mid \omega_0) \times \mathbb{P}(d_2 \mid \omega_1) \times \cdots \times \mathbb{P}(d_p \mid \omega_{p-1}) \times \mathbb{P}(d_{p+1} \mid \omega_p) \\ &= \frac{\mathcal{Z}(\omega_1) - \overline{\mathcal{Z}}(\omega_1)}{\mathcal{Z}(q_w) - \overline{\mathcal{Z}}(q_w)} \times \frac{\mathcal{Z}(\omega_2) - \overline{\mathcal{Z}}(\omega_2)}{\mathcal{Z}(\omega_1) - \overline{\mathcal{Z}}(\omega_1)} \times \cdots \times \frac{\mathcal{Z}(\omega_p) - \overline{\mathcal{Z}}(\omega_p)}{\mathcal{Z}(\omega_{p-1}) - \overline{\mathcal{Z}}(\omega_{p-1})} \times \frac{\mathcal{Z}(\omega_{t^*}) - \overline{\mathcal{Z}}(\omega_{t^*})}{\mathcal{Z}(\omega_p) - \overline{\mathcal{Z}}(\omega_p)} \\ &= \frac{\mathcal{Z}(\omega_{t^*}) - \overline{\mathcal{Z}}(\omega_{t^*})}{\mathcal{Z}(q_w) - \overline{\mathcal{Z}}(q_w)} = \frac{e^{-\beta \cdot E(T^*)} (1 - \mathbf{1}_{T^* \in \mathcal{F}})}{\mathcal{Z}_{q_w} - \sum_{T \in \mathcal{F}} e^{-\beta \cdot E(T)}} \\ &= \left\{ \frac{\mathbb{P}(T^*)}{1 - \sum_{T \in \mathcal{F}} \mathbb{P}(T)} \text{ if } T^* \notin \mathcal{F}; 0 \text{ otherwise} \right\} \end{aligned}$$

In other words, it suffices to follow the derivations probabilities of Equation (2.20) to induce the restricted Boltzmann-Gibbs distribution defined in Equation (2.18).

The only missing ingredient is a (fast) way to access  $\overline{\mathcal{Z}}(\omega)$ . To that purpose, we introduced a data structure  $P_{\mathcal{F}} = (V_{\mathcal{F}}, \cdot)$ , analogous to a **prefix tree**, which represents the sequences of derivations

$$\mathcal{D}_{\mathcal{F}} = \left\{ \omega_0 \xrightarrow{d_1^{[i]}} \omega_1^{[i]} \xrightarrow{d_2^{[i]}} \omega_2^{[i]} \xrightarrow{d_3^{[i]}} \cdots \right\}_{i=1}^{|\mathcal{F}|}$$

**Algorithm 3:** Non-redundant sampling algorithm

**Input:** Unambiguous DP scheme  $\Delta$ ; Number  $M$  of samples.

**Output:** Set of  $M$  random terms, distributed according to the non-redundant distribution of Equation (2.18).

```

Compute  $Z_q, \forall q \in Q$  ▷ Partition function algorithm (c.f. Problem 2)
 $P_{\mathcal{F}} \leftarrow (\perp, \emptyset);$  ▷ Tree initially restricted to root  $\perp$ 
 $\pi(\perp) \leftarrow 0;$ 
 $\mathcal{F} \leftarrow \emptyset$  ▷ Terms generated over previous iterations  $\rightarrow$  avoided
for  $i \in [1, M]$  do
   $(\omega, Z_\omega, u_\omega) \leftarrow (q_\omega, Z_{q_\omega}, \perp);$  ▷ Start from root
   $\mathcal{N} \leftarrow \{u_\omega\}$  ▷ Nodes traversed during generation
  while  $\omega \cap Q \neq \emptyset$  do
     $\omega_1.q_L.\omega_2 \leftarrow \omega$  s.t.  $\omega_1 \in \delta^*$  and  $q_L \in Q$  ▷  $q_L$  leftmost state in  $\omega$ 
     $r \leftarrow \text{UnifRand}(Z_\omega - \pi(u_\omega))$  ▷ Random  $r \in [0, Z_\omega - \overline{Z}(\omega)]$ 
    foreach  $d := q_L \xrightarrow{\lambda} q_1 \cdots q_k \in \delta$  do
       $\omega' \leftarrow \omega_1.d.q_1 \cdots q_k.\omega_2;$  ▷ Simulate derivation  $d$  in  $\omega$ 
       $u_{\omega'} \leftarrow \text{Child}(P_{\mathcal{F}}, u_\omega, d);$  ▷ Creates  $u_{\omega'}$  if absent ( $\pi(u_{\omega'}) \leftarrow 0$ )
       $Z_{\omega'} \leftarrow Z_\omega \times (e^{-\beta \cdot E(\lambda)} \prod_{j=1}^k Z_{q_j}) / Z_{q_L};$  ▷ Update  $Z$  per (2.19)
       $r \leftarrow r - (Z_{\omega'} - \pi(u_{\omega'}));$ 
      if  $r < 0$  then ▷ Happens with probability  $\frac{Z_{\omega'} - \overline{Z}(\omega')}{Z_\omega - \overline{Z}(\omega)}$ 
         $(\omega, Z_\omega, u_\omega) \leftarrow (\omega', Z_{\omega'}, u_{\omega'});$  ▷ Move to selected child
         $\mathcal{N} \leftarrow \mathcal{N} \cup \{u_{\omega'}\};$ 
        break;
     $\mathcal{F} \leftarrow \mathcal{F} \cup \{T_\omega\};$  ▷ Interpretation of  $\omega \in \delta^*$  as a term  $T_\omega$ 
     $\pi_\omega \leftarrow \prod_{\lambda \in \omega} e^{-\beta \cdot E(\lambda)};$ 
    foreach  $u \in \mathcal{N}$  do  $\pi(u) \leftarrow \pi(u) + \pi_\omega;$  ▷ Update traversed node weights
return  $\mathcal{F};$  ▷ Final non-redundant list of generated elements

```

performed during the generation  $\mathcal{F}$ . Any **node**  $u_\omega \in P_{\mathcal{F}}$  represents (implicitly) some immature word  $\omega \in \mathcal{D}_{\mathcal{F}}$ . There exists a **directed edge**  $(u_\omega \rightarrow u_{\omega'}) \in P_{\mathcal{F}}$ , **labeled by a derivation**  $d(u_\omega \rightarrow u_{\omega'}) \in \delta$  if and only if  $\omega \xrightarrow{d} \omega'$  occurs in some sequence of  $\mathcal{D}_{\mathcal{F}}$ . Finally, on each node  $u \in \mathcal{D}_{\mathcal{F}}$  we store a weight  $\pi(u)$ , updated after each generation such that the invariant  $\pi(u_\omega) = \overline{Z}(\omega)$  is maintained, allowing  $\mathcal{O}(1)$  access to  $\overline{Z}(\omega)$  while computing probabilities (2.20).

We obtain Algorithm 3, which generalizes earlier context-free versions [118, 133, 147] and can be essentially found in Juraj Michalik's PhD thesis [132]. The complexity of the algorithm is the same, up to implementation constants, as the redundant stochastic backtrack.

**Remark 2.3.1 (Rejection-based non-redundant sampling):** In the **uniform distribution** ( $\beta = 0$ ), a rejection-based sampling constitutes a reasonable alternative to produce  $M$  unique elements. Indeed, the search space is typically exponential  $|\Omega| \approx \zeta^n$  for some  $\zeta > 1$ . A classic **coupon collector analysis** shows that the expected number of generations to get  $|\Omega|$  distinct elements, is  $|\Omega| \cdot \mathcal{H}_{|\Omega|}$  with  $\mathcal{H}_m = \sum_{i=1}^m 1/i \in \Theta(\log n)$  the  $m$ -th harmonic number. Moreover, the expected number of generation is an increasing function of  $M$ , and is bounded by  $M \log |\Omega| \in \Theta(M.n)$ . In other words, a repeated generation of elements until  $M$  distinct elements are produced has  $\Theta(M.n)$  expected time in a uniform distribution, i.e. non-redundancy sampling only induces a **linear overhead** in comparison with the redundant one.

In Boltzmann-Gibbs sampling, however, the overhead is expected to grow exponentially with  $n$ , so a **rejection-based strategy is exponentially less efficient** than Algorithm 3.

#### 2.3.4 Adaptive sampling of constrained sequences (18, 90, 158)

As shown in Section 2.2.5, using explicit convolutions to compute the classified partition function quickly becomes prohibitively costly when (combinations of) expressive scoring functions are considered. Unfortunately, such a computation would be required to perform a **constrained sampling**, by adapting the stochastic backtrack of Section 2.3.2 to restrict the generation to objects having a value of interest.

As an alternative, in collaboration with Olivier Bodini [18], we introduced a **multidimensional Boltzmann sampling** method, also called **adaptive sampling** in the context of Bioinformatics applications [193] (due to the ubiquity of Boltzmann in Bioinformatics, leading to an overloaded nomenclature). This method is inspired both by the versatility of rejection methods in random generation (see Section 4.1), and by the typical concentration of distributions for additive parameters. This concentration is illustrated in 1D by Figure 2.1 for the number of base pairs in RNA, and in 2D by Figure 2.2 for two types of base pairs distinguishable in the basic Nussinov DP scheme.

##### **Problem 9 (Constrained sampling):**

**Input:** Unambiguous DP scheme  $\Delta$  + scoring  $f$ , correct w.r.t. function  $E$ ;  $\beta \in \mathbb{R}$ ; **scoring functions**  $F_1, \dots, F_k : \Lambda \rightarrow \mathbb{N}^+$  with  $V := \max_{T \in \mathcal{T}_\Delta} F(T)$  associated with **objective values**  $v_1^*, \dots, v_k^* \in \mathbb{N}^k$ ; and **number of samples**  $M$

**Output:** A collection  $C := [s_1, \dots, s_M]$  of random, Boltzmann-distributed w.r.t.  $E$ , elements of  $\Omega_\Delta$  such that

$$F_i(s_j) = v_i^*, \forall s_j \in C, \forall i \in [1, k]. \quad (2.21)$$

Rather than computing the costly convolution products described in Equation (2.12), we introduced a rejection based approach called **multidimensional Boltzmann sampling**,

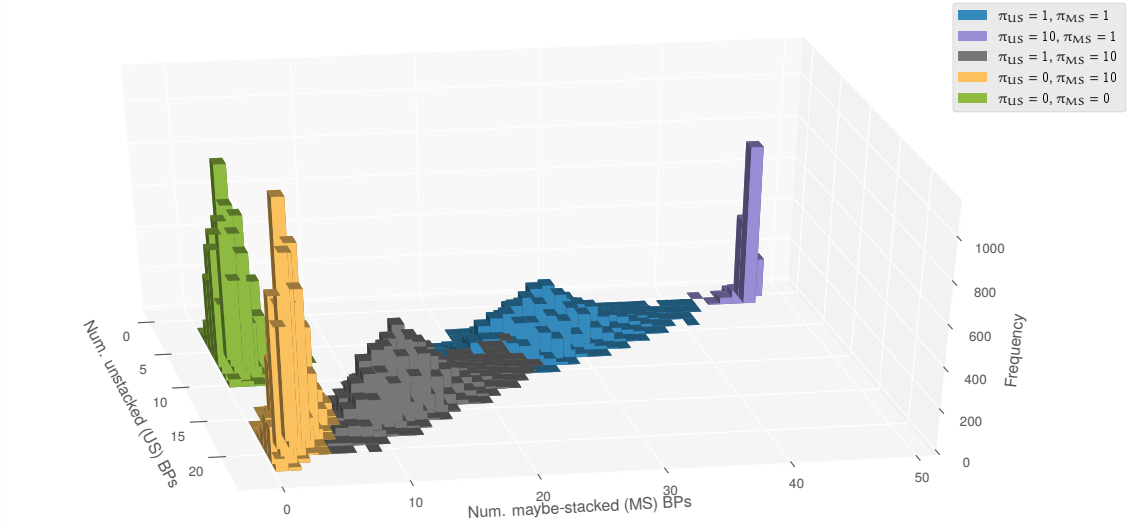


Figure 2.2: **Impact of weights on the distribution of base pairs in RNA 2D folding.** We distinguish **maybe-stacked** base pairs (MS), i.e. involving both ends  $(i, j)$  of an investigated interval  $[i, j]$  (alt. such that  $j + 1$  is base paired) from other base pairs, called **unstacked (US)**. Those two contributions can be distinguished on the classic Nussinov-Jacobson DP scheme, such that a weighted distribution can be induced.

whose principles are further described in Section 4.2.2. At its core, one uses a **weighting scheme** where numerical values  $(\pi_1, \dots, \pi_k)$  are associated to scoring functions, inducing a **weighted distribution**

$$\mathbb{P}_{\pi_1, \dots, \pi_k}(T) = \frac{e^{-\beta \cdot E(T)} \prod_{i=1}^k \pi_i^{F_i(T)}}{\mathcal{Z}_{\pi_1, \dots, \pi_k}} \quad (2.22)$$

where  $\mathcal{Z}_{\pi_1, \dots, \pi_k}$  is the weighted equivalent of the partition function.

Sampling within a weighted distribution can be done using Algorithm 2 based on a unifying scoring function  $E^*$  that aggregates all the contributions of individual features, such that

$$E^*(T) := \sum_{\lambda \in T} E(\lambda) - \frac{\sum_{i=1}^k F_i(\lambda) \times \log \pi_i}{\beta} = E(T) - F_i(T) \times \frac{\sum_{i=1}^k \log \pi_i}{\beta}.$$

This generation is coupled with a **rejection** step, which filters out the objects that do not fulfill the constraints described in Equation (2.21).

A crucial observation is that, in many cases, the successive choices performed during a stochastic backtrack can be interpreted as a large collection of **independent events** (i.e. occurrences of constructors), associated with bounded values of which the scoring functions are just the sum. The **central limit theorem** thus applies, and induces distributions that are (multidimensional) normal (a.k.a. Gaussian) with relatively low variance/co-variance values. In particular, once the weights are **calibrated** such that

$$\mu_i := \mathbb{E}_{\pi_1, \dots, \pi_k}(F_i(S)) = v_i^*, \forall i \in [1, k],$$

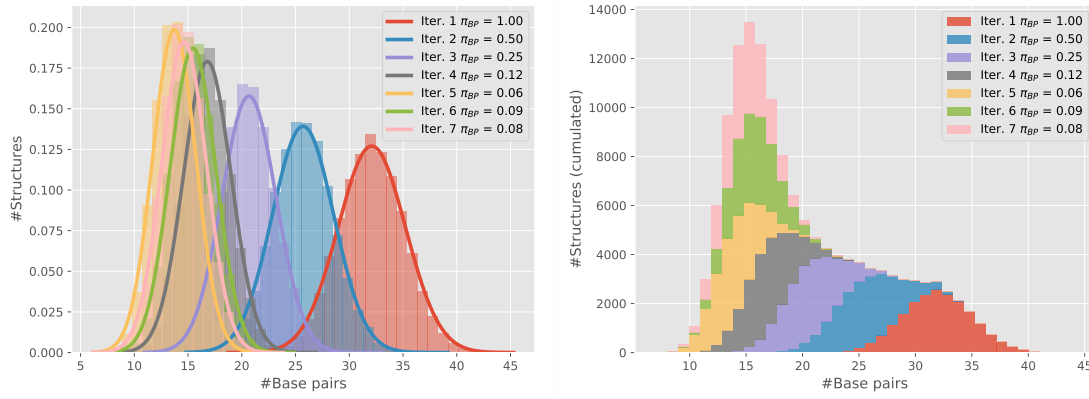


Figure 2.3: Using Algorithm 4 to sample 10 000 Boltzmann-distributed secondary structures having exactly 15 base pairs. For the RNA  $w = (ACGU)^{25}$  the expected number of base pairs is approximately 32.51 in the Boltzmann distribution, as shown in Figure 2.1 ( $T = 37^\circ\text{C}$ ,  $E(S) = -|S|$ ). A binary search is used to determine a weight  $\pi_{BP}$  such that the expected number of base pairs, empirically estimated at each iteration, matches the targeted one, accumulating across iterations.

then one has

$$\mathbb{P}(F_i(S) = v_i^*, \forall i \in [1, k]) = \frac{1}{\sqrt{(2\pi)^k \det \Sigma}}$$

where  $\Sigma$  is the co-variance matrix induces by scoring functions, such that  $\Sigma_{i,j} := \mathbb{E}((F_i(S) - \mu_i)(F_j(S) - \mu_j))$ . Assuming that  $k$  is constant, and denoting by  $V_i$  the max value taken by  $F_i$  over  $\Omega_\Delta$ , one expects  $\Theta(M\sqrt{\prod_i V_i})$  attempts in order to obtain  $M$  suitable element.

Our overall strategy to solve Problem 9 is summarized by (pseudo) Algorithm 4, and performs the weights optimization in parallel to the generation, accumulating suitable samples. Indeed, while following different overall distributions, samples generated across iterations of the weight optimization still follow the distribution (2.21) after the rejection step (see Section 4.1 and 4.2.1 for further discussions). The **weight optimization** can usually be tackled efficiently, since the expected number of occurrences of a given constructor is a strictly increasing function of the associated weight. The problem can generally be solved using tools in convex optimization [15], but can also be solved using **steepest descent** heuristics, or even a simple **binary search** in a mono-objective setting, are usually sufficient to solve the problem.

#### Example 10: RNA BP folding – Probing the Boltzmann ensemble of higher energy

Let us illustrate the general principle, and typical complexity analysis, of multidimensional sampling in a simple setting. Here, we want to estimate statistical properties of 2D structures with exactly  $v_{BP}^*$  base pairs, potentially far from the MFE, energy-wise, in a base-pair based energy model. First, we introduce a weight  $W_{BP} \in ]0, +\infty[$  that controls the expected number of base-pairs. Its incorporation to the partition function computation (see Example 5) and sampling procedure (see Example 9) only requires the introduction of  $\pi_{BP}$  within the contribution of valid base pairs, which becomes  $\pi_{BP} \times e^\beta \times$



**Algorithm 4:** High-level description of a multidimensional Boltzmann strategy for constrained sampling

**Input:** Num. samples  $M$ ; Functions  $F_1, \dots, F_k$  with targeted values  $(v_i^*)_{i=1}^k$ ;  
 Estimate  $\gamma$  for  $\sqrt{(2\pi)^k \det \Sigma}$

**Output:** Collection  $\{s_j\}_{j=1}^M$  of random, Boltzmann-distributed, elements such that  $F_i(s) = v_i^*, \forall i \in [1, k]$

$C \leftarrow \emptyset$ ;  
 $\pi \equiv (\pi_1, \dots, \pi_k) \leftarrow (1, \dots, 1)$ ;  
**while**  $|C| < M$  **do**  
      $D \leftarrow$  Sample  $M \times \gamma$  elements in the  $\pi$ -weighted distribution;  
     Update weight vectors towards  $\mathbb{E}_{\pi_1, \dots, \pi_k} (F_i(S)) = v_i^*, \forall i \in [1, k]$ ;  
      $C \leftarrow C \cup \{s \in D \mid F_i(s) = v_i^*, \forall i \in [1, k]\}$ ;  
**return**  $C$ ;

$$\mathcal{Z}_{[i+1, k-1]} \times \mathcal{Z}_{[k+1, j]}.$$

Then a simple binary search, explicited in the pseudocode below, is typically sufficient to guarantee a quick convergence towards suitable weights, as illustrated by Figure 2.3, and we get

$C \leftarrow \emptyset$ ;  
 $(\pi_{BP}, \pi_{min}, \pi_{max}) \leftarrow (1, 0, +\infty)$ ;  
**while**  $|C| < M$  **do**  
     Compute  $\mathcal{Z}_{BP}$ ; ▷ Using  $\Theta(n^3)$  algorithm  
      $D \leftarrow \{\text{SampleRNA2D}([1, n], w, \mathcal{Z}_{BP})\}_{i=1}^{2M}$ ;  
      $\widehat{\#bp} \rightarrow \frac{\sum_{S \in D} |S|}{|D|}$ ; ▷ Estimate expected #BPs for  $\pi_{BP}$   
     **if**  $\widehat{\#bp} < v_{BP}^*$  **then** ▷ Binary search update  
          $(\pi_{BP}, \pi_{min}, \pi_{max}) \leftarrow (\min(2 \pi_{BP}, \frac{\pi_{BP} + \pi_{max}}{2}), \pi_{BP}, \pi_{max})$ ;  
     **else**  
          $(\pi_{BP}, \pi_{min}, \pi_{max}) \leftarrow (\frac{\pi_{min} + \pi_{BP}}{2}, \pi_{min}, \pi_{BP})$ ;  
      $C \leftarrow C \cup \{S \in D \mid |S| = v_{BP}^*\}$ ;  
**return**  $C$ ;

## 2.4 Applications of ensemble dynamic programming

As can be seen in the previous section, the design of a suitable dynamic programming algorithm unlocks a wide array of potential inquiries regarding the properties of the ensemble of solutions, using algorithms that can be generically derived. As a consequence, the most crucial aspect of a DP algorithm, arguably the only one worthy of human attention, is the **design of a suitable dynamic programming scheme**, and the combinatorial decomposition onto which it is based. In particular, it is not only essential

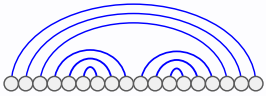
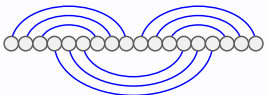
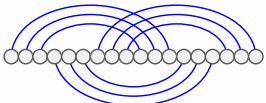
Conformation space		Base-pairs	Stacking-Pairs	Nearest-Neighbor
	MFE	P Nussinov and Jacobson [143]	P Jeong et al. [100]	P Zuker and Stiegler [213]
Non-crossing	Approx.	–	–	–
	MFE	???	NP-Hard Jeong et al. [100]	NP-Hard Jeong et al. [100]
Planar	Approx.	2-approx. $\pm$ Jeong et al. [100]	2-approx. Jeong et al. [100]	???
	MFE	P Tabaska et al. [181]	NP-Hard Lyngsø [124], Sheikh et al. [175] (any* $\Delta$ model) $\epsilon$ -approx. $\in \mathcal{O}(n^{4^{1/\epsilon}})$ Lyngsø [124], Sheikh et al. [175] 1/5 (any $\Delta$ model)	NP-Hard Lyngsø and Pedersen [123], Akutsu [3]
General	Approx.	–		APX-Hard Sheikh et al. [175]

Table 2.2: **Computational complexity of RNA 2D prediction with pseudoknots in the energy minimization paradigm.**

to ensemble analysis that the DP schemes are correct, but also that they are unambiguous and complete. In the following, I briefly mention some contributed schemes, addressing particular problems in RNA bioinformatics and comparative genomics.

#### 2.4.1 (Recursive) simple type pseudoknots and kissing hairpins [150]

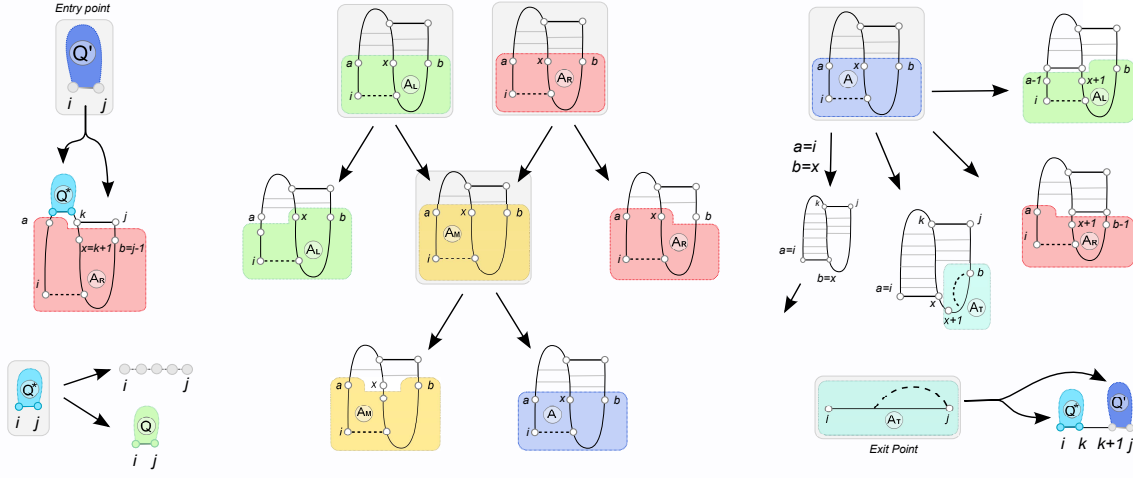
The absence of crossing base pairs in ensemble of conformations explored by classic algorithms represents a real limitation, and can be mainly attributed to:

1. The computational complexity of folding with pseudoknots in realistic models [3, 123], as summarized in Table 2.2, which we extended to large classes of energy models in collaboration with Saad Sheikh and Rolf Backofen [175];
2. The difficulty to imprint, at a combinatorial level, complex geometric constraints (e.g. chain closure) induced by base pairs, leading to ensemble of conformations that are highly dominated by unrealistic structures.

To work around such issues, a very rich body of literature [3, 19, 29, 53, 154, 156, 164, 183] has been dedicated to the design of dynamic programming algorithms capturing a subset of feasible conformations. However, many of those algorithms relied on an ambiguous DP scheme, preventing the natural exploration of ensembles of structures.

In collaboration with Cédric Saule [150], we revisited or designed popular DP schemes, visually described in Figure 2.4, for RNA folding with pseudoknots. In order to describe the generic derivation of ensemble algorithms (partition function, statistical sampling,

A – Unambiguous/complete  $\Theta(n^4)$  version of a DP scheme for **simple type pseudoknots** (3)



B – Unambiguous/complete  $\Theta(n^5)/\Theta(n^4)$  version of a DP scheme for recursive kissing hairpins (183)

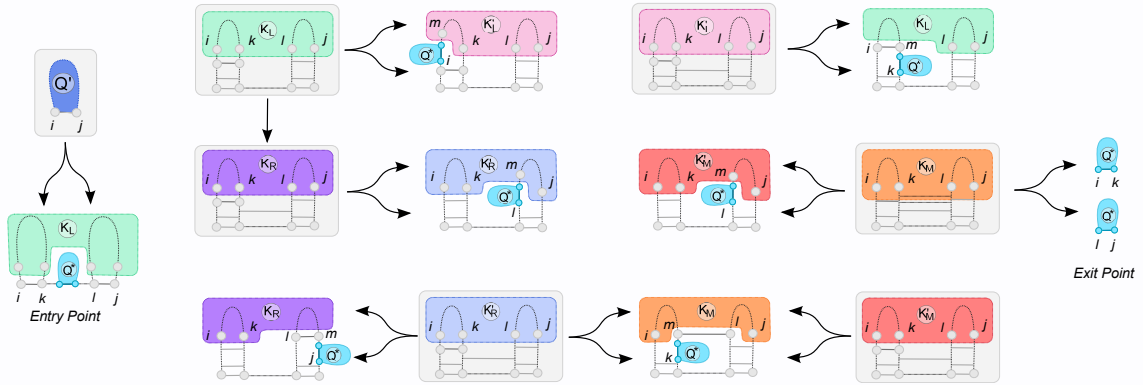


Figure 2.4: Unambiguous DP schemes for RNA folding in presence/absence of (recursive) pseudoknots [150].

inside/outside...), we used an **oriented hypergraph** (a.k.a. directed monotonic hypergraph) formalism introduced by Finkelstein and Roytberg [66], also occurring as a generic representation of DP in the field of natural language processing [98].

In addition to designing new DP schemes for two popular families, we adopted an enumerative combinatorics perspective to help in the proof of their unambiguity/correctness. Indeed, many conformation spaces amenable to dynamic programming can be enumerated, and generating functions  $PK(z) = \sum_{n \geq 0} p_n z^n$  with  $p_n$  the number of (pseudokotted) conformations over  $n$  nucleotides, have been established for the most popular DP schemes [168]. On the other hand, a full relaxation of base pair compatibility constraints (a.k.a. homopolymer model), allows to associate a generating function

$S_{\Delta}(z) = \sum_{n \geq 0} |\Omega_{\Delta_n}| z^n$  to a family of DP schemes  $\Delta_1, \Delta_2, \dots$  for instances of increasing length. In particular, the equality  $PK(z) = S_{\Delta}(z)$ , in conjunction with a proven completeness of the DP scheme, is sufficient to conclude on its unambiguity. Conversely, the conjunction of  $PK(z) = S_{\Delta}(z)$  and a proven unambiguity implies the completeness, effectively halving the work required to prove, typically through tedious inductions, those two properties. We used this technique to verify the properties of two contributed schemes described in Figure 2.4.

#### 2.4.2 Dual partition functions and evolutionary robustness of RNAs

Over several collaborative projects, we considered a dual version of the partition function, integrating over the space of compatible sequence/structure pairs [192, 193], or over the set of sequences associated with one [117, 157, 158] or several [88, 90] given structures.

In a first collaboration with Jérôme Waldispühl [192, 193], we leveraged a dynamic programming scheme introduced by the `RNAmutants` [191] method, to investigate the interplay between sequence and structure at prescribed GC-content. Its DP scheme explores all pairs  $(w, S)$  of compatible sequence and structure, classified by **Hamming distance**  $\delta(w, w^*)$  to a reference **wild type sequence**  $w^*$  of length  $n$ , allowing for a  $\Theta(n^5)/\Theta(n^3)$  computation of the **partition function of  $k$ -mutants**

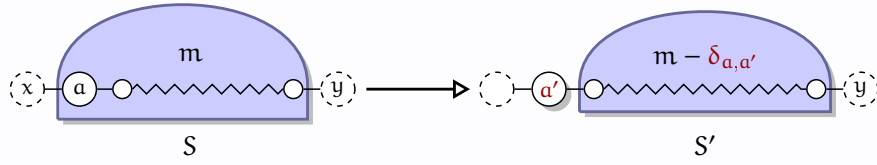
$$\mathcal{Z}_{[k]} := \sum_{\substack{w \in \Sigma^n \text{ s. t. } S \text{ comp. with } w \\ \delta(w, w^*)=k}} \sum e^{-\beta \cdot E(w, S)}.$$

A restricted version of the partition function, only considering the subinterval  $w_{i,j}^*$  of sequence  $w^*$ , can then be computed through

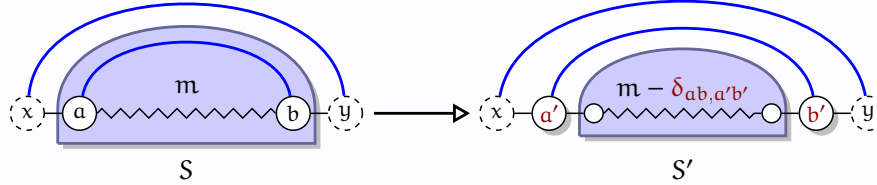
$$\mathcal{Z}_{i,j}^{[k]} = \sum_{b \in \Sigma} \mathcal{Z}_{i+1,j}^{[k-\delta_{w_i^*,b}]} + \sum_{b,b' \in \Sigma^2} \sum_{l=i+\theta+1}^j \sum_{k'=0}^{k-\delta_{w_i^*,b} - \delta_{w_l^*,b'}} e^{-\frac{\beta \cdot E_{b,b'}}{RT}} \cdot \mathcal{Z}_{i+1,l-1}^{[k']} \cdot \mathcal{Z}_{l+1,j}^{[k-k'-\delta_{w_l^*,b'}]} \quad (2.23)$$

in the case of a base-pair based energy model (although the implementation supports the Turner nearest-neighbor energy model [185]).

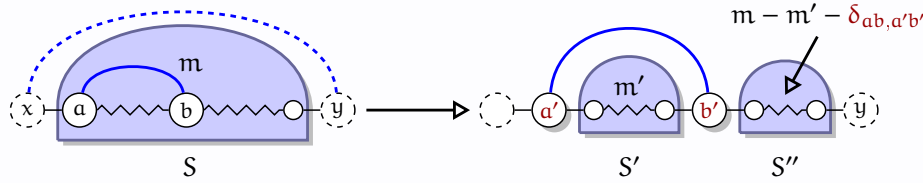
The same partition function was used to perform statistical sampling, revealing a major caveat of the model: due to the overwhelming impact of the GC-content on the partition function of a given sequence, the Boltzmann distribution was observed to disproportionately focus on sequences having extreme GC-contents, potentially impacting the accuracy of conclusions drawn while studying the evolutionary robustness of concrete sequences. To circumvent this issue, we introduced an adaptive sampling algorithm, an instance of multidimensional Boltzmann sampling [18], to direct the sampling towards areas of predetermined GC-content, allowing us to study the interplay between **evolutionary robustness** and **thermodynamic stability** [192, 193].



**Case 1:** First position is unpaired.



**Case 2:** Extremities are paired, nested within a consecutive base-pair, forming a stacking base-pair.



**Case 3:** First position is paired to some position, but not involved in a stacking pair.

Figure 2.5: **Classified DP scheme for the mutational landscape with respect to a single dominant conformation** [157].

Another instance of a DP scheme generating sequences considers the space of sequences compatible with a single structure, in an attempt to use structure conservation to detect **sequencing errors in metagenomic data sets** [157, 159]. In collaboration with Jérôme Waldispühl and Vladimir Reinharz, we considered an energy model of intermediate complexity, mainly including stacking pairs, and computed the **dual partition function** associated with a reference structure  $S^*$  and sequence  $w^*$ :

$$\mathcal{Z}_{S^*,[k]} = \sum_{\substack{w \text{ comp. with } S^* \\ \delta_{w,w^*}=k}} e^{-\beta \cdot E(w, S^*)}$$

using a modified version of (2.23), showcased in Figure 2.5, which runs in  $\Theta(n \cdot M^2)$  time and  $\Theta(n \cdot M)$  memory, with  $M$  the maximum number of suspected errors. In combination with an **outside contribution**, computed in a similar fashion, we were able to compute individual probabilities of assignments for each position. We also included contributions in our scoring scheme for the **isostericity** of base pair substitutions [115, 178] revealing a good capacity to discriminate, and to some extent correct, erroneous positions.

Dual partition functions are also essential for the **positive design of RNAs**, which we addressed as a random generation task over a series of collaborative works [88, 90, 117, 160, 209, 210]. In this context, the control of the maximum number of mutations is lifted, allowing a  $\Theta(n)$  time/space computation in the case of a target secondary structure. Recently [88, 90], we tackled the problem in its most general setting, adopting

a declarative approach where constraints are associated to a subset of positions within an RNA, either restricting their potential joint content or inducing specific free-energy contributions. The entire network of dependencies induced by the constraints can then be abstracted as an hypergraph, whose **tree decompositions** can be used to derive systems of DP equations, optimal in some sense, without any direct human intervention! We refer the reader to Chapter 3 for further context and motivation.

### 2.4.3 Unambiguous tree alignments [32, 33]

Generalizing the notion of sequence alignment, a **tree alignment** for two trees  $T_1$  and  $T_2$ , is a set of pairwise correspondences, or **matches**, between the nodes of  $T_1$  and  $T_2$ , consistent with the ancestry relationships in  $T_1$  and  $T_2$ . Nodes without a partner in  $T_1$  (resp.  $T_2$ ) are called insertions (resp. deletions). Equivalently, a tree alignment can be represented by a **supertree**, a tree  $T$  whose nodes are pairs of nodes  $(u_1, u_2) \in (T_1 \times T_2) \cup (T_1 \times \{-\}) \cup (\{-\} \times T_2)$  such that  $T_1$  (resp.  $T_2$ ) can be recovered from  $T$  by considering the first component  $u_1$  (resp. second component  $u_2$ ) in each pair, and finally contracting insertions/deletions nodes  $-$ . Tree alignments are natural in the context of RNA bioinformatics, where they represent a reasonable approach to measure the similarity between two secondary structures [17, 171].

Algorithmically, a classic problem is to find the optimal alignment between two given trees  $T_1$  and  $T_2$ , e.g. the supertree that minimizes the accumulated log odds of matches and indels. It is solved by Jiang, Wang and Zhang [103] (JWZ) based on the DP scheme:

$$\begin{aligned}
 \text{Align}_{\text{[Tree-alignment rules]}} \left( \begin{array}{c} \text{Tree 1} \\ \text{Tree 2} \end{array} \right) &= \min \left\{ \begin{array}{l} \text{Align} \left( \begin{array}{c} \text{Tree 1} \\ \text{Tree 2} \end{array} \right) + \text{Del}(\bullet) \\ \text{Align} \left( \begin{array}{c} \text{Tree 1} \\ \text{Tree 2} \end{array} \right) + \text{Ins}(\bullet) \\ \text{Align} \left( \begin{array}{c} \text{Tree 1} \\ \text{Tree 2} \end{array} \right) + \text{Subst}(\bullet, \bullet) \end{array} \right\} \\
 \text{Align}_{\text{[Forest-alignment rules]}} \left( \begin{array}{c} \text{Forest 1} \\ \text{Forest 2} \end{array} \right) &= \min \left\{ \begin{array}{l} \min_{\text{Forest 1}} \left\{ \text{Align} \left( \begin{array}{c} \text{Tree 1} \\ \text{Tree 2} \end{array} \right) + \text{Align} \left( \begin{array}{c} \text{Tree 1} \\ \text{Tree 2} \end{array} \right) + \text{Del}(\bullet) \right\} \\ \min_{\text{Forest 2}} \left\{ \text{Align} \left( \begin{array}{c} \text{Tree 1} \\ \text{Tree 2} \end{array} \right) + \text{Align} \left( \begin{array}{c} \text{Tree 1} \\ \text{Tree 2} \end{array} \right) + \text{Ins}(\bullet) \right\} \\ \text{Align} \left( \begin{array}{c} \text{Tree 1} \\ \text{Tree 2} \end{array} \right) + \text{Align} \left( \begin{array}{c} \text{Tree 1} \\ \text{Tree 2} \end{array} \right) \end{array} \right\}
 \end{aligned}$$

which can be computed in time  $\Theta(n_1 \cdot n_2 \cdot \max(n_1, n_2)^2)$ . Moreover, as shown by Herrbach et al. [92], the expected time complexity of the JWZ algorithm is actually in  $\Theta(n_1 \cdot n_2)$  for uniform random trees, the quartic complexity being circumscribed to trees with large degrees, thus having extremely low probability.

However, this decomposition scheme features a **high level of redundancy**, i.e. an alignment of  $T_1$  and  $T_2$  has, on average, an exponential number (on  $|T_1| + |T_2|$ ) of associated supertrees generated by the above DP scheme. Worse, some alignments are associated

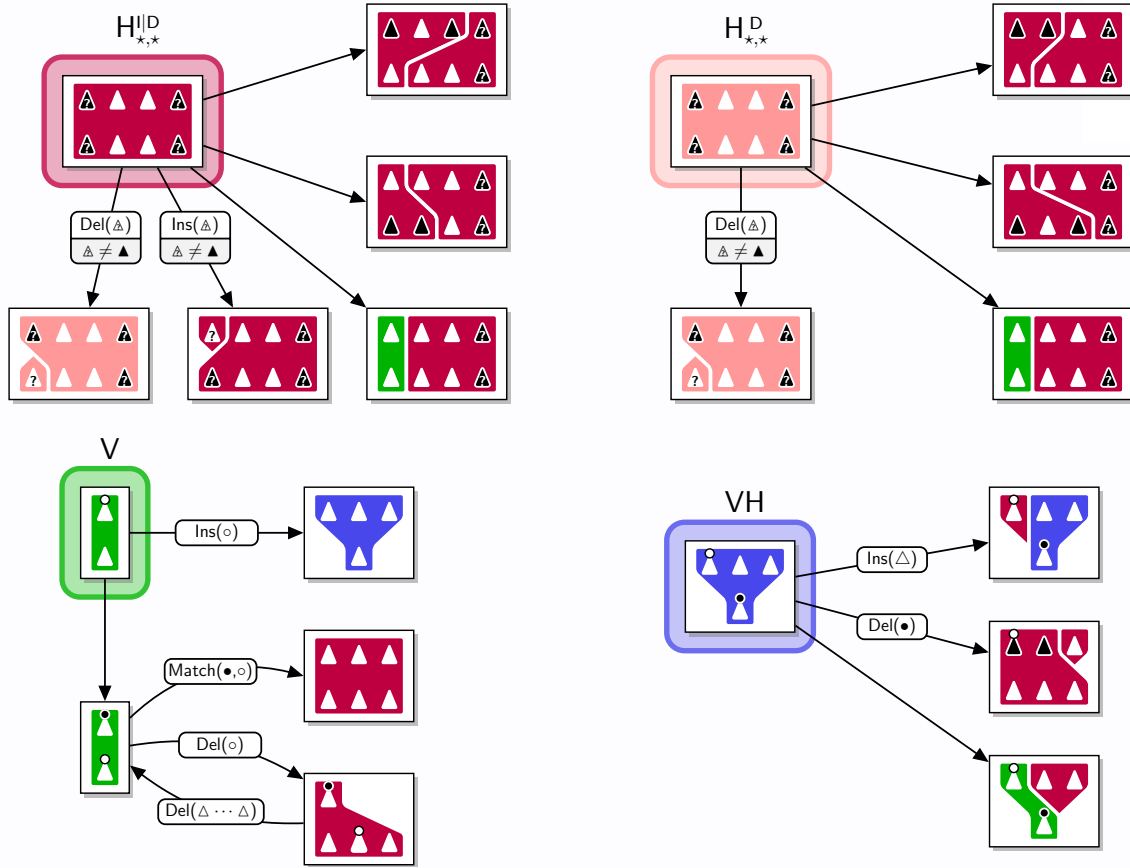


Figure 2.6: Unambiguous DP scheme for tree alignments.

by a unique supertree, so the algorithm cannot be easily adapted into a tool for ensemble analysis (e.g. to associate a notion of support for a given alignment).

With Cédric Chauve and Julien Courtiel [32, 33], we designed a **correct, unambiguous and complete DP scheme for tree alignments**, summarized in Figure 2.6. While highly technical in its design and proof, it preserves both the worst-case  $\Theta(n_1 \cdot n_2 \cdot \max(n_1, n_2)^2)$  and average-case  $\Theta(n_1 \cdot n_2)$  complexities of the JWZ algorithm. In fact, using analytic combinatorics, we could show that the average-case complexity of both algorithms, expressed as a function of the overall size  $n := n_1 + n_2$  of trees, only grows like  $\Theta(n\sqrt{n})$ . This counter-intuitive complexity, remarkably much lower than the  $\Theta(n^2)$  average-case complexity of sequence, is however only an artifact induced by the bimodal partition of length across a pair of trees. Indeed, upon generating a pair of trees  $(T_1, T_2)$  uniformly at random such that  $|T_1| + |T_2| = n$ , one typically gets a tree of size  $\Theta(\sqrt{n})$  and a tree of size  $\Theta(n)$ , rather than two sequences of length  $n/2$  in the case of sequences. One should therefore exercise caution when performing an average case analysis on pairs of objects, lest the final result may be more representative of the length distribution than of the underlying phenomenon...

The DP scheme could also be interpreted as a grammar generating a subset of supertrees,

bijectionally associated with tree alignments, and we used analytic combinatorics techniques to derive asymptotic properties of alignments (e.g. average # matches, number. . . ). The same could be done for the (redundant) family of supertrees produced by the JWZ DP scheme, and taking the ratio gave the expected number of supertrees, generated by JWZ for each alignment, confirming the exponential level of ambiguity.



## Chapter 3

### RNA design

The good performances of RNA structure prediction, especially at the secondary structure level, suggest harnessing their underlying models and methods to perform a **rational design of RNAs**, broadly defined here as the construction of novel nucleotide sequences achieving one (or several) predefined function(s). In the context of my work, a strong emphasis is put on the development of **computational methods for an automated RNA design**.

The originality of my contributions, spanning several distinct collaborations, lies in the adoption of a **random generation perspective** over the positive and, to some extent, negative design of RNA. Additionally, I have attempted to revisit negative design in a minimal setting, amenable to exact (if partial) combinatorial analysis, uncovering a promising, yet unexploited, notion of approximate design, that could lead to future developments.

**Outline.** I first describe in Section 3.1 the motivation, and various use-cases for RNA design. Section 3.2 describes partial exact solutions for a combinatorial version of the NP-hard inverse folding. Finally, Section 3.3 presents an original contribution to positive design based on a constrained random generation, surprisingly providing a satisfactory initialization steps for negative design.

The following summarizes, and attempts to unify, contributions described within the following list of articles published in journals and/or presented at conferences.

#### Associated contributions

A. Levin, M. Lis, Y. Ponty, C. W. O'Donnell, S. Devadas, B. Berger, and J. Waldispühl. A global sampling approach to designing and reengineering RNA secondary structures. *Nucleic Acids Research*, 40(20):10041–10052, Nov. 2012

Y. Zhou, Y. Ponty, S. Vialette, J. Waldispühl, Y. Zhang, and A. Denise. Flexible RNA design under structure and sequence constraints using formal languages. In **ACM-BCB 2013**, Bethesda, Washington DC, United States, Sept. 2013

V. Reinharz, Y. Ponty, and J. Waldispühl. A weighted sampling algorithm for the design of RNA sequences with targeted secondary structure and nucleotide distribution. **Bioinformatics**, 29(13):i308–15, July 2013

↑ V. Reinharz, Y. Ponty, and J. Waldispühl. A weighted sampling algorithm for the design of RNA sequences with targeted secondary structure and nucleotides distribution. In **ISMB/ECCB 2013**, Berlin, Germany, July 2013

Y. Zhang, Y. Ponty, M. Blanchette, E. Lecuyer, and J. Waldispühl. SPARCS: a web server to analyze (un)structured regions in coding RNA sequences. **Nucleic Acids Research**, 41(Web Server issue):W480–5, July 2013

M. Drory Retwitzer, V. Reinharz, Y. Ponty, J. Waldispühl, and D. Barash. incaRNAfbinv : a web server for the fragment-based design of RNA sequences. **Nucleic Acids Research**, 44(W1):W308 – W314, 2016

J. Hales, A. Héliou, J. Manuch, Y. Ponty, and L. Stacho. Combinatorial RNA Design: Designability and Structure-Approximating Algorithm in Watson-Crick and Nussinov-Jacobson Energy Models. **Algorithmica**, 79(3):835–856, 2017

↑ J. Hales, J. Manuch, Y. Ponty, and L. Stacho. Combinatorial RNA Design: Designability and Structure-Approximating Algorithm. In **CPM 2015**, Ischia Island, Italy, June 2015

A. Churkin, M. D. Retwitzer, V. Reinharz, Y. Ponty, J. Waldispühl, and D. Barash. Design of RNAs: comparing programs for inverse RNA folding. **Briefings in Bioinformatics**, 19(2):350–358, Jan. 2018

S. Hammer, W. Wang, S. Will, and Y. Ponty. Fixed-parameter tractable sampling for RNA design with multiple target structures. **BMC Bioinformatics**, 20(1):209, Dec. 2019

↑ S. Hammer, Y. Ponty, W. Wang, and S. Will. Fixed-Parameter Tractable Sampling for RNA Design with Multiple Target Structures. In **RECOMB 2018**, Paris, France, 2018

H.-T. Yao, C. Chauve, M. Regnier, and Y. Ponty. Exponentially few RNA structures are designable. In **ACM-BCB 2019**, pages 289–298, Niagara-Falls, United States, 2019. ACM Press

### 3.1 Why do we design RNAs?

At a fundamental level, designing RNAs represents the **ultimate stress-test** for our understanding of how RNA folds and acts on its environment. In this setting, one designs RNA sequences expected to fold into a predefined structure with respect to a folding prediction model. Synthesizing the resulting sequences, and using experimental methods to verify the actual adoption of the desired structure, one either validates the model or reveals some of its flaws, fueling and prioritizing further developments. Indeed, misfolding designed RNAs reveal gaps in our energy models and descriptors of the conformation spaces used by predictive algorithms. A similar strategy can be more generally used to test functional hypotheses involving the structure of RNA.

Molecular design also represents one of the primitives of **synthetic biology**, and RNAs have been used as biosensors [65], regulators, nano-material. . . Some naturally-occurring RNAs are sufficiently stable to fold in a modular fashion, enabling a *copy/paste approach* that simply combines existing RNAs. However, such a strategy is hindered, in an *in vivo* context, by the competition of artificial and endogenous RNAs, and by the intrinsic difficulty to produce orthogonal constructs over a limited number of available architectures. A rational design, coupled with an experimental filtering phase, is thus likely to represent the method of choice for future endeavors in RNA-based synthetic biology.

At a (primarily) sequence-based level, design is essential for future developments of **RNA-based therapeutics**. For instance, the recent discovery of viable treatments based on RNA interference is fueled by an understanding of how small RNAs can interfere with selected messenger RNAs to activate or inhibit them. In this context, an optimization of the nucleotide sequence of small interfering RNAs, akin to a design task, is crucial to ensure its efficacy and selectivity, mitigating the risk of side-effects for the drugs. Similarly, the specificity of genetic contents targeted by CRISPR-based editing, and thus its limited toxicity, is ensured by a redesign of guide RNAs. More generally, the adoption of a stable structure is very often a prerequisite for the interaction of RNAs with selected proteins [45], and are therefore crucial for the functionality of designed RNAs in a cellular context.

Design also helps in the **search for homologous RNAs**. Indeed, in multiple RNA families, selective pressure mostly applies at the structure level, hindering the discovery of new occurrences of a given RNA genes within the same organism (paralogs) or across available genomes (orthologs). If a structural model is known, and if the number of identified homologs is limited, a natural approach is to enrich homologs with sequences designed as to fold into the shared structure, in order to cover a larger proportion of the (neutral) sequence space.

### 3.1.1 The potential of RNA design for statistical evolutionary studies

I believe that RNA design should be systematically used within families of structured RNAs to assess the **significance of observed phenomena**. Indeed, a popular strategy consists in analyzing multiple sequence alignments of RNAs, and interpret quantitative observations in the light of an accepted pressures. To that purpose, one uses statistical tests to refute the hypothesis that the observation is merely induced by established properties. For instance, in the context of messenger RNAs, codon bias and the flanking of ORFs by start/stop codons should be accounted for when drawing consequences from the distribution of k-mers.

However, in the context of structural RNAs, most analyses only indirectly capture the overall stability (free-energy) of transcripts through a **preservation of the dinucleotide content**, based on the observation that sequences sharing a dinucleotide composition tend to have similar free-energy in their most stable 2D conformation. However, nothing prevents alternative dinucleotides compositions from achieving the same expected free-energy, so the distribution considered by classic tests is arguably *too narrow* to convincingly refute the hypothesis that an observed phenomenon can be imputed to the overall MFE stability (rather than be indicative of a new, yet to be explained, selective pressure).

Moreover, most families of functional RNAs not only share a comparable stability, but also share the adoption of a common functional structure. In this context, the significance of a property (e.g. overrepresentation of a motif, hypothesized as involved in a molecular recognition mechanism) should be assessed within the set of sequences that are believed fold into the common structure (+ additional established properties). In other words, one should be able to **perform a random generation of RNA designs** in order to suitably assess a notion of significance.

### 3.1.2 The different flavors of RNA design

RNA design targets a wide diversity of biological functions and, as such, encompasses of wide array of computational tasks. Two dominant paradigms dominate current computational approaches.

#### 3.1.2.1 Positive design

**Positive design** can be loosely defined as focusing on to **propensity** of produced RNAs to achieve a certain function. In a structural context, positive design usually involves

generating one or several sequences having good affinity (i.e. stability  $\approx$  minimum free-energy) towards a targeted structure. A more functional perspective will also consider the presence/absence of sequences motifs, or the affinity towards interacting with another molecule, as assessed by an energy model (e.g. RNA/DNA or RNA/RNA interactions) or a probabilistic model (e.g. protein/RNA interactions). This may include compositional biases, such as the GC-content.

### 3.1.2.2 Negative design

**Negative design** tasks loosely focus on the **specificity** of produced sequences, attempting to avoid unwanted functions, interactions, structures. . . From a computational perspective, such tasks are usually harder than the one induced by positive design. Indeed, they typically require solving an **inverse combinatorial problem**, generally formulated as follows.

**Problem 10 (Inverse combinatorial problem):**

**Input:** Expected output  $y^*$  and objective function  $f$ .

**Output:** Instance  $x$  such that

$$\left\{ y \mid f(y; x) = \max_{y'} f(y'; x) \right\} = \{y^*\}.$$

The difficulty of those problems stems from the potential difficulty of optimizing the objective function, so that the natural decision version of the problem may not even belong in NP. Moreover, even when the optimization (coupled with the production of a certificate for the unicity of the optimal solution) can be performed in polynomial time, the inverse problem can still be NP-hard [20].

In the context of structural RNA design, negative design is usually referred to as the **inverse folding** problem, and historically consists in producing a nucleotide sequence, predicted to (uniquely) fold into a targeted structure with respect to the Minimum Free-Energy (MFE) criterion. Variants of the inverse folding problem consider the minimization of various notions of **defects**, including the: **MFE defect**, the free-energy difference between the target and MFE structure (or first suboptimal structure if the target and MFE structures coincide); **probability defect**, the Boltzmann probability for the sequence to adopt an alternative structure; or the **ensemble defect**, the expected base-pair distance between the target and a random structure in the Boltzmann-Gibbs distribution.

### 3.1.2.3 Computational approaches

Due to the robust NP-hardness of inverse folding [20, 91, 172], the dominant paradigm, shared by multiple computational approaches [7, 9, 28, 77, 89, 95, 116, 208], combine an initial generation of **seed** sequences, akin to a (sometimes very rudimentary) positive design, with further rounds of **optimization** using a **local search**. Inverse folding has also been tackled using a variety of metaheuristics, including genetic algorithms [63, 64, 125], ant-colony optimization [107]. . .

Finally, the imperfect nature of the objectives of RNA design has left the door open for machine learning, recently attracting the predatory attention of neural networks [166] and/or crowdsourcing [109], with modest performances and disputable rationales.

## 3.2 Exact combinatorial negative design (85, 86)

The **computational hardness of inverse folding**, which had essentially eluded a whole community for almost three decades, was finally rigorously established in 2018 by Bonnet et al. [20].

It was preceded by an effort worthy of a mention by Schnall-Levin et al. [172], who generalized the RNA design problem into the **inverse Viterbi** problem, consisting of figuring out a sequence  $w$ , in a probabilistic grammar such as a e.g. Hidden Markov Model (HMMs) or a Stochastic Context-Free Grammar (SCFG), such that a predefined sequence of states is the most likely parsing for  $w$ . The problem was found to be NP-hard, even for HMMs, leading the authors to argue on an implied hardness of RNA design, since RNA folding can be viewed as a special case of SCFG parsing. However, while the hardness of the inverse Viterbi problem for SCFGs clearly follows from their result, it did not imply the hardness of the problem for a given input grammar, as is the case in RNA design.

Over the course of a collaboration with Jozef Hales, Alice Héliou, Jan Manuch and Ladislav Stacho [85, 86], we investigated a minimal installment of negative design, aiming to provide partial characterization for the space of designable structures. Namely, we consider the **Watson-Crick energy model**, initially restricting admissible base pairs to  $\{A, U\}$  and  $\{G, C\}$ . In this restricted context, inverse folding greatly simplifies and we get the following formulation.

**Problem 11 (Watson-Crick combinatorial design):****Input:** Secondary Structure  $S^*$  over  $n$  nucleotides**Output:** Sequence  $w$  such that

$$\forall S \in \mathcal{S}_n \text{ compatible with } w, S \neq S^* : |S| < |S^*|.$$

Bonnet et al. [20] showed that a slightly restricted version of the problem, forcing the assignment of predefined nucleotides to positions in  $w$ , is NP-hard. We thus focused our efforts on characterizing of designable subclasses of structures.

**3.2.1 Basic results**

Our results require a reminder of the representation for the secondary structure, illustrated in Figure 3.1. Such a representation abstracts a structure as a tree, rooted in a virtual node, with other nodes either being **base-pairs nodes** (labeled by interacting positions) or **unpaired nodes** positions (labeled by a single position). We define the **inner degree** of a node as its number of base pair neighbors. For instance, the node 4-37 in Figure 3.1 has inner degree 3.

**Theorem 1 (Basic designable classes):** *Any structure  $S$ , represented as a tree, can be designed in linear time as soon as (at least one) of the following conditions is met:*

1. *Each node in  $S$  has inner degree at most two;*
2.  *$S$  is saturated (no unpaired base) + any node has inner degree at most four;*

Moreover, we established that the set of designable structures is **closed under  $k$ -stuttering**. In other words, if a structure  $S$ , represented as a dot-parenthesis word  $S_1.S_2 \dots S_n$ , admits a known design  $w := w_1.w_2 \dots w_n$ , then

$$w_1^k.w_2^k \dots w_n^k \text{ is a design for } S_1^k.S_2^k \dots S_n^k, \forall k > 0.$$

The stuttering operation can also be described at the tree level, and essentially (up to renumbering of nodes) amounts to: i) duplicating each unpaired node  $k$  times; and ii) replacing any base pair node with a sequence of base pair nodes of length  $k$ .

We also identified two undesignable motifs, a.k.a. **local obstructions**, whose presence is sufficient to forbid any design for the structure. Namely, it is probably impossible to design any structure containing a multiloop with degree  $\geq 5$  ( $m_5$  **motif**), or with degree  $\geq 3$  with  $\geq 1$  unpaired base(s) ( $m_3 \circ$  **motif**). Such motifs are universal to any version of design, as they are essentially a consequence of the local nature of classic

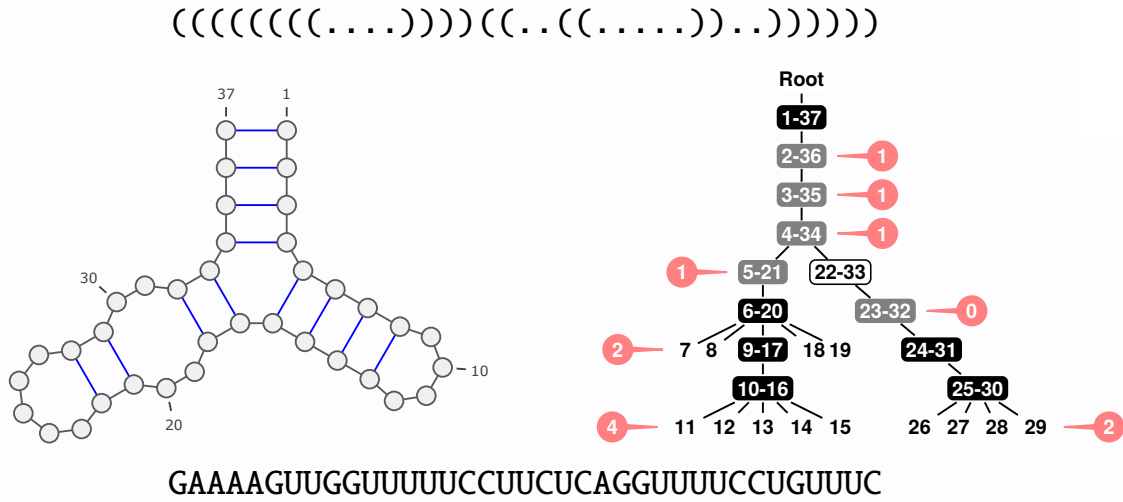


Figure 3.1: **Tree representation and separated coloring for a designable secondary structure.** The secondary structure admits a separated black/white/gray coloring of its base pairs, i.e. levels (red callouts) of gray base pairs (1 and 0) and unpaired base/leaves (2 and 4) do not overlap. The structure can thus be designed, e.g. through the sequence below.

energy models, defects definition and DP-schemes used for structure prediction. We exploited this property in subsequent work focusing on the enumeration of designable structures Yao et al. [206], further described in Section 5.2.4.

### 3.2.2 Design as a tree coloring problem

Finally, we introduced a sufficient condition for designability, based on a special coloring of the tree representation. Specifically, we considered a **3-coloring**  $\chi$ , associating either black (G-C), white (C-G) or gray (A-U or U-A) to the base-pair nodes of a structure  $S$ . Unpaired leaves are not colored, and are typically assigned a U. We define  $\chi$  as **proper** such that:

- Any node has at most one black node, one white node and two gray nodes;
- White nodes have no black children;
- Black nodes have no white children;
- Gray nodes have at most one gray child.

Any failure to obey those rules implies the existence of **local rewirings of base pairs**, leading to alternative configurations, and the sequence implied by the coloring is thus not a design. However, a suitable design also requires the global absence of competing structures, so we need a second property to conclude on the validity of the induced design.



We introduce the notion of **level**  $\text{level}_\chi(v)$  of a node  $v$  within a coloring  $\chi$ , defined as the number of black nodes, minus the number of white nodes, encountered while going from the node to the root. It can also be defined recursively as

$$\text{level}_\chi(v) = \begin{cases} 0 & \text{if } v = \text{Root} \\ \text{level}_\chi(p) + 1 & \text{if } \chi(p_v) = \text{black} \\ \text{level}_\chi(p) - 1 & \text{if } \chi(p_v) = \text{white} \\ \text{level}_\chi(p) & \text{if } \chi(p_v) = \text{gray}. \end{cases}$$

where  $p_v$  denotes the parent of  $v$ . Figure 3.1 illustrates the levels associated with a proper coloring.

**Definition 3.2.1 (Separated 3-coloring):** Given a secondary structure  $S$ , a **proper** 3-coloring  $\chi$  is **separated** if and only if

$$\{\text{level}(v) \mid v \text{ base paired in } S\} \cap \{\text{level}(v) \mid v \text{ unpaired in } S\} = \emptyset.$$

This property is actually sufficient to rule out the absence of an alternative structure having as many (or more) base pairs. The underlying intuition is that, in any competing structure, some  $U$  from an unpaired node must be paired to an  $A$  within a gray node. In a separated coloring, those interacting nucleotides are at different levels, so the alternative  $A = U$  base pairs provably delimits two regions, each having different numbers of  $G$ s and  $C$ s. It follows that, since pseudoknots are forbidden, then some  $G$  and  $C$  must remain unpaired in the final structure. In other words, the competing structure has less base pairs than the target.

**Theorem 2 (Designability of separable structures):** *Any secondary structure  $S$  that admits a proper and separated 3-coloring  $\chi$  can be designed. Moreover, if  $\chi$  is known, then a design can be produced in linear time.*

Figure 3.1 illustrates the concept of separated coloring, and some associated sequence. It is worth noting that the existence of a separated coloring does not represent a necessary condition for design. Moreover, finding a proper/separated coloring remains an open algorithmic problem, so this result may appear, at first sight, devoid of any practical consequence. Nonetheless, it is at the root of the notion of structure-approximating design, developed over the next section.

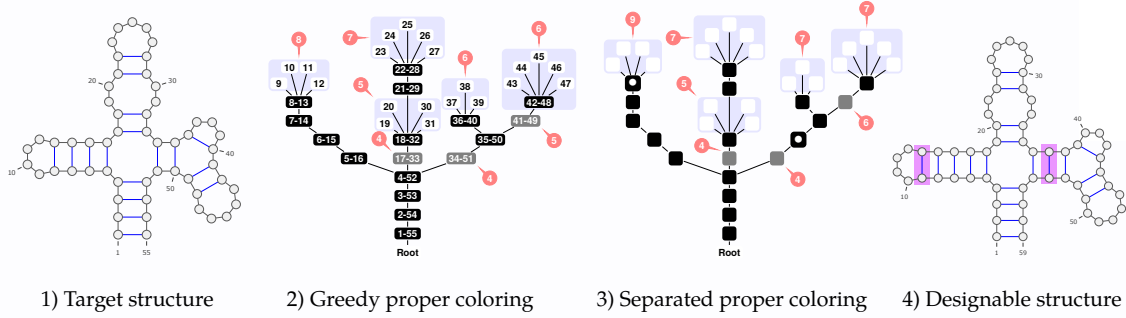


Figure 3.2: **Execution of the structure-approximating design algorithm.** An admissible target structure (1) is represented as a tree, and decorated into a proper 3-coloring by a greedy algorithm (2). A separated proper coloring (3) is obtained by adding selected nodes (dotted) to prevent direct adjacency and assign different parities for gray and unpaired nodes. The resulting coloring guarantees a linear-time design for a structure (4) that only differs by a few base pairs (purple) from the target.

### 3.2.3 Structure-approximating design

While it remains unclear how a separated coloring can be efficiently found, it is easy to adapt a proper, yet non-separated, coloring into a separated one through the insertion of a limited number of nodes. Indeed, one can insert a small number of additional base paired nodes in order to separate gray paired and unpaired nodes, i.e. assigning them different levels to ensure the separated nature of the resulting coloring. This strategy works for any tree that avoids the  $m_5$  and  $m_{3\circ}$  motifs, noted that the presence of those motifs remains unchanged by the addition of base pairs.

A first idea, not necessarily optimally parsimonious, consists in **offsetting gray nodes to even levels**, and **unpaired nodes to odd levels**. This can be done, for any structure  $S$  avoiding  $m_5$  and  $m_{3\circ}$ , in three steps:

1. Produce a initial **greedy proper coloring** of the tree:
  - Assign to the ( $\geq 4$ ) children of the root, colors black→white→gray→gray in this order, and descend recursively into its children;
  - On any internal nodes, colored in black (resp. white and gray), assign colors b→g→g (resp. w→g→g and b→w→g) to its children;
2. Correct the coloring into a separated one:
  - Insert a black paired node to prevent any direct adjacency between a gray paired node and an unpaired leaf. Note that such cases only arise for gray nodes associated with **isolated base pairs**;
  - Traversing the nodes of the tree in preorder, duplicate the parent of any gray node at odd level, or unpaired node at even level (the parent of such nodes is provably either black or white).

After executing this algorithm on a structure  $S$ , one obtains a tree/coloring pair  $(S', \chi')$  where  $\chi'$  is separated. Moreover, at most two base pairs are added to each of the helices of  $S$  (brought down to one base pairs in the absence of isolated base pairs).

**Theorem 3 (Approximated design):** *Given a structure  $S$  (resp. with no isolated base pair) avoiding  $m_5$  and  $m_{3\circ}$ , then a designable structure  $S'$  can be obtained by adding  $\leq 2$  (resp.  $\leq 1$ ) base pair(s) to each helix in  $S$ . Moreover,  $S'$  can be built, and designed for, in linear time.*

In practice, the number of added base-pairs usually remains modest, as shown in Figure 3.2. Future extensions of this work include the design of algorithms for adding a minimal collection of base pairs to make the structure/coloring separated.

More importantly, we believe that the **separated** property should find a natural equivalent in more sophisticated energy models, including as the Turner energy model [185]. A first extension of our main results to virtually any weighted base-pair model substantiates this hope.

**Proposition 4 (Extension to weighted models):** *Theorems 1, 2 and 3 hold in any energy model that is defined additively on base pairs, each having individual free-energy contributions:*

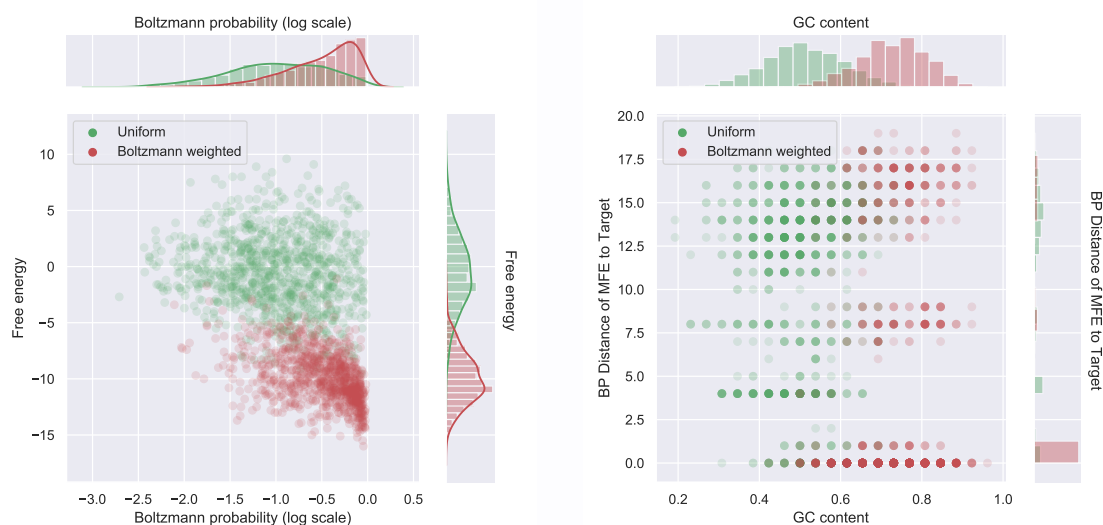
$$E(G, C) = E(C, G) \rightarrow \alpha, E(A, U) = E(U, A) \rightarrow \beta \text{ and } E(G, U) = E(U, G) \rightarrow \gamma$$

*such that  $\alpha, \beta < \gamma$ .*

### 3.3 Stochastic positive design under constraints

In the context of positive design, computational objectives are typically sufficiently simple to be amenable to a random generation. Such an approach is preferable to a **one-shot resolution** of the constraints imposed by the model of functions, for a variety of reasons:

- Positive design is typically used to generate an initial sequence, the **seed sequence** of existing methods based on meta-heuristics. Generating the initial at random thus helps meta-heuristics (local search, or genetic algorithms) overcome barriers in the objective function;
- Although currently underexploited, statistics derived from a set of random designs provide information on a null model for RNA evolution, allowing to assess the significance of observed phenomenon [151, 209];



1) Boltzmann probability *vs* Free-energy of target

2) GC-content *vs* BP distance between MFE and target

Figure 3.3: **Comparison of uniform and Boltzmann-Gibbs distributed seeds.** For the target structure  $(((((\dots)))) \dots (((((\dots))))))$ , 1 000 sequences were generated, either uniformly or in the Boltzmann distribution, using *IncaRNA* [160].

- A diverse set of sequences allows for further filtering based on properties that are not easily necessarily imprinted at a computational level.

### 3.3.1 Sampling from the dual-partition function [117, 158, 160]

This last point is at the foundation of one of the PhD contribution of Vladimir Reinharz, supervised by Jérôme Waldispühl and myself [158, 160]. Namely, as illustrated in Figure 3.3 for a toy structure, the Boltzmann probability typically correlate strongly negatively meaning with the free-energy. This means that sequences with low free-energy (positive design) is also likely to optimize classic objectives of design, and thus represents a potentially promising approach for the inverse folding problem. This property was exploited by [28], which started its local search from the MFE sequence for a given structure, leading to improved performances.

We considered the **dual-partition function**  $\mathcal{Z}_{S^*}$  of a structure  $S^*$ , defined as:

$$\mathcal{Z}_{S^*} = \sum_{w \text{ compatible with } S^*} e^{-\beta \cdot E(w, S^*)}$$

where  $E(w, S)$  represents a simplified free-energy model, restricted to stacked base pairs, and  $\beta$  is an arbitrary constant. We then turned to a random generation of sequences in the Boltzmann-Gibbs distribution over sequences, where any sequence  $w$ , compatible

with  $S^*$ , is emitted with probability

$$\mathbb{P}(w \mid S^*) = \frac{e^{-\beta \cdot E(w, S^*)}}{\mathcal{Z}_{S^*}}.$$

We refer the reader to Chapter 2.3.4 for the algorithmic aspects of computing the dual partition function, and performing the stochastic backtrack required for the sampling.

**Remark 3.3.1:** Note that, while it constitutes an approximation of the established Turner energy model [185], our stacking-based model correlates very well with the former, and virtually does not impact the above probability. In particular, the lack of support for multiple loops provably does not impact the distribution. Indeed, the contribution  $E_M$  of a multiple loop within the Turner model depends on its structural characteristics, but remains independent of its precise nucleotide content. Including such a contribution to our energy model would therefore increase the Boltzmann factor of any sequence by a factor  $e^{-\beta \cdot E_M}$ , homogeneous across sequences. Meanwhile, the partition function would be inflated by the exact same factor  $e^{-\beta \cdot E_M}$ , so that the two terms would cancel in the emission probability, leaving it unchanged.

This approach generates sequences which not only have, unsurprisingly, lower free-energies than those produced by uniform sampling, but also much larger Boltzmann probabilities, as illustrated by the left panel of Figure 3.3. However, further analyses (Figure 3.3, right panel) reveal that this increased suitability to the objectives of design comes at the cost of a **drift of the GC-content towards high values** (typically 75-80%). This drift is problematic since as wild-type sequences within living organisms often present medium or low GC-content, presumably to offer better transcription rates and/or structural plasticity [54].

As a countermeasure, we introduced a **rejection strategy**, an instance of multidimensional Boltzmann sampling (see Section 2.3.4 and 4.2.2), to generate sequences at a constrained GC-content, introducing a weight  $\kappa_{gc}$  to induce a distribution

$$\mathbb{P}_{\kappa_{gc}}(w \mid S^*) \propto \kappa_{gc}^{|w|_G + |w|_C} \times e^{-\beta \cdot E(w, S^*)}.$$

As shown in Figure 3.4, the drift of Boltzmann-generated sequences towards high GC contents can be counterbalanced, while preserving much better free-energies than induced by the Boltzmann distribution. Moreover, this strategy induces Boltzmann probabilities that are much higher than induced by a uniform sampling, as well as a much better proportion of sequences adopting the target structure as its MFE. In other words, this **positive design strategy** constitutes a promising approach, possibly in combination with further **local optimizations**, for negative design.

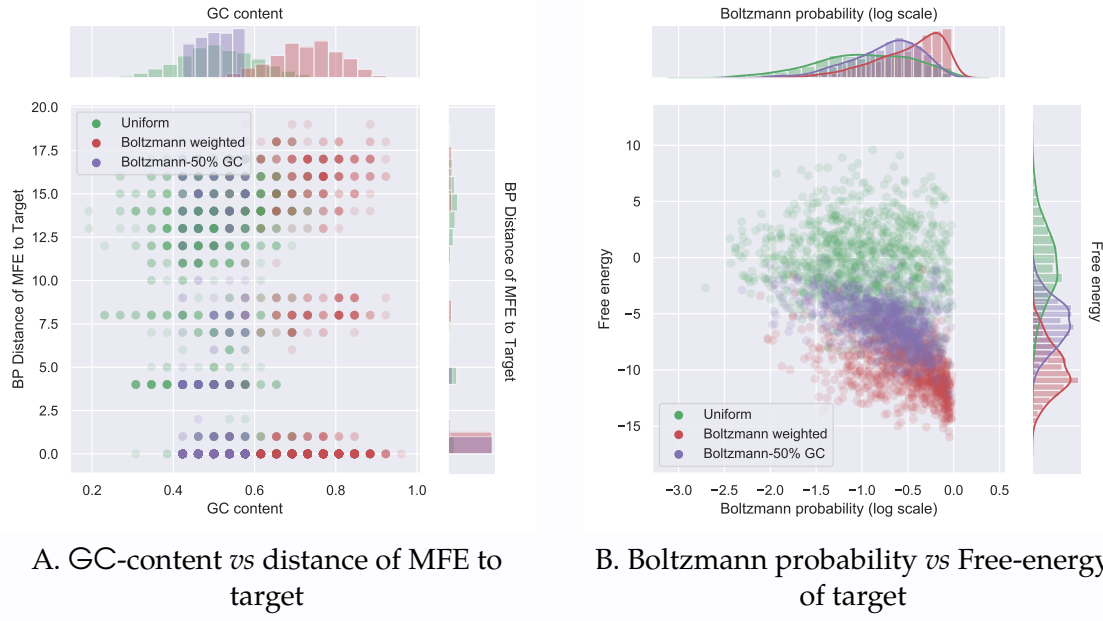


Figure 3.4: **Influence of targeted GC-content on characteristics of seeds.** For the structure of Figure 3.3, IncARNation [160] is used to produce Boltzmann-weighted sequences, constrained to feature ~50% GC content (blue), comparable to the one induced by a uniform random generation (green). GC-constrained sequence achieve much better free-energies and Boltzmann probabilities than uniformly-distributed sequences, while

### 3.3.2 Avoiding and forcing motifs (210)

This weighted sampling strategy can also be extended to accommodate further constraints, such as collections of **forbidden** ( $\mathcal{F}$ ) and **mandatory** ( $\mathcal{M}$ ) motifs.

#### Problem 12 (Motifs-constrained dual partition function):

**Input:** Two sets  $\mathcal{F}, \mathcal{M} \subset \Sigma^*$ , respectively representing **forbidden** and **mandatory** sequence motifs; target structure  $S \in \mathcal{S}_n$

**Output:** The motifs-constrained dual partition function:

$$\mathcal{Z}_{S^*, \mathcal{F}, \mathcal{M}} = \sum_{\substack{w \text{ compatible with } S^* \\ \text{s. t. } f \notin w, \forall f \in \mathcal{F} \\ \text{and } m \in w, \forall m \in \mathcal{M}}} e^{-\beta \cdot E(w, S^*)}$$

This problem can be solved by constructing the deterministic finite-state automaton  $\mathcal{A}$  that recognizes words featuring all elements of  $\mathcal{M}$ , and none of  $\mathcal{F}$ .  $\mathcal{A}$  is built from specialized versions  $\mathcal{A}_{\mathcal{M}'}$  of the Aho-Corasick automaton [2], which we call **components**, for all sets  $\mathcal{F} \cup \mathcal{M}' \subseteq \mathcal{M}$ . Components are then collated, by redirecting any terminal state  $m \in \mathcal{M}'$  of  $\mathcal{A}_{\mathcal{M}'}$  towards a suitable state in  $\mathcal{A}_{\mathcal{M}' \setminus \{m\}}$ . Thus, starting from  $\mathcal{A}_{\mathcal{M}}$  and reading a word  $w$ , one ends up in the component  $\mathcal{A}'_{\mathcal{M}'}$ , where  $\mathcal{M} \setminus \mathcal{M}'$  is exactly the set

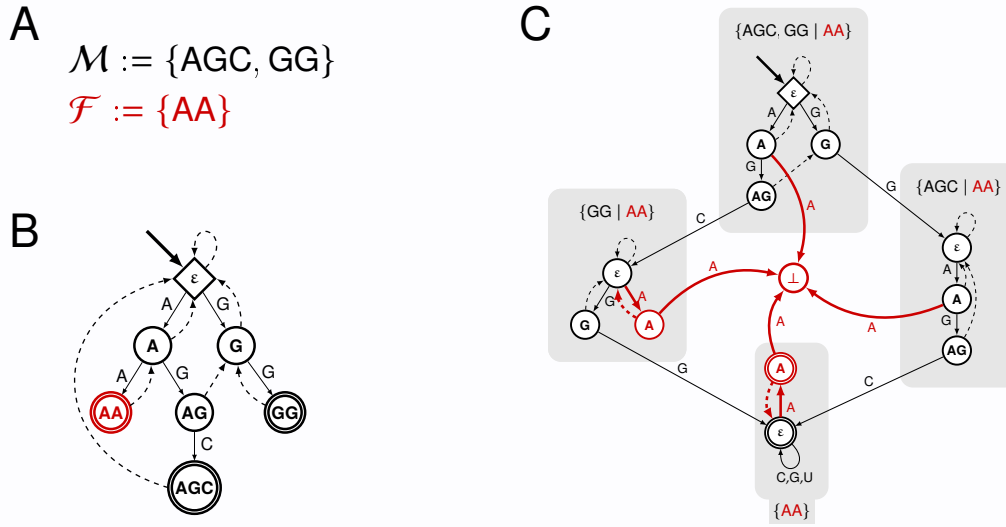


Figure 3.5: **Construction of the automaton for sequence-constrained positive design.** From lists of forbidden ( $\mathcal{F}$ ) and mandatory ( $\mathcal{M}$ ) words (A), the Aho-Corasick automaton (B) can be built in time linear on the accumulated length of words. The full automaton (C) ensures the presence of all words in  $\mathcal{M}$ , by constructing **components**, i.e. AC automata for  $\mathcal{F}$  and subsets  $\mathcal{M}' \subseteq \mathcal{M}$  of yet-to-be-generated words. Transitions between components are defined to account for new elements of  $\mathcal{F}$ , and the only accepting states are those of the component  $\mathcal{M}' := \emptyset$ . Finally, any terminal state in (B) associated with a word in  $\mathcal{F}$  is rerouted to a ground/*garbage* state  $\perp$ .

of mandatory words in  $w$ . Additionally, all final states associated with motifs in  $\mathcal{F}$  are rerouted to a non-accepting ground state. The whole process is further explained and illustrated by Figure 3.5.

The number of states  $|\mathcal{A}|$  of  $\mathcal{A}$  grows like  $\Theta(2^{|\mathcal{M}|} \sum_{m \in \mathcal{F} \cup \mathcal{M}} |m|)$ . The exponential dependency on  $|\mathcal{M}|$  seems, in some sense, unavoidable. Indeed, R  ih   and Ukkonen [162] have shown the NP-hardness of deciding whether or not all motifs of a collection  $\mathcal{M}$  ( $\mathcal{F} = \emptyset$ ) can be represented within a string of length  $n$ .

Then, the progression within  $\mathcal{A}$  can be simulated within the dynamic programming scheme through explicit convolutions of the state space. We give an example of such a construct in the example below.

#### Example 11: Dual-partition function and constraints

In a classic **base-pairing energy model**, assigning a free-energy contribution  $E_{a,b}$  to a base pair involving nucleotides  $a$  and  $b$ , the dual partition function for a structure  $S^*$  is computed for any interval  $[i, j]$  as:

$$\forall i, j \in [1, n] : \mathcal{Z}_{i,j} = \begin{cases} 1 & \text{if } i > j \\ \sum_{a \in \{A, U, C, G\}} \mathcal{Z}_{i+1, j} & \text{if } i \text{ unpaired in } S^* \\ \sum_{(a,b) \in \mathcal{B}} e^{-\beta \cdot E_{a,b}} \times \mathcal{Z}_{i+1, k+1} \times \mathcal{Z}_{k+1, j} & \text{if } i \text{ paired to } k \text{ in } S^* \end{cases}$$

and  $\mathcal{Z}_{S^*} := \mathcal{Z}_{1,n}$ . Moreover, starting from  $[1, n]$ , only a linear number of subintervals  $[i, j] \subset [1, n]$

is reached by the computation, so the recurrence can be computed in linear-time using dynamic programming.

Then we consider the adjunction of sequence constraints, forcing the presence/absence of motifs  $\mathcal{M}/\mathcal{F}$  modeled by an automaton  $\mathcal{A}$  with state space  $Q$ , transition function  $\delta$  and final states  $F \subset Q$ . It can be achieved by convolving the dynamic-programming schemes and the automaton  $\mathcal{A}$ , essentially simulating its transitions. For all pair of states  $q, q' \in Q$  and indices  $i, j \in [1, n]$ , one has :

$$Z_{i,j}^{q \rightarrow q'} = \begin{cases} 0 & \text{if } i > j \text{ and } q \neq q' \\ 1 & \text{if } i > j \text{ and } q = q' \\ \sum_{a \in \{A, U, C, G\}} Z_{i+1,j}^{\delta(q,a) \rightarrow q'} & \text{if } i \text{ unpaired in } S^* \\ \sum_{(a,b) \in \mathcal{B}} \sum_{q'' \in Q} e^{-\beta \cdot E_{a,b}} \times Z_{i+1,k+1}^{\delta(q,a) \rightarrow q''} \times Z_{k+1,j}^{\delta(q'',b) \rightarrow q'} & \text{if } i \text{ paired to } k \text{ in } S^* \end{cases}$$

where  $Z_{i,j}^{q \rightarrow q'}$  represents the dual partition function, restricted to words  $w$  that are: i) compatible with the interval  $[i, j]$  in  $S^*$ ; and ii) starting from a state  $q$ , one ends in  $q'$  upon reading  $w$  in  $\mathcal{A}$ .

### 3.3.3 Multiple structures and extended design principles (88, 90)

In the previous section, the computation of the dual partition function was greatly helped by the tree-like nature of the secondary structure, inducing recurrences at the origin of our dynamic-programming schemes. The presence of complex networks of constraints, for instance induced by **multiple secondary structures** for multistable RNAs [65] or to prevent the extension of helices [91], breaks the substructure property induced by a single secondary structure, and impedes the design of efficient algorithms.

In collaboration with Stefan Hammer, Wei Wang and Sebastian Will [88, 90], we considered a generalized version of the dual partition function, where multiple target structures  $S^*, S^* \dots S_m^*$  are jointly considered, and the energy of a sequence is additively defined on a collection of functions  $(F_1, F_2, \dots, F_k)$ , called **contribution**. Each contribution  $F$  **observes** a set  $\text{dep}(F) \subset [1, n]$  of positions in the designed sequence, and returns a numerical values accordingly. In other words, each contribution is a function  $F : \Sigma^{|\text{dep}(F)|} \rightarrow \mathbb{R}$ , whose values accumulate to define the **energy** of any given sequence.

#### Problem 13 (General dual partition function):

**Input:** Length  $n$ ; Collection of **contributions**  $\mathbf{F} := [F_1, F_2, \dots, F_k]$ ; Collection of **target structures**  $\mathbf{S}^* := [S_1^*, S_2^* \dots S_m^*]$ .

**Output:** The **generalized dual partition function**:

$$Z_{\mathbf{S}^*, \mathbf{F}} = \sum_{\substack{w \in \Sigma^n \text{ such that} \\ w \text{ comp. with } S^*, \forall S^* \in \mathbf{S}^*}} e^{-\beta \cdot \mathbf{F}(w)}$$

where  $\mathbf{F}(w) = \sum_{i=1}^k F_i([w_p \mid p \in \text{dep}(F_i)])$ .



### 3.3.3.1 Complexity aspects

Complexity-wise, the problem is generally NP-hard, as it specializes into a variant of negative design studied by, and proven NP-hard, by Hellmuth et al. [91]. Namely, the authors introduced **extended shapes**, possibly described as secondary structures whose base-pairs are partitioned into **regular** base pairs, which should be populated by valid pairs of nucleotides, and **pseudo-edges** which should not be pairable. Given several such extended shapes, the **Min-Edge-Deletion in Shapes (MES) problem** asks for the minimal number of (regular or pseudo) edges to delete before a satisfying sequence assignment can be found, i.e. such that nucleotides on regular edges  $(i, j)$  (resp. pseudo edges  $(\tilde{i}, \tilde{j})$ ) forming valid pairs in  $\mathcal{B}$  (resp. invalid pairs in  $\mathcal{U} := \{AA, CC, GG, UU\}$ ). This problem essentially constitutes a specialization of the above problem, obtained by: i) setting  $\mathbf{S}^* := \emptyset$ ; ii) Choosing suitable contributions  $F_{(i,j)}(a, b) = -1_{(a,b) \in \mathcal{B}}$  and  $F_{(\tilde{i}, \tilde{j})}(a, b) = -1_{(a,b) \in \mathcal{U}}$ . In this setting,  $Z_{\emptyset, \mathbf{F}} = \sum_{\mathbf{w}} e^{-\beta \cdot \# \text{DelEdges}(\mathbf{w})}$ , and the existence of a sequence violating  $k$  edges can be decided from the dual partition function in polynomial time (e.g. through interpolation, or increasing  $\beta$  so that co-optimal overwhelm the partition function).

Moreover, in a *uniform setting* ( $\mathbf{F} := \emptyset$ ), a key object is the **dependency graph**  $G_{\mathbf{S}}$  of a collection of structures  $\mathbf{S}^* := [S_1^*, \dots, S_m^*]$  of length  $n$ , defined as

$$G_{\mathbf{S}} := ([1, n], \cup_{i=1}^m S_i).$$

This graph is simply the union of the graphs of secondary structures, and must be **bipartite** for the collection to admit any design [73].

**Theorem 5 (Independent set and multidesign):** *Given several secondary structures  $\mathbf{S}^* := [S_1^*, \dots, S_m^*]$  such that  $G_{\mathbf{S}}$  is **bipartite**, the **dual partition function**  $Z_{\mathbf{S}^*, \emptyset}$  is exactly twice the number of independent sets in  $G_{\mathbf{S}}$ .*

It follows that computing the general partition function is as hard as **counting the number of independent sets in a bipartite graph (#BIS)**, a thoroughly-studied #P-hard problem [79].

Finally, we described in Hammer et al. [88, 90] a general **Fixed Parameter Tractable** solution. The entire network of contributions is jointly considered, and automatically transformed into a set of DP equations through a **tree decomposition**. Resulting DP schemes contrast with most of the schemes described in this document due to the automated nature of their design, but also due to their optimality in some well-defined sense.

## Chapter 4

# Constrained random generation through rejection and beyond

Generic approaches for the random generation of combinatorial objects rely on the existence of a specification for the combinatorial class. However, the design of a specification for a given combinatorial class can be a highly complex task, or altogether be provably impossible for a given a restricted set of constructors. In the case of combinatorial classes that are naturally expressible as languages, one can use classic pumping-lemma to rule out the existence of regular or context-free specifications. However, such results do not preclude the existence of a grammar for a language, which itself would be in bijection with the language of interest, as shown in the following example.

### Example 12: Copy language

Consider the 3-copy language

$$L_{3C} = \{a^n.b^n.c^n \mid n \in \mathbb{N}\}$$

A textbook application of the pumping lemma (algebraic version) shows that the language is not context-free: there is no context-free grammar that generates  $L_{3C}$ . However, the language

$$L_3 = \{(a.b.c)^n \mid n \in \mathbb{N}\}$$

can be generated using a context-free grammar (or even a simpler finite-state automaton), and admits a trivial bijection with  $L_{3C}$ . Such a bijection may simply push occurrences of  $a$  (resp.  $b$  and  $c$ ) to the beginning (resp. ending and middle), preserving the relative order within occurrences of a given letter. This transform can be easily reversible, and can be implemented in linear time.

Combinatorial properties of languages can be used to rule out any such workaround and, more generally, establish the unsuitability of classes of grammars for the generation of some languages.

For instance, Flajolet [67] showed that some context-free languages are **intrinsically ambiguous**, i.e. any grammar that generates such a language must be ambiguous. Indeed, languages generated from a context-free grammar are associated with ordinary generating functions that are algebraic. However, certain context-free languages are known

to admit transcendental generating functions, and thus cannot admit an unambiguous context-free grammar, as shown in Example 13.

**Example 13: A simple intrinsically ambiguous language<sup>a</sup>**

Consider the language

$$L_{IA} = \{w \in \{a, b, c\}^* \mid |w|_a \neq |w|_b \text{ or } |w|_b \neq |w|_c\}.$$

It is generated by a context-free grammar (axiom  $S$ )  $S \rightarrow T_{a<b} \mid T_{b<a} \mid T_{b<c} \mid T_{c<b}$ , which breaks down the language into four (overlapping, thus inducing an ambiguous grammar) sub languages that are context free, and whose union covers all possible cases. For instance, the non-terminal  $T_{a<b}$  is tasked with generating the language

$$L_{a<b} = \{w \in \{a, b, c\}^* \mid |w|_a < |w|_b\}$$

using the following set of production rules

$$\begin{aligned} T_{a<b} &\rightarrow a T_{a=b} b T_{a<b} \mid b T_{a=b} a T_{a<b} \mid b T_{a<b} \mid b T_{a=b} \mid c T_{a<b} \\ T_{a=b} &\rightarrow a T_{a=b} b T_{a=b} \mid b T_{a=b} a T_{a=b} \mid c T_{a=b} \mid \varepsilon \end{aligned}$$

which essentially amounts to a case distinction based on the first encountered character, reminiscent of a *first passage decomposition*. Similar productions rules can be obtained for the languages generated by  $T_{b<a}$ ,  $T_{b<c}$  and  $T_{c<b}$  respectively, establishing the context-free nature of  $L_{IA}$ .

To study the generating function of  $L_{IA}$ , remark that  $L_{IA} = \{a, b, c\}^* \setminus L_{\overline{IA}}$ , where

$$L_{\overline{IA}} = \{w \in \{a, b, c\}^* \mid |w|_a = |w|_b = |w|_c\}$$

. The ordinary generating functions  $S_{IA}(z)$  and  $S_{\overline{IA}}(z)$  are thus related through

$$S_{IA}(z) = \frac{1}{1-3z} - S_{\overline{IA}}(z).$$

Remind that algebraic generating functions are closed under addition/subtraction, so  $S_{IA}(z)$  is algebraic only if  $S_{\overline{IA}}(z)$  is algebraic itself. However,  $S_{\overline{IA}}(z)$  can be expressed explicitly as

$$S_{\overline{IA}}(z) = \sum_{n \geq 0} \binom{3n}{n, n, n} z^{3n}.$$

It follows from Stirling's expansion that, for  $n \equiv 0 \pmod{3}$ , one has

$$[z^n] S_{\overline{IA}}(z) = \frac{n!}{(n/3)! \times (n/3)! \times (n/3)!} = \frac{3 \times 3^n}{2\pi \times n} (1 + \mathcal{O}(1/n)).$$

Such an asymptotic expansion is not compatible with the universal asymptotic form of coefficients of rational/algebraic generating functions, as seen in the Flajolet/Sedgewick *bible* [69], or a recently refined result by Banderier and Drmota [11] (see Remark 5.1.2).

$S_{\overline{IA}}(z)$  is therefore not algebraic, and neither is  $S_{IA}(z)$ . Since generating functions associated with unambiguous context-free grammars are either rational or algebraic, we conclude that no such grammar exists, and  $L_{IA}$  is context-free, yet *intrinsically ambiguous*.

<sup>a</sup>Example borrowed from the [magisterial exposition](#) of Cyril Nicaud.

More generally, combinatorial classes associated with transcendental generating functions, regardless of their language-theoretic properties, cannot be modeled using the standard operators introduced by the symbolic method. In this chapter, we remind,

generalize and exploit versatile rejection principles for the generation of constrained combinatorial objects which are hard to specify (at least in their constrained version).

**Outline.** Section 4.1 starts by reminding the general objectives and principles of rejection methods for random generation. Section 4.2 describes the main principles behind the general technique called multidimensional Boltzmann sampling, whose application to selected problems in Bioinformatics is described in Section 4.3. Section 4.4 provides an analysis of the level of redundancy within weighted samples, and describes possible countermeasures to such an unproductive phenomenon.

The following summarizes, and attempts to unify, contributions described within the following list of articles published in journals and/or presented at conferences.

#### Associated contributions

M. Bousquet-Mélou and Y. Ponty. Culminating paths. *Discrete Mathematics and Theoretical Computer Science*, 10(2):125–152, 2008

A. Denise, Y. Ponty, and M. Termier. Controlled non uniform random generation of decomposable structures. *Theoretical Computer Science*, 411(40-42):3527–3552, 2010

O. Bodini and Y. Ponty. Multi-dimensional Boltzmann Sampling of Languages. In *AOFA 2010*, volume AM of *DMTCS Proceedings*, pages 49–64, Vienna, Austria, June 2010. Discrete Mathematics and Theoretical Computer Science

D. Gardy and Y. Ponty. Weighted random generation of context-free languages: Analysis of collisions in random urn occupancy models. In *GASCOM 2010*, page 14pp, Montréal, Canada, Sept. 2010. LACIM, UQAM

J. Waldspühl and Y. Ponty. An unbiased adaptive sampling algorithm for the exploration of RNA mutational landscapes under evolutionary pressure. *Journal of Computational Biology*, 18(11):1465–79, Nov. 2011

↑ J. Waldspühl and Y. Ponty. An unbiased adaptive sampling algorithm for the exploration of RNA mutational landscapes under evolutionary pressure. In *RECOMB 2011*, volume 6577 of *Lecture Notes in Computer Science*, pages 501–515, Vancouver, Canada, Mar. 2011. Springer Berlin / Heidelberg

J. Du Boisberranger, D. Gardy, and Y. Ponty. The weighted words collector. In *AOFA 2012*, volume AQ, pages 243–264, Montreal, Canada, June 2012. DMTCS

C. Banderier, O. Bodini, Y. Ponty, and H. Tafat. On the diversity of pattern distributions in rational language. In *ANALCO 2012*, pages 107–116, Kyoto, Japan, Jan. 2012. Omnipress

A. Lorenz and Y. Ponty. Non-redundant random generation algorithms for weighted context-free languages. *Theoretical Computer Science*, 502:177–194, Sept. 2013

↑ Y. Ponty. Non-redundant random generation from weighted context-free languages. In *GASCOM 2008*, page 12pp, Bibbiena, Italy, 2008

V. Reinharz, Y. Ponty, and J. Waldspühl. A weighted sampling algorithm for the design of RNA sequences with targeted secondary structure and nucleotide distribution. *Bioinformatics*, 29(13):i308–15, July 2013

↑ V. Reinharz, Y. Ponty, and J. Waldspühl. A weighted sampling algorithm for the design of RNA sequences with targeted secondary structure and nucleotides distribution. In *ISMB/ECCB 2013*, Berlin, Germany, July 2013

Y. Zhang, Y. Ponty, M. Blanchette, E. Lecuyer, and J. Waldspühl. SPARCS: a web server to analyze (un)structured regions in coding RNA sequences. *Nucleic Acids Research*, 41(Web Server issue):W480–5, July 2013

J. Lumbroso, M. Mishna, and Y. Ponty. Taming reluctant random walks in the positive quadrant. In *GASCOM 2016*, volume 59 of *Electronic Notes in Discrete Mathematics*, pages 99 – 114, Bastia, France, 2017

J. Michálik, H. Touzet, and Y. Ponty. Efficient approximations of RNA kinetics landscape using non-redundant sampling. In **ISMB/ECCB 2017**, volume 33, pages i283 – i292, Prague, Czech Republic, 2017

S. Hammer, W. Wang, S. Will, and Y. Ponty. Fixed-parameter tractable sampling for RNA design with multiple target structures. **BMC Bioinformatics**, 20(1):209, Dec. 2019

↑ S. Hammer, Y. Ponty, W. Wang, and S. Will. Fixed-Parameter Tractable Sampling for RNA Design with Multiple Target Structures. In **RECOMB 2018**, Paris, France, 2018

## 4.1 It's not you, it's me: The unreasonable power of rejection

Before we move any further, let us define precisely the problem of random generation. We are tasked with the random generation of objects of size  $n$  within a *challenging* class  $\mathcal{A}$  (i.e. hard to specify, e.g. using a grammar), according to a predefined probability distribution  $p_w^*$ .

### Problem 14 (Random generation for $\mathcal{A}$ ):

**Input:** Number  $m$  of objects

**Output:** Sequence of  $[w_1, \dots, w_m] \in \mathcal{A}^m$  such that

$$\mathbb{P}_{\mathcal{A}}(W = w) = p_w^*.$$

The provable inexistence of a specification can be frequently circumvented at a reasonable cost through a **rejection strategy**. Its main idea is to consider another *simpler* set  $\mathcal{B}$ , chosen to enforce the following properties:

1. **Correctness:**  $\mathcal{A} \subseteq \mathcal{B}$  (or  $\mathcal{A} \subseteq \mathcal{B}'$ , admitting a computable bijection with  $\mathcal{B}$ );
2. **Effectiveness:** Objects from  $\mathcal{B}$  can be generated (uniformly) at random;
3. **Efficiency:** The distribution over  $\mathcal{B}$  is *reasonably close* to its target over  $\mathcal{A}$ .

In this setting, the rejection algorithm consists in an iterated generation of objects from a superset  $\mathcal{B} \supseteq \mathcal{A}$ , discarding any object in  $\mathcal{B} \setminus \mathcal{A}$ , until an object of  $\mathcal{A}$  is obtained and returned.

The subsequent rejections act as a conditioning, and the **emission probability**  $p_w$  of a given object  $w \in \mathcal{A}$  simply becomes

$$p_w = \mathbb{P}_{\mathcal{B}}(W = w \mid W \in \mathcal{A}) = \frac{\mathbb{P}_{\mathcal{B}}(W = w)}{\mathbb{P}_{\mathcal{B}}(W \in \mathcal{A})} = \frac{\mathbb{P}_{\mathcal{B}}(W = w)}{\sum_{w' \in \mathcal{A}} \mathbb{P}_{\mathcal{B}}(W = w')}$$

where  $\mathbb{P}_{\mathcal{B}}(W = w)$  indicates the probability of generating  $w$  using the above-mentioned generator for  $\mathcal{B}$ .

Typically, when a **uniform distribution**  $p_w^* = 1/|\mathcal{A}|$  is targeted, and enforced over  $\mathcal{B}$  by the initial generator, one has  $\mathbb{P}_{\mathcal{B}}(W = w) = 1/|\mathcal{B}|$ , and the emission probability simplifies into

$$p_w = \frac{1/|\mathcal{B}|}{|\mathcal{A}|/|\mathcal{B}|} = \frac{1}{|\mathcal{A}|} = p_w^*.$$

In other words, a uniform generator for  $\mathcal{B}$  readily yields a uniform generator for  $\mathcal{A}$ .

This correctness generalizes **beyond uniform distributions**, to any case where the generated and targeted probability distributions are proportional over  $\mathcal{A}$ :

$$\exists \kappa \in \mathbb{R} \text{ such that } \forall w \in \mathcal{A}, \quad \mathbb{P}_{\mathcal{B}}(W = w) = \frac{p_w^*}{\kappa} \quad (4.1)$$

where  $p_w^*$  denotes the targeted probability distribution. Again, the emission probability simplifies into

$$p_w = \frac{p_w^*/\kappa}{\sum_{w' \in \mathcal{A}} p_{w'}^*/\kappa} = p_w^*,$$

and the rejection generator enforces the targeted probability distribution.

#### 4.1.1 Correcting a (bounded) bias

A similar strategy can be used when the distribution of  $\mathcal{B}$ , renormalized over  $\mathcal{A}$  (i.e. after rejection), does not fully coincide with the targeted distribution:

$$\exists w \in \mathcal{A} \quad \text{such that} \quad p_w \neq p_w^*.$$

Denote as  $w_{\ominus}$  the object in  $\mathcal{A}$  which is most unfavorably discriminated against by the initial generation, such that

$$w_{\ominus} := \operatorname{argmin}_{w \in \mathcal{A}} \delta(w) \quad \text{with} \quad \delta(w) := \frac{p_w}{p_w^*}.$$

Then, following the ideas of [142], the rejection algorithm is coupled with an additional rejection step, accepting the generated  $w \in \mathcal{A}$  with probability

$$q_w = \frac{\delta(w_{\ominus})}{\delta(w)} \leq 1 \quad (4.2)$$

and rejecting it otherwise. The emission probability  $p'_w$  of the combined rejection is then

$$p'_w = \frac{p_w \times q_w}{\sum_{w' \in \mathcal{A}} p_{w'} \times q_{w'}} = \frac{p_w^*}{\sum_{w' \in \mathcal{A}} p_{w'}^*} = p_w^*$$

so that any generator for  $\mathcal{B}$  such that  $\mathbb{P}_{\mathcal{B}}(W = w) > 0, \forall w \in \mathcal{A}$ , can be used (potentially at the cost of frequent rejections) to generate from  $\mathcal{A}$  within the targeted distribution.

### 4.1.2 Complexity aspects

Given the existence of unbounded sequences of rejections the worst-case complexity of rejection generators is infinite. However, such scenarios are associated with abysmally small probabilities and, overall, populate a negligible proportion of the probability space, so worst-case analyses provides little insight into the concrete requirements of rejection-based methods. It is thus arguably more relevant to consider the trade-off between cost and probability, and **average-case complexities** are thus usually sought to measure the practical complexity of generators.

The main contributing factor to the average-case complexity usually resides in the repeated (discarded) calls to the generator for  $\mathcal{B}$ . Denote by  $T$  the **number of attempts**, i.e. the number of time the generator for  $\mathcal{B}$  must be called before generating an object from the targeted distribution over  $\mathcal{A}$ , then we **generally** have

$$\begin{aligned} \mathbb{E}(T) &= \frac{1}{\sum_{w'' \in \mathcal{A}} \mathbb{P}_{\mathcal{B}}(W = w'') \times q_{w''}} \\ &\leq \frac{1}{\mathbb{P}_{\mathcal{B}}(W \in \mathcal{A})} \times \frac{p_{w_{\ominus}}^*}{\mathbb{P}_{\mathcal{B}}(W = w_{\ominus})} \times \frac{p_{w_{\oplus}}^*}{\mathbb{P}_{\mathcal{B}}(W = w_{\oplus})}, \end{aligned}$$

with  $w_{\oplus} = \max_{w \in \mathcal{A}} \delta(w)$  being the object whose emission probability is the most positively biased in comparison to the targeted distribution.

Note that, when the **uniform distribution** is targeted over  $\mathcal{A}$ , and exactly enforced over  $\mathcal{B}$ , we have  $q_w = 1, \forall w \in \mathcal{A}$ , and the expected number of attempts greatly simplifies into

$$\mathbb{E}(T) = \frac{|\mathcal{B}|}{|\mathcal{A}|}.$$

#### Example 14: 2D walks with small steps (122)

Many types of 2D walks are associated with non-algebraic generating functions when confined to the positive quadrant.

For instance, Mishna and Rechnitzer [137] showed that the generating function of positive 2D walks taking steps in  $\mathcal{S} = \{\nearrow, \searrow, \nearrow\}$  is not D-finite and, in particular, non-algebraic. However, the number  $w_n$  of such walks taking  $n$  steps admits an asymptotic equivalent in

$$c_n = \kappa 3^n + \mathcal{O}(\sqrt{8}^n)$$

where  $\kappa = 0.173 \dots$  is a computable constant.

Now consider the set of unconstrained walks over  $\{\nearrow, \searrow, \nearrow\}^n$ , consisting of exactly  $3^n$  objects. A trivial algorithm for the uniform generation simply chooses  $n$  steps, each one uniformly at random.

Clearly, the set of unconstrained walks (set  $\mathcal{B}$ ) includes the set of positive 2D walks (set  $\mathcal{A}$ ). A rejection algorithm can then be envisioned, generating unconstrained walks from  $\mathcal{B}$  until a positive walks from  $\mathcal{A}$  is found. The expected number of attempts from this rejection algorithm is then asymptotically

equivalent to

$$\mathbb{E}(T) = \frac{|\mathcal{B}|}{|\mathcal{A}|} = \frac{3^n}{\kappa 3^n} \approx 5.77 \dots$$

an, in particular, does not grow with  $n$ , leading to a linear-time rejection algorithm.

With Marni Mishna and Jérémie Lumbroso, we generalized this strategy [122] to more complex small steps set  $\mathcal{S} \subset [-1, 0, 1] \times [-1, 0, 1]$ , crucially allowing for an efficient generation in cases of **reluctant walks**, i.e. walks having a natural attraction, called **drift**, towards the axes. This tendency to violate the positivity condition in turn leads to a number of positive walks exponentially smaller than  $|\mathcal{S}|^n$ , thus rejecting from unconstrained walks would yield an exponential expected time algorithm.

However, it as been shown in Duraj [61] that the number  $w_n$  of positive 2D walks is such that  $w_n \in \Theta(\alpha^n n^\beta)$ , for two constants  $\alpha < |\mathcal{S}|$  and  $\beta$  that only depend on the step set  $\mathcal{S}$ . Moreover, this exponential growth constant  $\alpha$  is shared by a well-chosen half-plane, including the positive quarter-plane, whose slope  $m$  can be determined numerically (or even exactly). When such a slope  $m$  is rational, then the projection of the steps onto the chosen-plane is a rational number, and efficient random generation algorithms for 1D positive walks can be used to generate walks in the half plane. Those walks are then lifted onto the 2D plane, and an additional rejection step ensures positivity at the 2D level.

Since the exponential growth of the 1D and 2D walks coincide, then the rejection only contributes an additional polynomial factor. In other words, the algorithm that generates 1D positive walks and rejects those that escape the 2D positive quadrant, has expected time per generation bounded by a polynomial in  $n$ .

Finally, we consider the **computational complexity**, expressed in an arbitrary unit (#arithmetic operations, bit-complexity [49]), and denoted as a random variable  $C$ .

In simple – yet frequent – cases, objects of a fixed size  $n$  are generated, and the time complexity  $c(w)$  required by the generation of an object can be seen as deterministic and only depending on  $n$ . Denoting by  $c(n)$  the complexity of generating objects in  $\mathcal{B}$ , one gets

$$\mathbb{E}(C) = \frac{c(n)}{\mathbb{P}_{\mathcal{B}}(W \in \mathcal{A})} \quad (4.3)$$

More generally, the average-case complexity of the rejection algorithm follows

$$\mathbb{E}(C) = \frac{\mathbb{E}_{\mathcal{B}}(c(W))}{\mathbb{P}_{\mathcal{B}}(W \in \mathcal{A})} \quad (4.4)$$

where  $W$  is a random object in  $\mathcal{B}$ . This formula also holds when the expected cost of generating objects in  $\mathcal{A}$  and  $\mathcal{B} \setminus \mathcal{A}$  differ, owing to the additivity, in expectation, of (dependent) random variables.

#### Example 15: Anticipated rejection/Florentine algorithm for Motzkin walks (13)

An example of such an analysis can be found in the generation of **prefixes of Motzkin walks**, i.e. random positive paths using  $\nearrow$ ,  $\rightarrow$  and  $\searrow$  steps, associate to 2D vectors  $(+1, +1)$ ,  $(+1, 0)$  and  $(+1, -1)$  respectively. The textbook algorithm, proposed by Barucci, Pinzani and Sprugnoli [13], uses an efficient **anticipated rejection** scheme. Its algorithmic part is straightforward, and simply consists, starting from 0, of choosing  $\uparrow$ ,  $\rightarrow$  or  $\downarrow$  with equal probability at each stage, rejecting the prefix as soon



as a negative ordinate is reached, until a requested length  $n$  is reached.

The analysis of its average-case complexity is slightly more involved. The expected number of attempts depends on the asymptotics of the numbers of positive paths ( $\mathcal{A}_n$ ) and unconstrained paths ( $\mathcal{B}_n$ ) respectively. While we clearly have  $|\mathcal{B}_n| = 3^n$ , it is a folklore result that  $|\mathcal{A}_n| \in \Theta\left(\frac{3^n}{\sqrt{n}}\right)$ , so the probability of accepting an object is

$$\mathbb{P}_{\mathcal{B}}(W \in \mathcal{A}) = \frac{|\mathcal{A}|_n}{|\mathcal{B}|_n} \in \Theta(1/\sqrt{n}).$$

The expected number of attempts thus grows in  $\Theta(\sqrt{n})$  and a classic rejection scheme, generating a  $n$  step unconstrained walk followed by a positivity test, would yield an overall complexity in  $\Theta(n\sqrt{n})$ .

Anticipated rejection has a sizable impact on the expected generation cost, but its analysis cost is slightly more involved. Denote by  $b_n$  the number of positive walks of length  $n$ , and by  $d_n$  the number of positive walks ending in  $(n, 0)$ . Being rejected before reaching  $n$  steps means having a positive prefix ending at  $(k, 0)$  of size  $k < n$  steps, followed by a  $\searrow$  step, with a cost of  $k$  rejections being incurred by such a rejection. The expected cost of a generation is therefore given by

$$\mathbb{E}(c(W)) = \sum_{k < n} k \frac{d_{k-1}}{3^k} + n \frac{b_n}{3^n},$$

where the first term denotes the expected cost of (anticipated) rejections, and the second term denoting the contribution of positive paths. It then follows from

$$d_n \in \Theta\left(\frac{3^n}{n\sqrt{n}}\right) \quad \text{and} \quad b_n \in \Theta\left(\frac{3^n}{\sqrt{n}}\right),$$

that the first term is bounded by a constant, while the second grows like  $\Theta(\sqrt{n})$ , and thus

$$\mathbb{E}(c(W)) \in \Theta(\sqrt{n}).$$

A direct application of (4.4) allows us to conclude that the overall average-case complexity is  $\Theta(n)$ . This result is optimal (up to constants) since the number of bits required despite an expected  $\Theta(\sqrt{n})$  number of attempts.

A similar reasoning, and complexity speedup, more generally holds for a very large class of combinatorial structures obtained by step-wise extensions, such that the extension of a rejected object is itself rejected, as shown by Denise [48].

#### Example 16: Culminating paths [21]

Using anticipated rejection, to spare the cost of rejecting a fully-generated object, was crucial in the design, with Mireille Bousquet-Mélou, of an efficient algorithm for the uniform generation of **culminating paths** [21]. A culminating path is a unidimensional walk which takes steps  $+1$  and  $-1$ , starts at position 0, and must both: 1) remain positive; and 2) be culminating, i.e. reaches its maximum height on its final step.

Now consider the **mirror image** of a positive walk, obtained by reversing its sequence of steps, and note that it is now culminating (but may not be positive). It follows that any culminating path  $w$  of length  $n$  can be factored into  $w = w^\perp \cdot w^\top$ , where  $w^\perp$  is a positive path of size  $\lceil n/2 \rceil$ , and  $w^\top$  is the mirror image of some positive path of length  $\lfloor n/2 \rfloor$ . In other words, the set  $\mathcal{B}_n := \mathcal{W}_{\lceil n/2 \rceil}^\perp \times \mathcal{W}_{\lfloor n/2 \rfloor}^\top$ , where  $\mathcal{W}_n^\perp$  (resp.  $\mathcal{W}_n^\top$ ) is the set of positive (resp. culminating) walks of length  $n$ , represents a superset of  $\mathcal{A}_n$  the set of culminating walks of length  $n$ .

Using the kernel method, we could establish that  $|\mathcal{A}_n| \in \Theta(2^n/n)$  [21], and it is a folklore result that  $|\mathcal{W}_m^\perp| = |\mathcal{W}_m^\top| \in \Theta(2^n/\sqrt{n})$ , so that the expected number of attempts is asymptotically

$$\mathbb{E}(T) = \frac{|\mathcal{W}_{\lceil n/2 \rceil}^\perp| \times |\mathcal{W}_{\lfloor n/2 \rfloor}^\top|}{|\mathcal{A}_n|} \in \Theta(1).$$

Moreover, the generation of a positive walk, or its reversal can be done in linear expected time using anticipated rejection. Overall, we get that the simple algorithm that generates two half walks, reverses the second one, and glues them together, restarting until the result is a culminating path, has linear average-case complexity.

## 4.2 Rejecting in dimension one and then some...

### 4.2.1 (Combinatorial) Boltzmann sampling

**Boltzmann sampling** [60] can be seen as a special instance of rejection sampling. It revisits the random generation of combinatorial objects of a given size  $n$  described by a specification  $B$ . Instead of precomputing the number of objects accessible through local alternatives using costly convolution products, it initially relaxes the size constraint, drawing objects of any size in  $\mathcal{B}$  until an object of suitable size is found.

More precisely, a **Boltzmann generator** emits an object  $\omega \in \mathcal{B}$  with probability

$$\mathbb{P}_x(W = \omega) = \frac{x^{|\omega|}}{B(x)} \quad (4.5)$$

where  $B(z)$  is the generating function of  $\mathcal{B}$  and  $x$  is a real-valued **Boltzmann parameter**. Such a distribution can be seen as a special instance of the Boltzmann distribution in statistical mechanics, where the free-energy of an object  $\omega \in \mathcal{B}$  is arbitrarily set to  $E(\omega) := -kT \log x^{|\omega|}$ . This leads to associating a Boltzmann factor of  $e^{E(\omega)/kT} = x^{|\omega|}$  to each object, in which one recognizes the distribution of Equation (4.5).

A careful tuning of  $x$  ensures a *reasonable* probability of acceptance and, in turn, an average-case complexity scaling in  $\mathcal{O}(n^2)$  for generating an object of size  $n$  (generating an object of approximate size in  $[(1 - \varepsilon) \cdot n, (1 + \varepsilon) \cdot n]$  can be performed in  $\Theta(n)$  time).

The appeal of Boltzmann samplers to a large subset of the Analysis of Algorithms (AOFA) community, can be attributed to two main factors:

- The elegance, and relative simplicity, of its algorithmic aspects: As shown in Table 4.1, generators for Cartesian products are deceptively simple to express. Their implementation, however, requires an evaluation of the **Boltzmann Oracle**, the value of the generating functions. While this task is relatively trivial when a closed form (or, generally, computable) formula is known for the generating function, its

Rule type	Generating function	Boltzmann generator $\Gamma V_x$
Product $v \rightarrow a.b$	$V(z) = A(z) \times B(z)$	<b>Return</b> $\Gamma A_x . \Gamma B_x$
Union $v \rightarrow a, v \rightarrow b$	$V(z) = A(z) + B(z)$	<b>If</b> Bernoulli( $A(x)/V(x)$ ) : <b>Return</b> $\Gamma A_x$ <b>Else</b> : <b>Return</b> $\Gamma B_x$
Terminal $v \rightarrow t$	$V(z) = z$	<b>Return</b> $t$
Epsilon $v \rightarrow \varepsilon$	$V(z) = 1$	<b>Return</b> $\varepsilon$

Table 4.1: **Uniform Boltzmann generators for context-free languages.**

automation for general specifications is more involved, yet possible using a mixture of formal calculus and numerical iterations [145]). The optimization of the Boltzmann parameter can also be addressed in its full generality using efficient numerical iterations[144];

- Its strong connection with analytic combinatorics: The time complexity of Boltzmann samplers is strongly related to the distribution of lengths under the Boltzmann distribution. The design of Boltzmann samplers thus motivates extensions of the symbolic method to complex operators (e.g. box-product, shuffle product. . . ) that lead to generic random generation principles, and to deeper characterizations of the rich singularities that arise from those operators.

## 4.2.2 Multidimensional Boltzmann sampling

In collaboration with Olivier Bodini [18], we introduced<sup>1</sup> and studied an extension of Boltzmann sampling, dubbed **multi-dimensional Boltzmann sampling**. This technique addresses cases where one wishes to express cardinality constraints on distinguished atoms/letters (or combination thereof). For instance, given a language  $L$  over an alphabet  $\Sigma := \{a, b, c\}$ , one may be interested in the subset of  $L$  where the number occurrences of  $a, b$  and  $c$  are equal, reducible to a pair of constraints  $\{a = n/3; b = n/3\}$ . Enforcing such constraints generally escapes the expressiveness of classic operators of the symbolic method, preventing the definition of a specification and, in turn, the use of generic generators.

### 4.2.2.1 General principle

The idea of generalizing the Boltzmann rejection scheme to higher dimensions stems from the observation that additive parameters (e.g. used to capture the #occurrence of atoms/letters, #use of production rules. . . ) occurring in combinatorial specifications tend to asymptotically follow normal distributions. Moreover, those distributions have

<sup>1</sup>This natural extension was already discussed in the conclusion of the seminal Boltzmann paper [60], yet remained unexplored for almost a decade prior to our work.

linear mean and variance. Such distributions are typically robust to the introduction of **weights**, also called **parameters** in subsequent work, which can be used to **shift the means**, thus providing a probabilistic control over the average composition of generated objects.

Once fitted to match desired expectations, a rejection algorithm performs a weighted random generation, only accepting objects with suitable composition. The concentration of the distribution usually induces a polynomial expected number of attempts, typically growing in  $\Theta(n^{k/2})$  when  $k$  is the number of cardinality constraints. Let us illustrate our general approach with a brief introductory example.

**Example 17: Multidimensional Boltzmann sampler for balanced Motzkin language**

Consider the Motzkin language  $M \subseteq \{ (, ), \bullet \}^*$  of well-parenthesized expressions interleaved with dots, formally defined as :

$$M = \{ \omega \in \{ (, ), \bullet \}^* \mid |\omega|_{(} = |\omega|_{)} \text{ and } |\omega'|_{(} \geq |\omega'|_{)}, \forall \omega' \sqsubseteq \omega \text{ (}\omega' \text{ prefix of } \omega) \}.$$

This language is context-free, and generated by the grammar:

$$S \rightarrow ( S ) S \mid \bullet S \mid \varepsilon$$

Now, consider the balanced Motzkin language  $M_{2B}$  the subset of the Motzkin words featuring exactly 50% of dots ( $\bullet$ ), i.e.

$$M_{2B} = \left\{ \omega \in M \mid (|\omega| \equiv 0 \pmod{2}) \text{ and } |\omega|_{\bullet} = \frac{|\omega|}{2} \right\}$$

Clearly,  $M_{2B}$  is not context-free. Moreover, the targeted  $1/2$  proportion of dots ( $\bullet$ ) characters differs substantially from the  $1/3$  proportion expected in large uniformly-distributed Motzkin words [50]. It follows from Drmota's theorem [56] that a naive rejection algorithm, generating words of length  $n$  from  $M$  until a word in  $M_{2B}$  is obtained, would require a number of attempts growing exponentially with  $n$ .

However, as previously described [50], a weight  $\pi \in \mathbb{R}^+$  can be associated to the dot ( $\bullet$ ), to induce a **weighted distribution** onto  $M$ , in which the emission probabilities become

$$P_{\pi}(W = \omega) = \frac{\pi^{|\omega|_{\bullet}}}{\sum_{\omega' \in M} \pi^{|\omega'|_{\bullet}}}. \quad (4.6)$$

Generating within such a distribution can be performed in time essentially  $\mathcal{O}(n \log n)$ , after a precomputation in  $\mathcal{O}(n^{1+o(1)})$ , using generic generators [50].

The expected proportion of dots then becomes a function of  $\pi$ , allowing the user to **shift the mean** by changing the value of  $\pi$ . Namely, the asymptotic proportion of dots is given by

$$\mathbb{E}_{\pi}(|W|_{\bullet}) = \frac{\pi}{2 + \pi} (1 + \mathcal{O}(1/\sqrt{n}))$$

and thus achieves the targeted  $1/2$  for  $\pi = 2$ . More importantly, it follows from the multidimensional version of Drmota's theorem [56] that, within the weighted version of  $M$ , the number of dots asymptotically follows a normal distribution with mean  $n/2$ , variance  $\sigma n$ , for some positive constant  $\sigma$ . The probability of generating a word in  $M_{2B}$  is thus in  $\Theta(1/\sqrt{n})$ .

Now, consider the rejection algorithm which repeatedly generates words in  $M$  of length  $n$  from the weighted distribution (4.6), setting  $\pi = 2$ , until a word in  $M_{2B}$  is obtained. This algorithm is uniform

within the restriction of  $M_{2B}$  to words of size  $n$ . Moreover, its expected number of attempts is in  $\Theta(\sqrt{n})$ , leading to an overall complexity in  $\Theta(n\sqrt{n} \log n)$ .

#### 4.2.2.2 Problem statement and algorithm

The problem addressed by multidimensional Boltzmann sampling is the uniform random generation of words/objects of size  $n$  satisfying:

- Specification/grammar  $C$  for a combinatorial class over atoms  $t_1, t_2, \dots$  (a.k.a. letters, or terminal symbols);
- Composition constraints of the form

$$\left\{ \frac{|W|_{t_i}}{n} = r_i \in [0, 1] \right\}_{i=1}^k \quad (4.7)$$

for a subset of  $k$  atoms, denoted  $\{t_1, \dots, t_k\}$  for the sake of convenience.

A multidimensional Boltzmann sampler [18] then proceeds in three stages:

1. Calibration of weights/parameters  $\pi_1, \pi_2, \dots$ , chosen such that

$$\left\{ \mu_i := \frac{\mathbb{E}_{\pi_1, \pi_2, \dots} (|W|_{t_i})}{n} = r_i \in [0, 1] \right\}_{i=1}^k \quad (4.8)$$

2. Random generation of objects from  $\mathcal{C}$  within the  $\pi$ -weighted distribution

$$\mathbb{P}_{\pi_1, \pi_2, \dots} (W = \omega) = \frac{\prod_{i=1}^k \pi_i^{|\omega|_{t_i}}}{[z^n] C(z; \pi_1, \pi_2, \dots)} \quad (4.9)$$

where  $C(z; u_1, u_2, \dots)$  represents the **multivariate (weighted) generating function** marking the occurrences of the atoms

$$C_\pi(z; u_1, u_2, \dots) = \sum_{w \in \mathcal{C}} z^{|w|} \prod_{i=1}^k \pi_i^{|w|_{t_i}} u_i^{|w|_{t_i}} \quad (4.10)$$

3. Rejection of objects having unsuitable composition (and size) until adequate ones are produced.

The requirement of an exact size  $n$  for the generated objects can be relaxed to accept words having more diverse lengths in  $[(1 - \varepsilon) \cdot n, (1 + \varepsilon) \cdot n]$  for some constant  $\varepsilon > 0$ . Similarly, words whose composition departs slightly from the expressed constraints, such that  $\#t_i/n \in [r_i - \varepsilon', r_i + \varepsilon']$ , can be deemed acceptable.

**Remark 4.2.1:** Note that these relaxed statements only enforce uniformity within a given size/composition class, and do not constrain the respective probabilities of two objects having different length/composition. Such a slight bias is generally deemed acceptable in contexts where statistical estimates are computed from generated structures.

Moreover those emission probabilities could, in principle, be corrected by a subsequent rejection to ensure an overall uniform distribution, as described above in Equation (4.2). However, this additional rejection would increase the expected complexity by an exponential factor, roughly equal to  $(\prod_i \max(\pi_i, 1/\pi_i))^{2\varepsilon n}$ .

**Weight calibration.** This step can be performed using two distinct approaches:

- Symbolically, it is often possible to find a simple expression relating the asymptotic proportion of each atom to the weight. For instance, in context-free languages generated by strongly-connected grammars, the dominant singularity is always of *square-root* type, and the dominant singularity  $\rho_{\pi_1, \pi_2 \dots}$  can be expressed as a function of the weights. The expected composition is then given by

$$\lim_{n \rightarrow \infty} \mathbb{E}(|W|_{t_i}/n \mid |W| = n) = -\pi_i \frac{\frac{\partial}{\partial \pi_i} \rho_{\pi_1, \pi_2 \dots}}{\rho_{\pi_1, \pi_2 \dots}}, \forall i \in [1, k].$$

This system can then be solved to obtain a closed-form formula, expressing the solution weights as a function of the targeted proportions, as shown in Example 17 and [50].

- Numerically, an iteration was initially proposed [18], whose efficient convergence was conditioned by the choice of a suitable initial weight vector.

This work was later revisited by Bendkowski, Bodini and Dovgal [15], leading to a more general and efficient solution. In their work, the authors were able to reformulate the calibration as a convex optimization problem, solved using the interior-point method of Nesterov and Nemirovskii [140] in time polynomial to the inverse of the required precision. The authors completed their study with estimates for the precision required to ensure a good acceptance probability, leading to an overall polynomial tuning of multidimensional Boltzmann sampling.

Whenever feasible, a symbolic approach is arguably more elegant and efficient, leading to a trivial algorithm for computing the weights. Unfortunately, it is easy to find examples where such a strategy involves solving systems whose complexity lies beyond the current capacities of symbolic mathematical engines (e.g. Maple, Mathemagix or Mathematica). Moreover, it does not represent a good approach for languages consisting of words of fixed size (e.g. representing the search space of a combinatorial problem for a given instance), since the associated generating functions typically have very complex close-form expressions (if any), are hard to manipulate, and have infinite radius of convergence. In such cases, numerical iterations represent the method of choice.

Rule type	Precomputation	Recursive generator $\Gamma V_n$
Product $v \rightarrow a.b$	$v_n \leftarrow \sum_{i=0}^n a_i \times b_{n-i}$	Choose $K$ with prob. $\mathbb{P}(K = k) = \frac{a_k \times b_{n-k}}{v_n}$ <b>Return</b> $\Gamma A_k . \Gamma B_{n-k}$
Union $v \rightarrow a, v \rightarrow b$	$v_n \leftarrow a_n + b_n$	<b>If</b> Bernoulli( $a_n/v_n$ ) : <b>Return</b> $\Gamma A$ <b>Else</b> : <b>Return</b> $\Gamma B_x$
Terminal $v \rightarrow t$	$v_n \leftarrow \begin{cases} \pi(t) & \text{if } n = 1 \\ 0 & \text{otherwise} \end{cases}$	<b>Return</b> $t$
Epsilon $v \rightarrow \varepsilon$	$v_n \leftarrow \begin{cases} 1 & \text{if } n = 0 \\ 0 & \text{otherwise} \end{cases}$	<b>Return</b> $\varepsilon$

Table 4.2: **Weighted generators for context-free constructors**, with  $\pi(t)$  denoting the **weight associated with a terminal letter/atom**  $t$ . Remark that, in the case of products, the convolution can be restricted to values of  $i$  in  $[1, n - 1]$  if the grammar is in  $\varepsilon$ -free form. Such a form can be assumed without loss of generality, and greatly eases the implementation by avoiding ill-defined equations, associated with non-productive circular dependencies in the precomputation.

**Weighted random generation.** Generating within a weighted distribution is arguably the easiest part of multidimensional Boltzmann sampling.

Using the **recursive method**, weights can straightforwardly incorporated in classic recurrences, allowing to precompute the weights accessible through local alternatives. Those weights can, in turn, be used to derive probabilities that induce the weighted distribution described in Equation (4.9). Table 4.2 summarizes some of the constructors used in the context of context-free languages. The computation of those recurrences can be optimized using properties of holonomic generating functions, leading to an overall complexity in  $\mathcal{O}(mn \log n + n^{1+o(1)})$  for the generation of  $m$  words of length  $n$  in the weighted distribution of Equation (4.9), as discussed in Denise *et al* [50].

**Weighted Boltzmann samplers** can also be adapted from their uniform counterparts. To that purpose, it suffices to substitute, in the generators of Table 4.1, **weighted generating functions** defined as

$$C_\pi(z) = \sum_{\omega \in \mathcal{C}} z^{|\omega|} \prod_{i=1}^k \pi_i^{|\omega|_{t_i}}, \quad (4.11)$$

instead of the ordinary ones used within classic Boltzmann sampling. As shown in Denise *et al* [50], such weighted generating functions are the solutions of systems of functional equations that is almost identical to those of Table 4.1, except for the addition of weights within **terminal rules**  $v \rightarrow t$ , namely

$$(\text{Ordinary gen. fun.}) C(z) = z \rightarrow (\text{Weighted gen. fun.}) C_\pi(z) = \pi(t) \times z \quad (4.12)$$

where  $\pi(t)$  denotes the weight of an atom  $t$ . For any given weight vector, the Boltzmann parameter calibration can

#### 4.2.2.3 Average-case complexity analysis

A usual with rejection-based samplers, the worst-case complexity is infinite, so we investigate the average-case complexity as the relevant metric to assess the efficiency of a sampler. In general, the probability to draw an object  $\omega \in \mathcal{C}$  having suitable composition

$$(|\omega|_{t_1} = n \cdot r_1, |\omega|_{t_2} = n \cdot r_2 \dots) \in \mathbb{N}^k$$

in the atoms  $(t_1, t_2 \dots)$  is given by

$$\mathbb{P}(|W|_{t_i} = n \cdot r_i, \forall i \in [1, k] \mid |W| = n) = \frac{\sum_{\substack{\omega \in \mathcal{C} \text{ s.t. } |\omega| = n, \\ |\omega|_{t_i} = r_i \cdot n, \forall i \in [1, k]}} \prod_{i=1}^k \pi_i^{|\omega|_{t_i}}}{\sum_{\substack{\text{s.t. } |\omega| = n}} \prod_{i=1}^k \pi_i^{|\omega|_{t_i}}} \quad (4.13)$$

$$= \frac{[z^n u_1^{n \cdot r_1} u_2^{n \cdot r_2} \dots] C_\pi(z; \pi_1 \cdot u_1, \pi_2 \cdot u_2 \dots)}{[z^n] C_\pi(z; \pi_1, \pi_2 \dots)} \quad (4.14)$$

where  $C_\pi(z; u_1, u_2 \dots)$  represents the multivariate generation function introduced in Equation (4.10). The expected number of attempts is simply obtained as the inverse of the probability of acceptance, *i.e.* by exchanging the numerator and denominator in Equation (4.14).

**Simple type grammars.** For **simple type grammars**, associated with **strongly connected** aperiodic specifications, the **multivariate version of the Drmota Theorem** [56] applies to weighted specifications [50], and the composition in the  $k$  elected atoms follows a  $k$ -dimensional Gaussian limiting distributions, centered on the expectation of Equation (4.7) [56]. More precisely, the coefficients of the weighted multivariate generating function admit the following asymptotic equivalent:

$$[z^n u_1^{m_1} u_2^{m_2} \dots] C_\pi(z; \pi_1 u_1, \pi_2 u_2 \dots) = \frac{\kappa_{\pi_1, \pi_2 \dots} \rho_{\pi_1, \pi_2 \dots}^{-n}}{2^{k/2} n^{k/2} n \sqrt{n}} \left( e^{-n \Xi_{\pi_1, \pi_2 \dots}} + \mathcal{O}\left(\frac{1}{\sqrt{n}}\right) \right)$$

where  $\kappa_{\pi_1, \pi_2 \dots}$  is a constant of both  $n$  and  $k$ ,  $\rho_{\pi_1, \pi_2 \dots}$  is the dominant singularity, and

$$\Xi_{\pi_1, \pi_2 \dots} := \sum_{i=1}^k \sum_{j=1}^k \delta_{\pi_1, \pi_2 \dots}^{(i,j)} \times (\mu_i - m_i/n) \times (\mu_j - m_j/n). \quad (4.15)$$

The  $\Xi_{\pi_1, \pi_2 \dots}$  term essentially grows quadratically with  $k$ , and depends both on the deviation of the requested composition to the expected one  $(\mu_1, \mu_2 \dots)$ , and of values  $\delta_{\pi_1, \pi_2 \dots}^{(i,j)}$  that are related to the variance/co-variance matrix of the system. Remark that, upon a successful clibration of the weights  $(\mu_i = r_i)$ ,  $\Xi_{\pi_1, \pi_2 \dots}$  vanishes at  $(m_i = n \cdot r_i)_{i=1}^k$  so one has

$$[z^n u_1^{n \cdot r_1} u_2^{n \cdot r_2} \dots] C_\pi(z; \pi_1 u_1, \pi_2 u_2 \dots) = \frac{\kappa_{\pi_1, \pi_2 \dots} \rho_{\pi_1, \pi_2 \dots}^{-n}}{2^{k/2} n^{k/2} n \sqrt{n}} \left( 1 + \mathcal{O}\left(\frac{1}{\sqrt{n}}\right) \right)$$



Moreover from the **Drmota-Lalley-Woods Theorem** [56, 112, 202] (see Theorem ??) we know that similar conditions lead a **square root type** singularity to dominate the ordinary generating function, inducing the following *universal* asymptotic equivalent for the coefficients

$$[z^n] C_\pi(z; \pi_1, \pi_2 \dots) = \frac{\lambda_{\pi_1, \pi_2 \dots} \rho_{\pi_1, \pi_2 \dots}^{-n}}{n\sqrt{n}} \left( 1 + \mathcal{O}\left(\frac{1}{\sqrt{n}}\right) \right).$$

The expected number of attempts probability of drawing an object of suitable composition, assuming successful calibration is then asymptotically equivalent to

$$\mathbb{E}_{\pi_1, \pi_2, \dots}(T) = \frac{[z^n] C_\pi(z; \pi_1, \pi_2 \dots)}{[z^n u_1^{n r_1} u_2^{n r_2} \dots] C_\pi(z; \pi_1 u_1, \pi_2 u_2 \dots)} \quad (4.16)$$

$$= 2^{k/2} n^{k/2} \left( \frac{\lambda_{\pi_1, \pi_2 \dots}}{\kappa_{\pi_1, \pi_2 \dots}} + \mathcal{O}\left(\frac{1}{\sqrt{n}}\right) \right) \in \Theta\left(2^{k/2} n^{k/2}\right). \quad (4.17)$$

### 4.2.3 Beyond simplicity

The validity of those complexities extends beyond simple type grammars, and applies to any specification whose associated dominant singularity is unique, and arises from the contribution of a single connected component for the chosen weight assignment. It also applies to **periodic systems** such that, beyond a certain size  $N$

$$[z^n] C(z) \neq 0 \Leftrightarrow n \equiv 0 [p] \text{ for all } n > N$$

for some period  $p > 1$  through a simple rewriting of the grammar, coupled with a suitable change of variables.

However, general extensions of those complexities beyond strongly-connected systems appear very challenging. Indeed relaxing the connectivity constraint gives rise to many counter-examples, and may even induce bimodal distributions that cause an exponential dependency on  $n$  for the number of attempts. In fact, we were able to design an algorithm which constructs and return a specification whose limiting distribution for a dedicated atom follows an arbitrary discrete (univariate) distribution, specified by a user as part of the input [12].

**Remark 4.2.2:** Empirically, in the worst case scenario of an abysmally small probability of acceptance, estimates for the expected waiting time can be refined during the execution of the rejection algorithm. Upon reaching the conclusion of impending doom, an exact recursive algorithm [49] can then be elected.

### 4.3 Applications of multidimensional sampling

As noted above, failure to satisfy simplicity only (potentially) affects the complexity, but does not bear any consequence on correctness (uniformity). One can then take a *leap of faith*, and adopt the multidimensional Boltzmann paradigm in contexts where the concentration of the distribution is not guaranteed in general (but typically does hold).

In the context of RNA design (see Chapter 3 for further details), we argue for a change of paradigm, where design is approached as a random generation problem [88, 90, 117, 158, 160, 210]. Our methods generally rely on a polynomial-time dynamic programming algorithm to compute the **dual partition function** of a target structure  $S^*$

$$Z'_{S^*} = \sum_{\substack{\text{Sequence } w \text{ such that} \\ w \text{ compatible with } S^*}} e^{-\beta \cdot E_{w,S^*}}$$

where  $E_{w,S^*}$  represents the free-energy of  $S^*$  assuming a nucleotide sequence  $w$ , and  $\beta$  represents an arbitrary constant. A **stochastic backtrack** algorithm reinterprets the recurrences into a rewriting system allowing the random generation of a compatible with emission probability

$$P(w | S^*) = \frac{e^{-\beta \cdot E_{w,S^*}}}{Z'_{S^*}}.$$

#### 4.3.1 Controlling the GC-content (88, 90, 158, 160, 192, 193)

The **GC-content** of a sequence  $w$  simply represents its proportion of Guanines and Cytosines, defined as

$$GC(w) = \frac{|w|_G + |w|_C}{|w|}.$$

It is observed to vary substantially (16% to 75%) across genomes [94], or even between coding and non-coding regions within certain species [99], resulting from a selective pressure on the overall stability of RNAs.

Earlier methods for RNA design did not identify this parameter as an objective, leading to a strong bias towards high GC-contents observed among designed sequences, as revealed by our studies [158, 160]. Even more troublesome, was the observation that the GC-content of designed sequences typically appears Normal-distributed with linear variance. This means that a rejection strategy from the output of those tools would yield exponential runtime whenever the targeted GC-content does not coincide with the average GC-content of their output.

Fortunately, a multidimensional sampling strategy can be used to *shift* the expected GC-content of designed sequences to a given target. To that purpose, the dynamic program-

ming recurrences used for computing  $Z'$  can be adapted through a minor modification to compute

$$Z'_{\gamma, S^*} = \sum_{\substack{\text{Sequence } w \text{ such that} \\ w \text{ compatible with } S^*}} e^{-\beta \cdot E_{w, S^*}} \times \gamma^{\text{GC}(w)}$$

where  $\gamma$  is simply a positive weight (akin to  $\pi_t$  above). A mirrored change in the stochastic backtrack allows to sample the GC-weighted distribution

$$\mathbb{P}_{\gamma}(w \mid S^*) = \frac{e^{-\beta \cdot E_{w, S^*}} \times \gamma^{\text{GC}(w)}}{Z'_{\gamma, S^*}}.$$

This allows to shift the expected GC-content to a targeted GC-content  $g \in [0, 1]$  by first calibrating  $\gamma$ , e.g. using a binary search, and then perform rejection sampling to restrict the output to sequences having a targeted GC-content.

The complexity analysis here becomes even harder to define than the above average-case analysis, as the GC distribution now depends on the target structure  $S^*$ . One way to measure the complexity is to consider, for any size  $n$ , the **worst average-case complexity** over all possible structures over  $n$  nucleotides. In this setting, we could not formally establish the asymptotic complexity, but strongly suspect the expected number of attempts to grow like  $\mathcal{O}(\sqrt{n})$  in the worst-case. Indeed, popular energy models assign free-energy contributions to **loops** depending on their content, meaning that the GC contributions of two non-adjacent loops can be treated as independent random variables. We thus expect such limited, non-transitive, dependencies to allow an application of the **Central Limit Theorem**, and conclude on the normal distribution of the GC-content (sum of  $\Theta(n)$  almost-independent variables).

Similar strategies were used for controlling the GC-content in the design of algorithms for probing evolutionary landscapes [192, 193], and for advanced versions of design requiring compatibility with multiple targets [88, 90].

#### 4.3.2 Dinucleotides content of protein-coding RNAs (209).

Since the potential of sequences to adopt a given thermodynamic behavior cannot be easily captured at the generation stage, background models for RNAs usually include the conservation of a given **dinucleotides (2-mers) content**, defined for a sequence  $w$  as the vector

$$\mathbf{d}(w) = (d_{AA}(w), d_{AC}(w), d_{AG}(w), d_{AU}(w), \dots, d_{GG}(w))$$

where  $d_{xy}(w)$  denotes the number of occurrences in  $w$  for the 2-mer  $xy \in \{A, C, G, U\}^2$ . This property was essentially elected as an invariant, as it was empirically shown to act as a good proxy for the thermodynamic stability of RNAs [37]. Random RNA sequences

preserving a given 2-mers content can be uniformly generated as random Eulerian path, using the **dinucleotide shuffling algorithm** of Altschul and Erickson [6].

However, in the context of studying a protein-coding RNA  $m$ , a background model should ideally capture all known properties, to assess the statistical significance of a given phenomenon. Therefore, in addition to the 2-mers content, emitted sequences should also preserve the **reference protein**  $\mathcal{P}(m)$ , encoded as an **amino-acids sequence**. This constraints allows for substitutions of **codons** (non-overlapping 3-mers), each encoding an amino-acid, such that the resulting sequence  $m'$  is said to be **synonymous** to  $m$ . In short, given a coding sequence  $m$ ,  $|m| \equiv 0 \pmod{3}$ , we must generate, uniformly at random, one or several RNA sequences synonymous to  $m'$  such as

$$d(m) = d(m') \quad \text{and} \quad \mathcal{P}(m) = \mathcal{P}(m').$$

Here,  $\mathcal{P}(\cdot)$  denotes the conversion of a coding RNA into a sequence of amino acids, achieved by a simple morphism

$$\mathcal{P}(xyz.w) \rightarrow AA_{xyz}.\mathcal{P}(w) \quad \text{and} \quad \mathcal{P}(\varepsilon) \rightarrow \varepsilon$$

where  $AA_{xyz}$  represents the amino-acid associated with the codon  $xyz \in \{A, C, G, U\}^3$ .

Due to the highly non-sequential nature of the Altschul and Erickson algorithm [6], there is little hope of adapting it to enforce synonymous substitutions. However, synonymous sequences can be modeled as a simple linear language  $\mathcal{L}(m)$ , defined by

$$\mathcal{L}(\varepsilon) = \varepsilon \quad \text{and} \quad \mathcal{L}(xyz.w) = \{x'y'z' \mid AA_{x'y'z'} = AA_{xyz}\} \times \mathcal{L}(w).$$

A predefined 2-mers content can then be enforced using multidimensional Boltzmann sampling, using 15 dimensions to represent the number of occurrences of each but one of the 2-mers (the last one being indirectly captured as the complement to the length). A weights vector  $\mu := (\mu_{xy})_{xy \in \{A, C, G, U\}^2}$  is then introduced, defining a weighted distribution

$$\mathbb{P}_{\mu}(m' \mid m) = \frac{\prod_{xy \in \{A, C, G, U\}^2} \mu_{xy}^{|m'|_{xy}}}{\mathcal{Z}_{\mu, m}} \quad (4.18)$$

where

$$\mathcal{Z}_{\mu, m} \equiv \sum_{\substack{m' \text{ s.t.} \\ \mathcal{P}(m) = \mathcal{P}(m')}} \prod_{xy \in \{A, C, G, U\}^2} \mu_{xy}^{|m'|_{xy}}$$

The above partition function can be computed in linear-time using dynamic programming

$$\mathcal{Z}_{\mu, abc.w; \alpha} := \sum_{\substack{\alpha' b' c' \in \{A, C, G, U\}^3 \\ \text{s.t. } AA_{\alpha' b' c'} = AA_{abc}}} \mathcal{Z}_{\mu, w; c'} \times \prod_{xy \in \{A, C, G, U\}^2} \mu_{xy}^{|\alpha abc|_{xy}}$$

with  $\mathcal{Z}_{\mu, \varepsilon; \alpha} := 1$  and  $\mathcal{Z}_{\mu, m} \equiv \mathcal{Z}_{\mu, m; \varepsilon}$ . A stochastic sampling algorithm can then be adapted in a straightforward manner to sample with respect to the distribution (4.18),

and coupled with a rejection scheme to obtain the algorithm at the core of the SPARCS method [209]. Complexity-wise, the 15 degrees of freedom induce an expected  $\Theta(n^{15/2})$  attempts for each admissible sequence, under the assumption of a Normal multidimensional distribution. In practice, we found that a tolerance of a couple of occurrences for each 2-mers was sufficient to induce a linear empirical runtime for the method.

## 4.4 Redundancy in weighted sampling and countermeasures

Redundancy within of sampled sets of objects constitutes a possible shortcoming of methods relying on a weighted sampling. Indeed, as argued in Section 2.3.3, redundancy is generally useless in inferential statistics whenever the emission probability of a sample can be exactly determined *a posteriori*. However, the extent to which redundancy prevails *quantitatively* within random samples, and thus has a measurable adverse effect on the efficiency of estimators, had not been the object of precise inquiries beyond the seminal contribution of Flajolet, Gardy and Thimonier [70].

Let us first define formally the key concept of redundancy.

**Definition 4.4.1 (Redundancy of sample):** Given a collection  $\Omega$ , the **redundancy** of a sample  $(e_1, \dots, e_M) \in \Omega^M$  is defined as

$$\mathfrak{R}(e_1, e_2, \dots, e_M) = \frac{M}{|(e_1, e_2, \dots, e_M)|} - 1 \quad (4.19)$$

### 4.4.1 Collisions in weighted languages (58, 78)

The study of the expected redundancy is very much connected to the **waiting times** of several events occurring in probabilistic allocations. The **Birthday Paradox** (BP) problem represents one such instance, and considers the time of the first collision.

**Definition 4.4.2 (Birthday paradox):** Given a collection  $\Omega$  equipped with a probability distribution, the **Birthday**  $\mathcal{B}_\Omega$  is defined as

$$\mathcal{B}_\Omega = \mathbb{E}(M \mid \mathfrak{R}(E_1, \dots, E_{M-1}) = 0 \wedge \mathfrak{R}(E_1, \dots, E_M) > 0)$$

where  $E_i$  are random independent variables in  $\Omega$ .

For instance, in the uniform distribution over  $\Omega$ , one has

$$\mathbb{P}(\text{Collision at time } i \mid \{e_1, \dots, e_{i-1}\}) = \frac{|\{e_j\}_{j=1}^{i-1}|}{|\Omega|} < \frac{\mathcal{B}_\Omega}{|\Omega|}, \forall i \leq \mathcal{B}_\Omega$$

and it follows that, after  $\mathcal{B}_\Omega$  generations, one has

$$\mathbb{E}(\#\text{Collisions}) < \frac{\mathcal{B}_\Omega \times \mathcal{B}_\Omega}{|\Omega|} \in \mathcal{O}(1)$$

and we conclude that  $\mathbb{E}(\mathfrak{R}(e_1, \dots, e_{\mathcal{B}_\Omega})) \rightarrow 0$  for asymptotically large collections  $\Omega$ .

Another setting of interest is that of the **Coupon Collector** (CC) problem, the time which considers the expected sample size before the full collection is obtained.

**Definition 4.4.3 (Coupon Collector):** Given a collection  $\Omega$  equipped with a probability distribution, the **Coupon Collector**  $\mathcal{C}_\Omega$  is defined as

$$\mathcal{C}_\Omega = \mathbb{E}(M \mid \{e_1, \dots, e_M\} = \Omega) \quad (4.20)$$

It can be shown that the expected redundancy is an increasing function of  $M$ , so it is natural to bound the redundancy by that of the full collection, namely

$$\mathbb{E}(\mathfrak{R}(e_1, \dots, e_M) \mid \{e_1, \dots, e_M\} = \Omega) = \frac{\mathcal{C}_\Omega}{|\Omega|} - 1.$$

In collaboration with Danièle Gardy and Jérémie du Boisberranger [58, 78], we revisited these quantities attempting to obtain precise specialization of the general results of Flajolet, Gardy and Thimonier [70] in the specific context of **weighted languages**.

#### 4.4.1.1 Main results

Let  $C$  be a specification generating a language  $\mathcal{L}$ , with a collection  $\pi$  of weights associated with atoms. We denote by  $\pi_n^\nabla$  (resp.  $\pi_n^\nabla$ ) the **minimal** (resp. **maximal**) **weight of a word** in  $\mathcal{L}_n$ . Consider the weighted generating function

$$C_\pi(z) = \sum_{\omega \in \mathcal{L}} \pi_\omega z^{|\omega|} = \sum_{n \geq 0} \mu_{\pi, n} z^n$$

and denote by  $\rho_\pi$  its dominant singularity. Finally let us define the **k-th moment** of a  $\pi$ -weighted distribution as

$$\alpha_{k, n} = \sum_{i=1}^{m_n} p_i^k = \frac{\sum_{w \in \mathcal{L}_n} \pi(w)^k}{\mu_{\pi, n}^k} = \frac{\mu_{\pi^k, n}}{\mu_{\pi, n}^k}. \quad (4.21)$$

We consider the following conditions:

- C1 Diversity:** The max. probability  $\pi_n^\Delta / \mu_{\pi,n}$  of a word in  $\mathcal{L}_n$  is exponentially decreasing with  $n$ ;
- C2 Log-positive weights:** For each terminal symbol  $t \in \Sigma$ ,  $\pi_t > 1$ ;
- C3 Bounded dependency:** For any rational number  $k > 1$  and any weight vector  $\pi$  such that Condition **C2** holds,  $\rho_\pi^k < \rho_{\pi^k}$  holds.

**Theorem 6 (Birthday paradoxes in weighted languages):** Under conditions **C1**, **C2** and **C3**, the birthday paradox of  $\mathcal{L}_n$  is given by

$$\mathcal{B}_{\mathcal{L}_n;\pi} \sim \frac{\sqrt{\pi}}{\sqrt{2\alpha_{2,n}}} = \frac{\mu_{\pi,n}\sqrt{\pi}}{\sqrt{2\mu_{\pi^2,n}}} \in \Omega(\gamma^n), \quad \gamma := \frac{\rho_\pi}{\sqrt{\rho_{\pi^2}}} > 1$$

In other words, the first collision typically occurs after a time exponential on  $n$  (albeit with typically small base), inducing essentially zero redundancy before  $\mathcal{B}_{\mathcal{L}_n;\pi}$  samples.

**Theorem 7 (Coupon collector for weighted languages):** The expected number of generations  $\mathcal{C}_{\mathcal{L}_n;\pi}$  needed to get all words in  $\mathcal{L}_n$  is such that

$$\frac{\mu_{\pi,n}}{\pi_n^\vee} \leq \mathcal{C}_{\mathcal{L}_n;\pi} \leq 2 \cdot \mathcal{H}_{m_n} \cdot \frac{\mu_{\pi,n}}{\pi_n^\vee} \quad (4.22)$$

This result is a direct corollary of Berenbrink and Sauerwald [16], and was later refined in Du Boisberranger et al. [58]. In particular, Equation (4.22) implies that, for a vast collection of languages the **expected redundancy** can grow exponentially with  $n$  for a given targeted occupancy.

**Remark 4.4.1 (Redundancy of RNA sampling (homopolymer model)):** In particular, this results impacts any specifiable family of Boltzmann-weighted RNA secondary structures (homopolymer models), for which one typically has  $\pi_n^\vee \in \mathcal{O}(1)$ . It implies that, when the full collection is reached, the **expected number of copies** of a secondary structure essentially grows (up to a possible  $\Theta(n)$  factor) like  $\mu_{\pi,n}/|\mathcal{L}_n| \in \Theta(\alpha^n)$  with  $\alpha > 1$ . This forbids using rejection-based generation strategies for the non-redundant generation of large sample sizes, since rejections would then become overwhelming.

Moreover, denote by  $\mathcal{W}_{n,\pi}$  the **weight classes** in  $\mathcal{L}_n$ , such that

$$\mathcal{W}_{n,\pi} := \{(w, m) \mid \omega \in \mathcal{L}_n : \pi(\omega) = w \text{ and } m = |\{\omega \in \mathcal{L}_n \mid \pi(\omega) = w\}|\}.$$

Then the expected **number of distinct elements after  $k$  generations** can be computed.

**Theorem 8 (Distinct weighted words):** *The expected number of distinct words after  $k$  generations is given by*

$$\mathbb{E}(|\{e_1, \dots, e_k\}|) = \sum_{(w, m) \in \mathcal{W}_{n, \pi}} m \cdot \left(1 - \left(1 - \frac{w}{\mu_{\pi, n}}\right)^k\right) = \sum_{(w, m) \in \mathcal{W}_{n, \pi}} m \cdot \left(1 - e^{-\frac{w}{\mu_{\pi, n}} k}\right) + \mathcal{O}(1).$$

A similar results holds for the **coverage**  $\mathfrak{C}$  of a sample  $(e_1, \dots, e_k)$ , defined as

$$\mathfrak{C}(e_1, \dots, e_k) = \frac{\sum_{e \in \{e_1, \dots, e_k\}} \pi(e)}{\mu_{\pi, n}} \quad (4.23)$$

which represents the proportion of the distribution that has already been sampled.

**Theorem 9 (Coverage of weighted words):** *The expected coverage of a set of distinct words after  $k$  generations is given by*

$$\mathbb{E}(\mathfrak{C}(e_1, \dots, e_k)) = \sum_{(w, m) \in \mathcal{W}_{n, \pi}} m \cdot \frac{w}{\mu_{\pi, n}} \cdot \left(1 - \left(1 - \frac{w}{\mu_{\pi, n}}\right)^k\right).$$

Since there are at most  $(n+1)^{|\Sigma|}$  different compositions/classes of weights, the expressions of Theorems 8 and 9 can be computed in polynomial time algorithms from the **weighted-classified partition functions**  $\mu_{\pi, n}$ , also known as the **density of states**.

Since redundancy can become overwhelming, it is natural to pursue algorithmic countermeasures. More precisely, the above result shows that the algorithmic problem of sampling without replacement from a weighted language cannot be efficiently tackled using rejection, motivating an unranking approach [118], or a modified instance of the recursive method [118, 147], illustrated in Section 2.3.3 and used to build RNA kinetics landscapes [133].



## Chapter 5

# Asymptotic properties of RNA secondary structures and other trees

The tree-like nature of RNA secondary structures not only makes it an excellent playground for dynamic programming, but also enables the study of its generic asymptotic properties. Such work goes back to the very early days of RNA bioinformatics, with pioneering works from Waterman [194], Zuker and Sankoff [212], Viennot and Vauchassade de Chaumont [187], Hofacker [96], Bundschuh and Hwa [24], Nebel [139], Clote [36], Reidys and Jin [104], and others too numerous to mention exhaustively. While initially motivated by algorithmic [194] and evolutionary [212] considerations, the field has gradually turned towards questions in statistical mechanics, addressing the behavior of RNA at the thermodynamic equilibrium, as exemplified by the work of Bundschuh and Hwa [25]. In this context, asymptotic enumerations can be used to elucidate the equilibrium properties of random large homopolymers, including their dependency to changes of temperatures (phase transitions [23, 26]).

In the **homopolymer model**, nucleotides are anonymized, so that any pair can form **base pairs**, inducing structural motifs that are equally contributing to the stabilization of their conformation. Implicitly, this amounts to coarsely abstract an RNA as its length  $n$ . While such a simplification may appear bold at first sight, many of the properties derived from this model are robust to the incorporation/perturbations of the energy model, or can be used as a first step to recover the dependency on an actual nucleotide content (heteropolymer model) [24, 212].

This chapter summarizes some contributed asymptotic analyses, using techniques in enumerative and analytic combinatorics, to address questions in algorithmic analysis and design [33, 148, 150], RNA thermodynamics [38], network properties [180], and design/evolution [206]. In the interest of self-completeness, we start by reminding basic tools and notions. A magisterial exposition, including much more versatile techniques, can however (and should!) be sought in the Flajolet/Sedgewick *bible* [69].

**Outline.** In this short chapter, I first remind in Section 5.1 some basic concepts, tools and principles in analytics combinatorics and, in Section 5.2, I describe the application of those techniques to selected problems involving the secondary structure of RNA.

The following summarizes, and attempts to unify, contributions described within the following list of articles published in journals and/or presented at conferences.

#### Associated contributions

W. A. Lorenz, P. Clote, and Y. Ponty. Asymptotics of RNA shapes. *Journal of Computational Biology*, 15(1): 31–63, 2008

Y. Ponty. Efficient sampling of RNA secondary structures from the Boltzmann ensemble of low-energy: The boustrophedon method. *Journal of Mathematical Biology*, 56(1-2):107–127, 2008

Y. Ponty and C. Saule. A Combinatorial Framework for Designing (Pseudoknotted) RNA Algorithms. In *WABI 2011*, Saarbrücken, Germany, 2011

P. Clote, Y. Ponty, and J.-M. Steyaert. Expected distance between terminal nucleotides of RNA secondary structures. *Journal of Mathematical Biology*, 65(3):581–99, Sept. 2012

C. Chauve, J. Courtiel, and Y. Ponty. Counting, generating, analyzing and sampling tree alignments. *International Journal of Foundations of Computer Science*, 29(5):741–767, 2018

D. Surujon, Y. Ponty, and P. Clote. Small-world networks and RNA secondary structures. *Journal of computational biology : a journal of computational molecular cell biology*, 26(1):16–26, Jan. 2019

H.-T. Yao, C. Chauve, M. Regnier, and Y. Ponty. Exponentially few RNA structures are designable. In *ACM-BCB 2019*, pages 289–298, Niagara-Falls, United States, 2019. ACM Press

## 5.1 Basic tools in enumerative and analytic combinatorics

Analyzing the asymptotic enumerative behavior of a combinatorial class can be done by the following four-step program, referred to as the **symbolic method** in the pioneering work of the late Philippe Flajolet:

1. Find a unambiguously grammar/specification that generates the objects;
2. Translate the grammar into a system of functional equations;
3. Solve the system and identify the dominant singularities;
4. Use some automatic Theorem to extract the leading term for the asymptotic equivalent for the number of objects of size  $n$ .

### 5.1.1 Context-free languages.

A **combinatorial class** is a, possibly infinite, set  $\mathcal{A}$  where every object  $a \in \mathcal{A}$  has a **length**  $|a|$ , and such that the restriction  $\mathcal{A}_n$  to objects of  $\mathcal{A}$  having any given length  $n$  is finite.

Combinatorial classes are usually described by a specification, which can be interpreted as a generative process. Among such processes are **context-free grammars**, abstract derivation systems formally defined as shown below.

**Definition 5.1.1 (Context-free grammar):** A Context-free grammar is a 4-tuple  $G = (\mathcal{N}, \Sigma, \mathcal{R}, S_0)$  where:

- $\mathcal{N}$  is a set of intermediate symbols called **non-terminal symbols**;
- $\Sigma$  is a set of letters, also called **terminal symbols**, used for encoding our combinatorial objects;
- $\mathcal{R}$  is a set of **productions rules** of the general form  $S \rightarrow w$ , where  $v \in \mathcal{N}$  is a non-terminal symbol, and  $w$  is a word composed of non-terminal or terminal symbols. When the right-hand side is empty ( $|w| = 0$ ), the rule is said to generate the **empty word**  $\varepsilon$ ;
- $S_0$  is the **axiom**, a special non-terminal from which the generation is initiated.

A **derivation** results from the application of a production rule  $S \rightarrow w$  in  $\mathcal{R}$ , replacing the non-terminal  $S$  on the left-hand side with the right-hand side  $w$ . Starting from  $w := S_0$ , one can iterative choosing a production  $S' \rightarrow w'$  for the leftmost non terminal  $S'$  in  $w$ , replacing its occurrence with  $w'$ , until  $w$  only consists of terminal letters only. One then obtains a tree-like sequence of choices, called a **parse tree**. Considering (implicitly) all possible parse trees leads to a, possibly infinite, set  $\mathcal{L}_G$  of sequences, which is called the **language** of  $G$  and can be interpreted as a combinatorial class.

**Definition 5.1.2 (Generated language):** The language  $\mathcal{L}_S$  of a non-terminal  $S \in \mathcal{N}$  within a context-free grammar  $G = (\mathcal{N}, \Sigma, \mathcal{R}, S_0)$  is the set  $\mathcal{L}_S$  such that

$$\forall t \in \Sigma : \mathcal{L}_t = \{t\} \quad \text{and} \quad \forall S \in \mathcal{R} : \mathcal{L}_S = \bigcup_{(S \rightarrow w) \in \mathcal{R}} \prod_{i=1}^{|w|} \mathcal{L}_{w_i} \quad (5.1)$$

where  $\prod$  is the Cartesian product of languages, i.e. all ordered concatenations of words from the sets, defined as  $\{\varepsilon\}$  for an empty word  $w = \varepsilon$ .

The **language** of a CFG grammar  $G$  is defined as  $\mathcal{L}_G := \mathcal{L}_{S_0}$ .

Any grammar can be transformed into an equivalent **Chomsky Normal Form** (CNF) grammar, where the rules associated to each non-terminal  $S \in \mathcal{N}$  are restricted to the following cases:

- **Product type** non-terminal  $S \rightarrow T.U$  where  $T, U \in \mathcal{N}$ ;
- **Union type** non-terminal  $S \rightarrow T$  and  $S \rightarrow U$ , where  $T, U \in \mathcal{N}$ ;

- **Terminal type** non-terminal  $S \rightarrow t$ , where  $t \in \Sigma$  is a terminal symbol;
- **Epsilon type** non-terminal  $S \rightarrow \varepsilon$ , denoting the empty word;

Next comes the crucial notion of **unambiguity** of a grammar, which guarantees a unique way to produce each word in  $\mathcal{L}_G$ .

**Definition 5.1.3 (CFG Unambiguity):** CFG  $G = (\mathcal{N}, \Sigma, \mathcal{R}, S_0)$  is **unambiguous** if and only if one of the following (equivalent) statements is true:

- there exists a bijection between the parse trees and the elements of  $\mathcal{L}$ ;
- all parse trees produce distinct words over the terminal alphabet;
- the unions in Equation 5.1 are disjoint, and any word produced by a Cartesian products can be uniquely factored into its parts.

Another way if For the rest of this exposition, we are going to assume that the considered grammars are unambiguous and. Moreover, this time without loss of generality, we will assume that our grammars do not have **infinite cycles of unit productions**:

$$S \rightarrow T \rightarrow \dots \rightarrow S$$

Those are non productive and can be detected and eliminated by standard algorithms

### 5.1.2 Enumeration and generating functions.

Within unambiguous grammars, one can easily characterize the set of words  $\mathcal{L}_S$  generated from each non-terminal  $S \in \mathcal{N}$  and the number  $s_n$  of its elements of length  $n$ , depending of its type:

- Product  $S \rightarrow T.U$ : Each word generated from  $S$  can be decomposed into elements in  $\mathcal{L}_T$  and  $\mathcal{L}_U$  of length  $i$  and  $n - i$  respectively. The grammar is unambiguous, so this **decomposition is unique**, so we have

$$s_n = \sum_{i=0}^n t_i \times u_{n-i};$$

- Union  $S \rightarrow T$  and  $S \rightarrow U$ : The unambiguity of the grammar implies that the **union is disjoint**, so we have  $s_n = t_n + u_n$ ;
- Terminal  $S \rightarrow t$ : Generates  $\{t\}$ , so we have  $s_1 = 1$  and  $s_n = 0, \forall n \neq 1$ ;
- Epsilon  $S \rightarrow \varepsilon$ : Generates  $\{\varepsilon\}$ , so we have  $s_0 = 1$  and  $s_n = 0, \forall n \neq 0$ .

We can now introduce the key tool of enumerative combinatorics.

**Definition 5.1.4 ((Counting) generating function):** A **generating function**  $A(z)$  for a combinatorial  $\mathcal{A}$  is a formal power series that summarizes the cardinality information over  $\mathcal{A}$ :

$$A(z) = \sum_{a \in \mathcal{A}} z^{|a|} = \sum_{n \geq 0} a_n z^n$$

where  $a_n = |\mathcal{A}_n|$  denotes the number of objects of size  $n$  in  $\mathcal{A}$ .

In the above definition,  $z$  is a formal complex variable, and can be thought as a medium of useful information, later retrieved using a Taylor expansion at  $z = 0$ . We use the notation  $[z^n] A(z)$  to represent the  $n$ -th coefficient of  $A(z)$ , noting that  $[z^n] A(z) = a_n$ .

Consider the generating function  $S(z)$  associated with (language of) a non-terminal  $S \in \mathcal{N}$ . By definition, one has

$$S(z) = \sum_{w \in \mathcal{L}_S} z^{|w|} = \sum_{n \geq 0} s_n z^n.$$

The recursive nature of the grammar then induces a fairly simple recursive scheme, and the generating function  $S(z)$  of a non-terminal  $S$  obeys a remarkably simple **system of functional equations**:

- Product  $S \rightarrow T.U$ : We have

$$S(z) = \sum_{n \geq 0} s_n z^n = \sum_{n \geq 0} \left( \sum_{i=0}^n t_i \times u_{n-i} \right) z^n = T(z) \times U(z) \quad (5.2)$$

- Union  $S \rightarrow T$  and  $S \rightarrow U$ : We have

$$S(z) = \sum_{n \geq 0} s_n z^n = \sum_{n \geq 0} (t_n + u_n) z^n = \sum_{n \geq 0} t_n z^n + \sum_{n \geq 0} u_n z^n = T(z) + U(z).$$

- Terminal  $S \rightarrow t$ :  $S(z) = \sum_{w \in \mathcal{L}_S} z^{|w|} = z^{|t|} = z$ ;
- Epsilon  $v \rightarrow \varepsilon$ :  $S(z) = \sum_{w \in \mathcal{L}_S} z^{|w|} = z^{|\varepsilon|} = z^0 = 1$ .

Any unambiguous context-free grammar in CNF can thus be transformed into a system of functional equation. This system is **algebraic**, as Gaussian elimination can be used to express it as

$$P(z, S(z)) = 0 \text{ where } P \text{ is a polynomial in } S(z) \text{ and } z, \quad (5.3)$$

and its solutions are called **algebraic generating functions** by analogy.

#### Example 18: DNA strings

DNA strings can be expressed as a language  $\{A, C, G, T\}^*$ , generated by the context-free grammar

$$S \rightarrow A.S \mid C.S \mid G.S \mid T.S \mid \varepsilon$$

This grammar can be transformed in the CNF grammar for illustrative purposes

$$\begin{array}{lllll} S \rightarrow S' \mid E & S' \rightarrow T.S & T \rightarrow T' \mid T'' & T' \rightarrow a \mid c & T'' \rightarrow g \mid t \\ a \rightarrow A & c \rightarrow C & g \rightarrow G & t \rightarrow T & E \rightarrow \varepsilon \end{array}$$

The transforms described above, coupled with minor simplifications, lead to the following system

$$S(z) = 4z \times S(z) + 1 = \frac{1}{1-4z}$$

which can be solved to obtain

$$S(z) = \frac{1}{1-4z} = \sum_{n \geq 0} 4^n z^n.$$

**Remark 5.1.1:** In general, algebraic systems associated with context-free grammars are non-linear, so that conjugate solutions may exist. However, only one will admit positive integers as its coefficients.

### 5.1.3 Basic singularity analysis

Finding a closed-form expression for  $s_n$ , the coefficients of  $S(z)$ , is generally a highly involved task (although such a form always exists [11]). An easier alternative is to use **analytic combinatorics** to extract a simple asymptotic equivalent for  $s_n$ . Singularity analysis simplifies this problem by focusing on the **singularities of  $S(z)$** , i.e. points of the complex plane where the generating function ceases to be analytic.

Of particular interest are the **dominant singularities**, i.e. singularities of smallest modulus that are the main contributors to the coefficients. From Pringsheim's Theorem [69, pp 240], we know that at least one of them lies on the real axis for non negative series. Moreover, only limited types of singularities may occur in algebraic generating functions. This leads to predictable behaviors for the coefficients, and allows to bypass complex analysis (almost) entirely thanks to *automatic* theorems.

Rule type	Language	Coefficients	Generating function
Product $S \rightarrow T.U$	$\mathcal{L}_S = \mathcal{L}_T \times \mathcal{L}_U$	$s_n = \sum_{i=0}^n t_i \times u_{n-i}$	$S(z) = T(z) \times U(z)$
Union $S \rightarrow T, S \rightarrow U$	$\mathcal{L}_S = \mathcal{L}_T \cup \mathcal{L}_U$	$s_n = t_n + u_n$	$S(z) = T(z) + U(z)$
Terminal $S \rightarrow t$	$\mathcal{L}_S = \{t\}$	$s_n = \begin{cases} 1 & \text{if } n = 1 \\ 0 & \text{otherwise} \end{cases}$	$S(z) = z$
Epsilon $S \rightarrow \varepsilon$	$\mathcal{L}_S = \{\varepsilon\}$	$s_n = \begin{cases} 1 & \text{if } n = 0 \\ 0 & \text{otherwise} \end{cases}$	$S(z) = 1$

Table 5.1: Generated languages, enumerations and ordinary (counting) generating functions for unambiguous Chomsky Normal Form grammars.

**Theorem 10 (Asymptotic growth (69, pp 244)):** Let  $\rho \in \mathbb{R}^+$  be the dominant singularity of an algebraic generating function  $S(z)$ , then:

$$\lim_{n \rightarrow +\infty} \sqrt[n]{[z^n] S(z)} = 1/\rho.$$

In other words, the exponential part of the asymptotics is solely driven by the value of the dominant singularity. This property is sometimes referred to as the **first principle of Analytic Combinatorics**, and is sufficient to obtain crude estimates.

**Example 19: DNA strings – Asymptotics**

We illustrate this principle on DNA strings, associated with a generating function  $S(z) = \frac{1}{1-4z}$ . Here, the (unique) dominant singularity is a pole (cancellation of the denominator) at  $z = 1/4$ .

Theorem 10 implies that, for some subexponential term  $f(n)$ ,  $(\lim_{n \rightarrow +\infty} \sqrt[n]{f(n)} = 1)$ ,

$$[z^n] S(z) = 4^n \cdot f(n).$$

An application of Theorem 11 below refines this estimate to  $[z^n] S(z) = 4^n(1 + \mathcal{O}(1/n))$ , meaning that we can rule out the existence of an additional polynomial term.

The **second principle of Analytic Combinatorics** considers the *nature* of singularities to determine more precise estimates. **Transfer theorems** can then be used to relate the asymptotic growth of  $s_n$  to the behavior of  $S(z)$  close to its dominant singularity.

**Theorem 11 (Rational Singularities (69, pp 256)):** Let  $\rho \in \mathbb{R}^+$  be the dominant singularity of an algebraic generating function  $S(z)$ , rewritten as

$$S(z) = \frac{U(z)}{(1 - z/\rho)^\alpha} + T(z) \quad (5.4)$$

where  $\alpha \in \mathbb{N}^+ - \{0\}$ ,  $\kappa \in \mathbb{R}$ ,  $(1 - z/\rho)^\alpha \times T(z)$  and  $U(z)$  are analytic at  $z = \rho$  and  $\kappa := U(\rho)$ . Then the coefficients of  $S(z)$  admit the following asymptotic equivalent

$$[z^n] S(z) = \frac{\kappa}{(\alpha - 1)!} \times n^{\alpha-1} \times \rho^{-n} (1 + \mathcal{O}(1/n)). \quad (5.5)$$

Polar singularities are the only singularities arising from left/right linear grammars (or, more generally, rational languages), associated with systems in Equation 5.3 that are linear in  $S(z)$ . In such cases, the solution of the system is a **rational generating function**, i.e. it can be reduced to a form  $A(z)/B(z)$ , for  $A$  and  $B$  some polynomials. The alternative formulation required by Equation (5.6), can thus always be found.

**Example 20: Fibonacci language**

The Fibonacci language is generated by

$$F \rightarrow a F \mid b b F \mid \varepsilon$$

The number  $f_n$  of words of size  $n$  is thus given by

$$f_n = \begin{cases} f_{n-1} + f_{n-2} & \text{if } n \geq 2 \\ 1 & \text{otherwise } (n \in \{0, 1\}) \end{cases}$$

such that  $f_n$  is the  $(n + 1)$ -th Fibonacci number. The generating function is then

$$F(z) = z F(z) + z^2 F(z) + 1 = \frac{1}{1 - z - z^2} = \frac{1}{(1 - z/\rho_+)(1 - z/\rho_-)} \text{ with } \rho_{\pm} = \frac{-1 \pm \sqrt{5}}{2}.$$

Here, the dominant singularity is  $\rho_+$  since  $|\rho_+| < |\rho_-|$ .  $F(z)$  can then be reformulated as

$$F(z) = -\frac{\rho_-/\sqrt{5}}{1 - z/\rho_+} + \frac{\rho_+/\sqrt{5}}{1 - z/\rho_-}$$

where the second term is already analytic at  $z = \rho_+$  (and so is its multiplication by  $1 - z/\rho_+$ ).

A direct application of Theorem 11, with  $\rho := \rho_+$ ,  $\alpha := 1$ , and  $\kappa := \rho_-/\sqrt{5}$ , provides a precise asymptotic estimate

$$[z^n] F(z) = \frac{\rho_-}{\sqrt{5}} \rho_+^{-n} (1 + \mathcal{O}(1/n)).$$

Polar singularities may also dominate in (non linear) cases, and their simplicity always make them worthy of an initial investigation. However, non-rational languages will typically see more complex singularities arise and dominate their asymptotics. Fortunately, as covered by the following result, similar asymptotics hold, and can be derived at the cost of marginally more work.

**Theorem 12 (Single Algebraic Singularity (69, pp 393) + (11, 68)):** Let  $\rho \in \mathbb{R}^+$  be the single dominant singularity of an algebraic generating function  $S(z)$ , rewritten as

$$S(z) = \frac{U(z)}{(1 - z/\rho)^\alpha} + T(z) \quad (5.6)$$

where  $\alpha \in \mathbb{Q} - \mathbb{N}$ ,  $T(z)$  and  $U(z)$  are analytic at  $z = \rho$ , and  $\kappa := \lim_{z \rightarrow \rho} U(z) > 0$ .

Then the coefficients of  $S(z)$  admit the following asymptotic equivalent

$$[z^n] S(z) = \frac{\kappa}{\Gamma(\alpha)} \times n^{\alpha-1} \times \rho^{-n} (1 + \mathcal{O}(1/n)).$$

The main difference with the rational case covered by Theorem 11, is that  $\alpha$  can take more general values than just positive ones. This way, it covers singularities arising from radical forms  $\sqrt{1 - z/\rho}$ , called **square-root type singularities**, that are ubiquitous when enumerating tree-like objects.

The case of **multiple dominant singularities**, i.e. singularities  $\rho_0, \rho_1, \rho_2 \dots$  such that  $|\rho_0| = |\rho_1| = |\rho_2| = \dots$ , is slightly more involved. In cases where the co-dominant



singularities lead to different values  $\alpha_0 < \alpha_1 \leq \alpha_2$ , then only the largest needs to be considered, and Theorem 12 holds with a slightly enlarged error term  $\mathcal{O}(1/n^{\min(\alpha_0 - \alpha_1, 1)})$ .

However, singularities are complex numbers and may team with, or cancel each others' contributions for certain values of  $n$  (e.g. in the case of complex conjugates). However, as mentioned in Banderier and Drmota [11], multiple dominant singularities will give rise to periodic behaviors, with asymptotic equivalents still obeying the general form given in Theorem 12 (or 0) for any value of  $n$  such that  $n \equiv r [p]$ , where  $p$  is the **period** of the system.

In practice, a grammar featuring multiple dominant singularities can be rewritten, to only generate words of length  $n$  with a prescribed modular value  $r$  which, in combination with a suitable change of variables/vocabulary  $\Sigma$ , will suppress such periodic behaviors and lead to a single dominant singularity.

**Remark 5.1.2:** Banderier and Drmota [11] provide a definitive characterization of singularities in algebraic generating functions, including those of context-free languages. They show that the  $\alpha$  value for dominant singularities can only take **dyadic values**

$$\alpha \in \left\{ -\frac{1}{2^k} \mid k \geq 1 \right\} \cup \left\{ \frac{m}{2^k} \mid m \geq 1 \wedge k \geq 0 \right\} \subset \mathbb{Q}.$$

By further restricting the type of dominant singularities in algebraic generating functions, this results provides new tools to prove the intrinsic ambiguity of certain classes of languages and, more generally, the impossibility to model combinatorial classes using certain types of specifications.

#### Example 21: RNA Secondary Structures

Following Waterman [194], RNA secondary structures can be generated by the grammar

$$S \rightarrow (S^{\geq 1}) S \mid \bullet S \mid \varepsilon$$

where  $S^{\geq 1}$  forbids  $S$  to generate the empty word  $\varepsilon$ , and thus has generating function  $S(z) - 1$ . The associated system of functional equations is then

$$S(z) = z(S(z) - 1)zS(z) + zS(z) + 1 \Leftrightarrow 0 = z^2S^2(z) + (-1 + z - z^2)S(z) + 1$$

Solving the quadratic equation, gives two solutions for  $S(z)$

$$S^{\pm}(z) = \frac{1 - z + z^2 \pm \sqrt{\Delta}}{2z^2}$$

where  $\Delta$  is the discriminant

$$\begin{aligned} \Delta &:= (1 - z + z^2)^2 - 4z^2 = (1 + z + z^2)(1 - 3z + z^2) \\ &= \left(1 - \frac{z}{x_-}\right) \left(1 - \frac{z}{x_+}\right) \left(1 - \frac{z}{y_-}\right) \left(1 - \frac{z}{y_+}\right) \text{ with } x_{\pm} := \frac{3 \pm \sqrt{5}}{2} \text{ and } y_{\pm} := \frac{-1 \pm i\sqrt{3}}{2} \end{aligned}$$

Extracting the first terms of  $S^+(z)$  reveals negative coefficients, inconsistent with our expectation of a non-negative number of structures, and we conclude that

$$S(z) := S^-(z) = \frac{1 - z + z^2 - \sqrt{(1 - z/x_-)(1 - z/x_+)(1 - z/y_-)(1 - z/y_+)}}{2z^2}.$$

The function admits singularities for  $z \in \{x_-, x_+, y_-, y_+\}$ , and is analytic everywhere else in  $\mathbb{C}$ . The dominant singularity is  $\rho := x_-$  as it has smallest modulus, and we can rewrite  $S(z)$  as

$$S = \frac{1 - z + z^2}{2z^2} - \frac{1}{(1 - z/\rho)^{-1/2}} \times U(z) \quad \text{with} \quad U(z) := \frac{\sqrt{(1 - z/x_+)(1 - z/y_-)(1 - z/y_+)}}{2z^2}$$

where  $U(z)$  can be verified to be analytic at  $z = \rho$ , and reaches the following value

$$\kappa := U(\rho) = \frac{\sqrt{12\sqrt{5} - 20}}{(7 - 3\sqrt{5}) \times \sqrt{3 + \sqrt{5}}} = \sqrt{\frac{15 + 7\sqrt{5}}{2}}.$$

A direct application of Theorem 12, with  $\alpha = -1/2$ , gives the final asymptotic equivalent:

$$[z^n] S(z) = \frac{\kappa}{2\sqrt{\pi}} \times \frac{\rho^{-n}}{n\sqrt{n}} (1 + \mathcal{O}(1/n))$$

This asymptotic first-order estimate is remarkably accurate, and is already within 1% of the true count for RNAs of length 250 nts, and keeps on getting better and better to reach asymptotic equality.

#### 5.1.4 Useful extensions and shortcuts

##### 5.1.4.1 Ensemble properties using multivariate gen. fun. and derivatives

So far, generating functions have only provided valuable information on the number of words of size  $n$  within a language. However, in many cases, the asymptotic growth of  $s_n = [z^n] S(z)$  is arguably less relevant to a downstream application than the expected properties of a word in  $\mathcal{L}_n$ . To study such properties, one needs to (transiently) introduce a bivariate generating functions, which not only considers the length of an object, but also some additional property, here called a feature.

Namely, consider **atomic contributions**  $\{f_r\}_{r \in \mathcal{R}}$ , associating a (possibly null) numeric value to the production rules of a grammar.

**Definition 5.1.5 (Additive feature):** An **additive feature**  $F : \mathcal{L} \rightarrow \mathbb{R}$  is defined additively based on a set of atomic contributions such as, for all  $w \in \mathcal{L}$  generated by a unique parse tree  $p_w$ , one has

$$F(w) = \sum_{r \in p_w} f(r).$$

Features naturally include the number of occurrences  $|w|_t$  of a given letter  $t \in \Sigma$  in a word  $w$ . More complex features may require the design of an *ad hoc* grammar to accommodate highly-specific atomic contributions.

This allows to define the **bivariate generating function** of a language with respect to a

feature.

**Definition 5.1.6 (Bivariate generating function):** Given a language  $\mathcal{L}$ , the bivariate generating function  $S(z, u)$  for a feature  $F$  is

$$S(z, u) = \sum_{w \in \mathcal{L}} u^{F(w)} z^n = \sum_{n \geq 0} \sum_{k \geq 0} s_{n,k} u^k z^n \quad (5.7)$$

where  $s_{n,k}$  is the number of words of length  $n$  having value  $k$  for the feature  $F$ .

The function  $S(z, u)$  is solution of a system of equations similar to the constructs of Table 5.1, but with additional **monomials**  $u^{f_r}$  being assigned to each production rule  $r \in \mathcal{R}$  (unless  $f_r = 0$ , since  $u^{f_r} = u^0 = 1$ ). Namely, for any non-terminal  $S \in \mathcal{N}$ , the system will feature an equation:

$$S(z, u) = \sum_{r: S \rightarrow w} u^{f_r} \prod_{x \in w} \begin{cases} T(z, u) & \text{if } x := T \in \mathcal{N}, \\ z & \text{if } x \in \Sigma, \\ 1 & \text{otherwise.} \end{cases}$$

Atomic contributions will accumulate at the exponent of  $u$ , ultimately leading to the term  $u^{F(w)}$  in Equation (5.7).

**Expected value of a feature.** Solving the system, one obtains an expression for  $S(z, u)$ , from which information can be derived for the distribution of  $F(W)$ , the feature value of a random word. In particular, the **expected value** of  $F$  for a uniformly distributed word in  $\mathcal{L}_n$  is given by

$$\begin{aligned} \frac{[z^n] \frac{\partial S(z, u)}{\partial u} \Big|_{u \rightarrow 1}}{[z^n] S(z, 1)} &= \frac{[z^n] \sum_{w \in \mathcal{L}} F(w) u^{F(w)-1} z^n \Big|_{u \rightarrow 1}}{[z^n] \sum_{w \in \mathcal{L}} z^n} = \frac{\sum_{w \in \mathcal{L}_n} F(w)}{s_n} \\ &= \sum_{w \in \mathcal{L}_n} F(w) \times \mathbb{P}(W = w \mid n) = \mathbb{E}(F(W) \mid n) \end{aligned}$$

Interestingly, algebraic functions remain algebraic upon partial derivatives, so singularity analysis can be used to obtain an asymptotic equivalent for the coefficients of the numerator.

#### Example 22: Bivariate Motzkin words

To illustrate how bivariate generating functions can be used to analyze the content of random words, we consider the classic example of Motzkin words (see also Fig. 5.1), generated by the grammar

$$S \rightarrow (S) S \mid \bullet S \mid \varepsilon$$

Notice that, while  $($  and  $)$  play a symmetrical role, the letter  $\bullet$  seems quite different. Indeed,  $\bullet$  is essentially responsible for the extension of sequences, while  $($  and  $)$  emulate a parenthesis system, inducing tree-like structures. It is thus natural to pursue the *expected number of occurrences of  $\bullet$  in a*

uniformly-distributed word of size  $n$ .

First, we define the atomic features as the number of occurrences of  $\bullet$  produced by each derivation:

$$r_1 : S \rightarrow (S)S \Rightarrow f_{r_1} = 0 \quad r_2 : S \rightarrow \bullet S \Rightarrow f_{r_2} = 1 \quad r_3 : S \rightarrow \varepsilon \Rightarrow f_{r_3} = 0$$

Then we translate the grammar into a system of functional equations involving the bivariate generating function

$$S(z, u) = z^2 u^0 S(z, u)^2 + z u^1 S(z, u) + u^0 = z^2 S(z, u)^2 + z u S(z, u) + 1$$

Solving the system gives

$$S(z, u) = \sum_{n \geq 0} \sum_{k \geq 0} s_{n,k} u^k z^n = \frac{1 - zu - \sqrt{(1 - uz - 2z)(1 - uz + 2z)}}{2z^2}$$

One then derives by  $u$ , and set  $u = 1$ , to obtain the generating function counting the total number of occurrences of  $\bullet$  within Motzkin words of length  $n$ .

$$\begin{aligned} \frac{\partial S(z, u)}{\partial u} &= -\frac{z}{2z^2} + \frac{\sqrt{1 - uz - 2z}}{4z\sqrt{1 - uz + 2z}} + \frac{\sqrt{1 - uz + 2z}}{4z\sqrt{1 - uz - 2z}} \\ \left. \frac{\partial S(z, u)}{\partial u} \right|_{u=1} &= -\frac{z}{2z^2} + \frac{\sqrt{1 - 3z}}{4z\sqrt{1 + z}} + \frac{\sqrt{1 + z}}{4z\sqrt{1 - 3z}} \\ &= -\frac{z}{2z^2} + \frac{(1 - 3z) + (1 + z)}{4z\sqrt{1 + z} \times \sqrt{1 - 3z}} = -\frac{z}{2z^2} + \frac{1 - z}{2z\sqrt{1 + z}} \times \frac{1}{\sqrt{1 - 3z}} \end{aligned}$$

The resulting (mono variate) generating function features two singularities at  $z = 1/3$  and  $z = -1$  respectively, so the dominant one is  $\rho = 1/3$ . Applying Theorem 12 with  $\alpha = 1/2$  and  $U(z) = \sqrt{1 + z}/2z$  gives the following equivalent

$$[z^n] \left. \frac{\partial S(z, u)}{\partial u} \right|_{u=1} = \frac{1 - \rho}{2\rho\sqrt{1 + \rho}\sqrt{\pi}} \frac{\rho^{-n}}{\sqrt{n}} (1 + \mathcal{O}(1/n))$$

On the other hand, for the denominator we have

$$S(z, 1) = S(z) = \frac{1 - z - \sqrt{(1 - 3z)(1 + z)}}{2z^2} \Rightarrow [z^n] S(z) = \frac{\sqrt{1 + \rho}}{4\rho^2\sqrt{\pi}} \frac{\rho^{-n}}{n\sqrt{n}} (1 + \mathcal{O}(1/n))$$

Taking the ratio finally gives the expected number of occurrences of the letter  $\bullet$

$$\begin{aligned} E(|W|_{\bullet} | n) &= \frac{[z^n] \left. \frac{\partial S(z, u)}{\partial u} \right|_{u=1}}{[z^n] S(z)} = \frac{\frac{1 - \rho}{2\rho\sqrt{1 + \rho}\sqrt{\pi}} \frac{\rho^{-n}}{\sqrt{n}} (1 + \mathcal{O}(1/n))}{\frac{\sqrt{1 + \rho}}{4\rho^2\sqrt{\pi}} \frac{\rho^{-n}}{n\sqrt{n}} (1 + \mathcal{O}(1/n))} \\ &= n \frac{2\rho(1 - \rho)}{1 + \rho} (1 + \mathcal{O}(1/n)) = \frac{n}{3} (1 + \mathcal{O}(1/n)) \end{aligned}$$

Quite surprisingly, the asymptotic proportion of  $\bullet$  is exactly  $1/3$ , and so are those of  $($  and  $)$  (having equal number of occurrences), despite the highly asymmetrical roles played by the three letters.

**Remark 5.1.3 (Higher Order Moments):** By iterating the partial derivative (compensating with  $u$  each time), **higher moments** can be extracted in a similar manner, granting access to the **variance**, **skewness**, **kurtosis** of the feature distribution induced by the uniform distribution over words of length  $n$ .

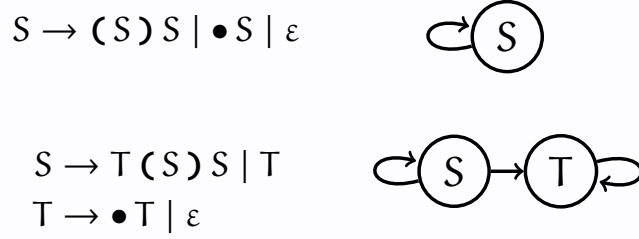


Figure 5.1: **Strong connectivity can be a matter of choice, as illustrated by two grammars for Motzkin words.** The top grammar has strongly-connected dependency graph, while the bottom grammar, although equivalent, induces a dependency graph which is not strongly-connected.

#### 5.1.4.2 Simple-type grammars

Shockingly, it is sometimes possible to skip steps 2, 3 and 4 of the symbolic method, and determine the type of singularity directly from properties of the grammar. To achieve *superlative laziness*, one first considers the dependencies in the grammar.

**Definition 5.1.7 (CFG Dependency Graph):** The **dependency graph** of a (weighted) grammar  $G = (\mathcal{N}, \Sigma, \mathcal{R}, S_0)$  is a directed graph, whose vertices are non-terminals ( $\mathcal{N}$ ), and whose arcs are any  $S \rightarrow T$  such that  $(S \rightarrow w) \in \mathcal{R}, T \in w$ .

A critical phenomenon occurs when the dependency graph is **strongly-connected**, i.e. when any vertex can be reached from any other vertex through a sequence of arcs. Note that the absence of strong connectivity for a dependency graph reflects on the chosen grammar, and does not necessarily represent an intrinsic property of the language  $\mathcal{L}$ . It is thus the responsibility of the modeler to find a suitable grammar, as shown in example described in Figure 5.1.

Intuitively, the combinatorial explosion can be understood as **flowing** through the dependency graph, and the strong-connectivity implies the existence of a steady-state, where the dominant growth homogeneously *contaminates* all vertices (a.k.a. non-terminals). Under this condition, the growth of the language can be restricted to two possible categories, depending on whether the grammar is **linear**, i.e. at most one non-terminal in its production rules, or **tree-like**, i.e. at least one production rule with two or more produced non-terminal.

**Theorem 13 (Strongly-connected Asymptotics: Perron-Frobenius (69, pp 343) + Drmota-Lalley-Woods (56, 112, 202) ):** Given a CFG  $G = (\mathcal{N}, \Sigma, \mathcal{R}, S_0)$  with strongly-connected dependency graph, only the following two types of dominant singularities may occur:

- If  $G$  is linear, then the dominant singularity  $\rho$  is a simple pole, i.e. there exists  $U(z)$  and  $T(z)$ , both analytic at  $z = \rho$ , with  $U(\rho) \neq 0$  such that:

$$S(z) = \frac{U(z)}{1 - z/\rho} + T(z) \Rightarrow [z^n] S(z) = U(\rho) \times \rho^{-n} (1 + \mathcal{O}(1/n));$$

- If  $G$  is tree-like, then the dominant singularity  $\rho$  has square-root type, i.e. there exists  $U(z)$  and  $T(z)$  analytic at  $z = \rho$ ,  $\kappa := \lim_{n \rightarrow +\infty} U(\rho) \neq 0$ , such that:

$$S(z) = -U(z) \times \sqrt{1 - z/\rho} + T(z) \Rightarrow [z^n] S(z) = \frac{\kappa}{2\sqrt{\pi}} \times \frac{\rho^{-n}}{n\sqrt{n}} (1 + \mathcal{O}(1/n)).$$

Such *stereotyped asymptotics* are immensely useful to study languages with large grammars, for which human patience and symbolic calculus can both reach their respective limits. In this case, computing the first coefficients of the generating function, e.g. using the recurrences of Figure 5.1, would allow to estimate  $\kappa$  and  $\rho$  to a sufficient precision in most applied contexts.

#### 5.1.4.3 Weighted languages and asymptotics

Another extension, crucial in applied contexts, considers the introduction of weights, and was explored in a collaboration with Alain Denise and Michel Termier [50].

**Definition 5.1.8 (Weighted grammar):** A **Weighted Language** is generated by a context-free grammar  $G = (\mathcal{N}, \Sigma, \mathcal{R}, S_0)$ , coupled with a **weight vector**  $\pi$ , associating a weight  $\pi_t$  to each terminal  $t \in \Sigma$ .

The **weight**  $\pi(w)$  of a word  $w \in \Sigma^*$  is defined by

$$\pi(w) = \prod_{i=1}^{|w|} \pi_{w_i}.$$

The overall weight of a language  $\mathcal{L}$  is then typically the quantity of interest, encapsulated in a **weighted generating function**

$$S_\pi(z) = \sum_{w \in \mathcal{L}} \pi(w) z^n = \sum_{n \geq 0} \sum_{w \in \mathcal{L}_n} \pi(w) z^n = \sum_{n \geq 0} s_n z^n$$

where  $s_n$  now represents the accumulated weight of all words of length  $n$  in  $\mathcal{L}$ . Weighted generating functions associated with unambiguous grammars are the solutions of an algebraic system of functional equations. Those can be obtained using a slightly modified version of the constructs in Table 5.1, where **terminals rules** now account for their respective weight:

$$(\text{Ordinary gen. fun.}) S(z) = z \rightarrow (\text{Weighted gen. fun.}) S_\pi(z) = \pi_t \times z. \quad (5.8)$$

While  $S_\pi(z)$  can be of interest in itself, e.g. as its coefficients represent **partition functions**, one can also adopt an alternative probabilistic view and think of the weights as inducing a Boltzmann-Gibbs distribution over the class of words of size  $n$ .

**Definition 5.1.9 (Weighted distribution):** Let  $G = (\mathcal{N}, \Sigma, \mathcal{R}, S_0, \pi)$  be an unambiguous CFG, generating a language  $\mathcal{L}$ , and  $n$  be a length, the **weighted distribution** is defined such that:

$$\mathbb{P}_\pi(w \mid n) = \frac{\prod_{i=1}^{|w|} \pi_{w_i}}{[z^n] S_\pi(z)} = \frac{\pi(w)}{s_n}$$

where  $s_n = [z^n] S_\pi = \sum_{w \in \mathcal{L}_n} \pi(w)$ .

The association of weights to terminal symbols (or, equivalently, to individual productions, as we showed [149]) grants more flexibility to the modeler in the design of a specification, particularly when weights are non-rational reals. Examples abound when real-valued weights are needed, e.g. when weights are optimized to achieve a targeted composition, or when classic Boltzmann-Gibbs distributions of the form  $\mathbb{P}(w) \propto e^{-E_w/\beta}$  are being investigated.

Moreover, their associated generating functions remain the solutions of algebraic systems of equations, so the entire *armada* of analytic combinatorics can be leveraged to analyze their asymptotic behavior.

**Proposition 14 (Asymptotics of Weighted Languages):** *The asymptotic equivalents of Theorems 10, 11, 12, and 13 hold for weighted generating functions.*

**Example 23: Expected number of secondary structures per RNA (212)**

The number of secondary structures of length  $n$  [194] only represents a crude upper bound for the number of secondary structures compatible with a given RNA sequence  $w$ ,  $|w| = n$ . indeed, canonical base-pairs can only form between pairs of nucleotides in

$$\mathcal{B} = \{(A, U), (U, A), (G, C), (C, G), (G, U), (U, G)\},$$

forbidding certain secondary structures to form for certain sequences.

While the precise number of such structures vary substantially depending on the actual sequence,

it is possible to obtain a more precise estimate by considering the **expected number of secondary structures**. To that purpose, one may consider, and enumerate, **pairs  $(w, S)$  of compatible sequence and structure**, i.e. such that, for any base pairs in  $S$  the associated nucleotides in  $w$  belong to  $\mathcal{B}$ . In other words, we need to find a grammar that jointly generates the sequence and structure, using an extended alphabet

$$\Sigma^{\mathcal{B}} = \{ (A, (C, (G, (U, )A, )C, )G, )U, \bullet A, \bullet C, \bullet G, \bullet U \}$$

leading to the grammar

$$\begin{aligned} S \rightarrow & (A S^{\geq 1})_U S \mid (U S^{\geq 1})_A S \mid (G S^{\geq 1})_C S \mid (C S^{\geq 1})_G S \\ & \mid (G S^{\geq 1})_U S \mid (U S^{\geq 1})_G S \mid \bullet A S \mid \bullet C S \mid \bullet G S \mid \bullet U S \mid \epsilon \end{aligned}$$

Equivalently, one can simply consider the usual grammar of Example 21, and use weights  $\pi_{\lceil} = 6, \pi_{\lrcorner} = 1, \pi_{\bullet} = 4$  to simulate the multiplicities of production rules. The associated system is then

$$S_{\pi}(z) = \pi_{\lceil} \pi_{\lrcorner} z^2 S_{\pi}(z) (S_{\pi}(z)^{\geq 1} - 1) + \pi_{\bullet} S_{\pi}(z) + 1 = 6 z^2 S_{\pi}(z) (S_{\pi}(z)^{\geq 1} - 1) + 4 S_{\pi}(z) + 1$$

Solving the system gives the weighted generating function

$$S_{\pi}(z) = \frac{1 - 4z + 6z^2 - \sqrt{1 - 8z + 4z^2 - 48z^3 + 36z^4}}{12z^2}$$

Singularity analysis can be performed, allowing estimates for the total number of compatible sequence/structure pairs

$$S_{\pi}(z) = \frac{\kappa \rho^{-n}}{n \sqrt{n}} (1 + \mathcal{O}(1/n))$$

with  $1/\rho \approx 8.164$ . Dividing by the number of RNA sequences of size  $n$  gives the average number of structures compatible with a uniformly distributed sequence of length  $n$ , and we get

$$\mathbb{E} (\# \text{Compatible structures} \mid n) \in \Theta \left( \frac{2.04^n}{n \sqrt{n}} \right)$$

This approach can be pushed even further, thanks to the flexibility of weighted/probabilistic generating functions, to produce asymptotics under more general random sequence models, including Bernoulli processes [212], or Markov chains [146].

## 5.2 Asymptotic combinatorics of RNA secondary structures

As mentioned in our introduction, RNA secondary structures have been the object of multiple works focusing on their asymptotic properties. We remind their formal definition, also defined in Section 1.2.3, for the sake of completeness.

**Definition 5.2.1 (RNA secondary structure):** An RNA secondary structure  $S$  of length  $n$  is a set of base-pairs  $(i, j), 1 \leq i < j \leq n$ , such that:

- Each position is **monogamous**,  $\forall (i, j) \neq (i', j') \in S : \{i, j\} \cap \{i', j'\} = \emptyset$ ;
- **Minimal distance**  $\theta$  between paired nucleotides,  $\forall (i, j) \in S : j - i > \theta$ ;
- **No pseudoknot** allowed,  $\forall (i, j), (i', j') \in S, i < i' : (j' < j) \text{ or } (j < i')$ .

In the  $\theta = 1$  case, i.e. no pairing of consecutive pairs, we obtain a popular combinatorial



$$\begin{array}{c}
\text{~~~~~} \bullet = \bullet \text{~~~~~} \cup \overset{\geq 1}{\text{~~~~~}} \cup \varepsilon \\
S \quad \rightarrow \quad \bullet S \quad | \quad (S^{\geq 1}) S \quad | \quad \varepsilon \\
\\
\text{Final grammar:} \quad S \rightarrow \bullet S \mid (T) S \mid \varepsilon \quad T \rightarrow \bullet S \mid (T) S
\end{array}$$

Figure 5.2: **Decomposition of secondary structures and associated context-free grammar.** Note that one can always systematically rewrite a grammar to enforce cardinality constraints such as the one found in  $S^{\geq 1}$ .

class ([OEIS:A004148](#)), admitting a wealth of equivalent characterizations, including:

1. Well-parenthesized expressions over  $\{ (, ), \bullet \}$  without occurrences of  $()$ ;
2. Positive walks from 0 to 0 taking steps  $\{ \nearrow, \searrow, \rightarrow \}$  without **peaks** ( $\nearrow \searrow$ );
3. Unary-binary trees without **sherries** (simple binary tree, occurring as a leaf)...

In particular, the representation as well parenthesized expressions immediately suggests a **decomposition for secondary structures**, ultimately leading to a grammar. Namely, consider a non- $\varepsilon$  secondary structure, and focus on its first nucleotide  $b$ :

- If  $b$  is unpaired, then what follows is itself secondary structure;
- If  $b$  is paired to some partner  $b'$ , then  $b$  and  $b'$  delimit an inner interval where a (non empty) structure can be found. Following the interval  $[b, b']$  is another (unconstrained) secondary structure.

It follows that secondary structures can be split into two subsets, themselves built from (smaller) structures. Those subsets are disjoint since a given position cannot be simultaneously paired and unpaired, leading to an **unambiguous** decomposition. Their union, complemented with the empty structure  $\varepsilon$ , is also **complete** with respect to secondary structures.

**Theorem 15 (Asymptotics of secondary structures (194)):** *The number  $s_n$  of secondary structures of length  $n$  is asymptotically equivalent to*

$$s_n = \sqrt{\frac{15 + 7\sqrt{5}}{8\pi}} \cdot \frac{\phi^n}{n\sqrt{n}} (1 + \mathcal{O}(1/n)) \text{ where } \phi = \rho^{-1} \approx 2.62 \dots$$

An illustrative proof of this result, using modern techniques of analytic combinatorics, is given in Example 21.

### 5.2.1 Expected 5'–3' distance (38)

A first application of the above principles considers the **5'–3' distance**, i.e. the graph distance induced by a(n ensemble of) secondary structure(s) between the ends of an RNA.

**Definition 5.2.2 (5'–3' distance):** Given an RNA secondary structure  $S$ , the **5'–3' distance**, graph distance induced by a(n ensemble of) secondary structure(s) between the ends of an RNA

Yoffe *et al* [207] argued that a small distance, independent of RNA length, between the two ends of RNA single-stranded molecules. The authors claimed that such a small distance was intrinsic to linear polymers, a claim supported by a heuristic argument arising from polymer physics, and empirically from the systematic MFE folding of randomly generated RNAs under various compositions. Such a short distance is indeed deemed crucial to the effective circularization of Eukaryotic messengers RNAs [196] through RNA-proteins and protein-protein complexes.

In collaboration with Peter Clote and Jean-Marc Steyaert [38], we decided to revisit this claim at three different levels: Theoretically, by checking its asymptotic validity in the homopolymer model; Algorithmically, by introducing a  $\Theta(n^3)$  dynamic programming algorithm to compute the expected 5'–3' distance for a given sequence at the thermodynamic equilibrium; Empirically, by computing the graph distance on a collection of RNAs of documented structures.

Our asymptotic analysis relies on the design of a grammar which, by identifying nucleotides on the 5' to 3' shortest path, allowed the derivation of a bivariate generating function for the 5'–3' distance. We achieved this by a classic, if technical, application of the symbolic method. Essentially, we **duplicated the production rules**, distinguishing rules that generate the **exterior bases** from those whose content is nested within some base pair, and are as such not relevant to the expected 5'–3' distance. A similar reasoning allowed to capture general values for the minimum distance  $\theta$  between two paired positions, and we obtained the following grammar:

$$S_\theta \rightarrow [R_\theta]S_\theta \mid \circ S_\theta \mid \varepsilon; \quad T_\theta \rightarrow (R_\theta)T_\theta \mid \bullet T_\theta \mid \varepsilon; \quad R_\theta \rightarrow (R_\theta)T_\theta \mid \bullet R_\theta \mid \bullet^\theta \quad (5.9)$$

Translating into a system of bivariate functional equation, one obtains

$$\begin{aligned} S_\theta(z, u) &= z^2 u^2 R_\theta(z) S_\theta(z, u) + z u S_\theta(z, u) + 1 \\ T_\theta(z) &= z^2 R_\theta(z) T_\theta(z) + z T_\theta(z) + z^\theta \\ R_\theta(z) &= z^2 R_\theta(z) T_\theta(z) + z R_\theta(z) + z^\theta. \end{aligned}$$

Solving the system gives the bivariate generating function

$$S_\theta(z, u) = \sum_{n \geq 0} \sum_{k \geq 0} s_{k,n;\theta} u^k z^n = \frac{1}{1 - zu - \frac{u^2(1-2z+z^{\theta+2}-\sqrt{\Delta_\theta})}{2(1-z)}} \quad (5.10)$$

with  $s_{k,n;\theta}$  the number of secondary structures of length  $n$ , with minimum base pair distance  $\theta$ , and having  $k$  nucleotides on the exterior face, and

$$\Delta_\theta = 1 - 4z + 4z^2 - 2z^{\theta+2} + 4z^{\theta+3} - 4z^{\theta+4} + z^{2\theta+4}. \quad (5.11)$$

In particular, one gets

$$S_\theta(z, 1) = \frac{1 - 2z + 2z^2 + 2z^\theta - z^{\theta+2} - \sqrt{\Delta_\theta}}{(1-z)2z^2}.$$

Next, we use the classic bivariate analysis

$$E_\theta(z) = \left. \frac{\partial S_\theta(z, u)}{\partial u} \right|_{u=1} = \frac{P(z) - (2 - 5z + 4z^2 - 2z^{\theta+2} + z^{\theta+3})\sqrt{\Delta_\theta}}{2(1-z)^2 z^4}$$

such that  $e_n := [z^n] E_\theta(z)$  is the accumulated number of exterior bases across all RNAs of length  $n$ , and  $P(z)$  is a polynomial of degree  $\theta + 4$ , whose precise value bears no consequence on the asymptotics. Applying Theorem 12, one gets

$$[z^n] S_\theta(z, 1) \sim -\frac{1}{(1-\rho)2\rho^2} \times [z^n] \sqrt{\Delta_\theta} \quad (5.12)$$

$$[z^n] E_\theta(z) \sim -\frac{(2 - 5\rho + 4\rho^2 - 2\rho^{\theta+2} + \rho^{\theta+3})}{2(1-\rho)^2 \rho^4} \times [z^n] \sqrt{\Delta_\theta}. \quad (5.13)$$

where  $\rho$  is the shared dominant singularity of  $S_\theta(z)$  and  $E_\theta$ , i.e. the root of  $\Delta_\theta$  having smallest modulus (real by Pringsheim). We obtain the following result by taking the ratio of estimates (5.13) and (5.12) to obtain the expected number of bases on the exterior base, to which we subtract 1 to obtain the graph distance.

**Theorem 16 (Expected 5'-3' distance - Homopolymer model (38)):** *The expected 5' - 3' distance  $D_n$  over all RNA secondary structures of length  $n$ , assuming minimum distance  $\theta$  between paired nucleotides, is given by*

$$D_n = \frac{[z^n] E_\theta(z)}{[z^n] S_\theta(z, 1)} - 1 = \frac{2 - 5\rho + 3\rho^2 + \rho^3 - 2\rho^{\theta+2} + \rho^{\theta+3}}{(1-\rho)\rho^2} (1 + \mathcal{O}(1/n)) \quad (5.14)$$

where  $\rho$  is the smallest modulus root of

$$\Delta_\theta := 1 - 4z + 4z^2 - 2z^{\theta+2} + 4z^{\theta+3} - 4z^{\theta+4} + z^{2\theta+4}. \quad (5.15)$$

As suspected by Yoffe *et al* [207], the expected value is constant, small and stable to different values of  $\theta$ . Of particular interest are the following numerical approximations,

for  $\rho$  and  $D_n$  when  $\theta = 1$  and 3:

$$\theta = 1 \rightarrow \rho \approx 0.381, D_n \approx 5.47$$

$$\theta = 3 \rightarrow \rho \approx 0.436, D_n \approx 4.15$$

The expected distance can also be analyzed in a refined model introduced by Zuker and Sankoff [212] (see Example 23 for details), which captures base pairing rules that prevent certain structures to be adopted by a given sequence. In this model, a **stickiness parameter**  $\sigma$  represents the probability that a sequence, randomly generated by a Bernoulli process, forms a given pair ( $\sigma := \sum_{\text{Valid BPs}(b,b')} p_b \times p_{b'}$ ). Incorporating the stickiness in our above analysis allows to derive the **expected 5'-3' distance induced by a random RNA sequence**.

**Theorem 17 (Expected 5'-3' distance - Random Sequence model (38)):** *The expected 5' – 3' distance  $D_n^\sigma$  in a random sequence of length  $n$ , generated by a Bernoulli process (stickiness  $\sigma$ ), under a min. base pair distance  $\theta$ , is given by*

$$D_n^\sigma = \frac{\Phi_{\theta,\sigma}(\rho) - 4(\rho - 1)^3 \sigma^2 \rho^4}{4(\rho - 1)^3 \sigma^2 \rho^4} (1 + \mathcal{O}(1/n)) \quad (5.16)$$

where  $\Phi_{\theta,\sigma}(z)$  is a (relatively) simple polynomial

$$\Phi_{\theta,\sigma}(z) := 4 - 14z - 18z^2 + (2\sigma - 10)z^3 + (2 - 2\sigma)z^4 - 4\sigma z^{\theta+2} + 6\sigma z^{\theta+3} - 2\sigma z^{\theta+4},$$

and  $\rho$  is the root of smallest modulus of

$$1 - 4z + (6 - 2\sigma)z^2 + 4(\sigma - 1)z^3 + (\sigma - 1)^2 z^4 - 2\sigma z^{\theta+2} + 4\sigma z^{\theta+3} - 2\sigma(1 + \sigma)z^{\theta+4} + \sigma^2 z^{2\theta+4}.$$

Again, the introduction of the stickiness does not change the nature of the expected distance, which remain a constant of  $n$ .

**Our conclusions [38].** Interestingly, while both the asymptotics and equilibrium analysis of RNA sequences appeared to support the hypothesis of Yoffe *et al* [207], our empirical analyses of the STRAND database [8], consisting of structured non-coding RNAs, indicated in a positive observed correlation between the 5'-3' distance and RNA length. This observation suggests that structural RNAs may be under selective pressure to have larger 5'-3' distance than expected from pure polymer theory, warranting further analyses.

## 5.2.2 RNA network properties (180)

First introduced by Watts *et al* [195], **Small-world networks** are defined as satisfying the following, informally-stated, properties:

- the **shortest path distance** between any two nodes is “small”, e.g. six degrees of separation between any two persons;
- the average **clustering coefficient** is large, e.g. friends of a person tend to be friends of each other.

Small-world networks are ubiquitous in biology, sociology, and information technology; examples abound, including the neural network of *C. elegans* [195], the gene co-expression in *S. cerevisiae* [186], protein folding networks [22, 170]. . . For additional examples, see the excellent review of Albert and Barabási [4]. In particular, Wuchty [204] observed a small-world property for the low energy RNA secondary structure network of *E. coli* phe-tRNA, and hypothesized this property to be crucial for the structural kinetics induced by tRNA modifications.

In collaboration with Defne Surujon and Peter Clote [180], we investigated asymptotic properties of the **RNA network**, the ensemble of all RNA secondary structures connected by a **move set**.

**Definition 5.2.3 (Move set):** Given an RNA length  $n$ , a **move set**

$$\mathcal{MS} = \{M_1, M_2, \dots\}$$

is a collection of **moves**, i.e. functions  $M : \mathcal{S}_n \rightarrow \mathcal{P}(\mathcal{S}_n)$ .

For any structure  $S \in \mathcal{S}_n$ , the execution of a move returns structures  $M(S) \subset \mathcal{S}_n$ , each resulting from the application of a (small) perturbation to  $S$ . Popular moves [72], in the context of a structure  $S$ , include:

- **Addition** of a base pair  $(i', j')$  such that  $S \cup \{(i', j')\}$  is a valid structure;
- **Removal** of a base pair  $(i, j) \in S$ ;
- **Shift** of  $(i, j) \in S$  into  $(i', j')$ ,  $i = i'$  or  $j = j'$ , only if resulting in a valid structure;

We considered two move sets:

- $\mathcal{MS1} := \{\text{Addition, Removal}\}$ ;
- $\mathcal{MS2} := \{\text{Addition, Removal, Shift}\}$

Notice that, while individual moves may not be reversible,  $\mathcal{MS1}$  and  $\mathcal{MS2}$  are both symmetric, and allow to define a notion of **neighborhood** crucial to the definition of a network.

**Definition 5.2.4 (RNA Network):** Given a length  $n$  and a symmetric move set  $\mathcal{MS}$ , the **RNA Network** is the undirected graph  $G_{\mathcal{MS}} = (V, E)$ , where  $V = \mathcal{S}_n$  are secondary structures of size  $n$ , and  $E$  contains pairs of structures  $\{S, S'\}$  such that  $S' \in M(S)$  for some move  $M \in \mathcal{MS}$  ( $\Rightarrow S \in M'(S')$ ).

We particularly focused on the expected degree and clustering coefficient of large RNA networks, and rigorously proved that **the RNA network is asymptotically not small-world**. Following [40], a family  $\{G_n, n = 1, 2, 3, \dots\}$  of graphs must satisfy certain properties to qualify as **small-world**.

**Definition 5.2.5 (Small-world property (40)):** A family  $\{G_n \mid n = 1, 2, 3, \dots\}$  of graphs is **small-world** if and only if the following conditions hold:

- A** The largest distance between any two nodes in  $G_n$  is in  $\mathcal{O}(\log |V|)$ ;
- B** The average degree of  $G_n$  is in  $\mathcal{O}(\log |V|)$ ;
- C** The global clustering coefficient  $\mathfrak{C}_g(G_n)$  is bounded away from zero.

We remind that the **global clustering coefficient**  $\mathfrak{C}_g(G)$  of a graph  $G = (V, E)$  is defined by Newman *et al* [141, Equation (77)] as:

$$\mathfrak{C}_g(G) = \frac{3 \times \#\text{triangles}}{\#\text{connected triples}} \quad (5.17)$$

where a **triangle** is a set  $\{x, y, z\} \in V$  of nodes such that  $\{(x, y), (y, z), (z, x)\} \subset E$ , and a **(connected) triple** is a set  $\{x, y, z\}$  of nodes, such that  $\{(x, y), (y, z)\} \subset E$ .

Clearly, families of RNA networks induced by  $\mathcal{MS}_1$  and  $\mathcal{MS}_2$  both satisfy condition **A**, since any structure can be reached from any other one in  $\Theta(n)$  operations (e.g.  $\leq n/2$  removals +  $\leq n/2$  additions), while  $|V| = |\mathcal{S}_n| \in \Theta(\rho^{-n}/n\sqrt{n})$ .

**Expected degree.** The expected degree of a secondary structure, the average number of distinct structures in the neighborhood of a structure  $S$ , obviously depends on the move set. For  $\mathcal{MS}_1$ , starting from a structure  $S$ , any base pair in  $S$  can be removed, but only **co-accessible positions** (i.e. belonging to the same loop) can be safely added to  $S$ . For  $\mathcal{MS}_2$ , shift moves allow more flexibility, allowing the binding of positions involved in two adjacent loops. In both cases, we can already conclude that Condition **B** holds, since the degree of any  $S$  is in  $\mathcal{O}(n^3)$ , i.e. bounded by a polynomial in  $n$ , while  $|V| = |\mathcal{S}_n|$  grows exponentially on  $n$ .

Still, a more precise analysis of the expected degree sheds some light on the techniques used to study the clustering coefficients, so we elaborate its main steps. Essentially, we

first observe that, in any structure  $S \in \mathcal{S}_n$ , any base pair can be removed, so the overall number of removal moves within  $\mathcal{S}_n$  is exactly characterized by the accumulated number of base pairs in  $\mathcal{S}_n$ .

Moreover, since additions and removals are reciprocal operations, there exists a bijection between pairs of structure/removal and structure/addition. The overall degree over  $\mathcal{S}_n$  is then  $2 \times b_n$ , where  $b_n$  is the overall number of base pairs in  $\mathcal{S}_n$ . Its associated generating function can be obtained by defining the number of base pairs as a feature, as shown in Section 5.1.4.1. Alternatively, we built an *ad hoc* grammar that generates all possible secondary structures with all possible *distinguished* base pair (here with  $\theta = 1$ ):

$$\begin{aligned} S^\bullet &\rightarrow [S^{\geq \theta}] S \mid (S^{\bullet, \geq \theta}) S \mid (S^{\geq \theta}) S^\bullet \mid \bullet S^\bullet \\ S &\rightarrow (S^{\geq \theta}) S \mid \bullet S \mid \varepsilon \end{aligned}$$

Executing the various steps of the symbolic method, followed by singularity analysis, one obtains an asymptotic equivalent for  $b_n$ . Dividing  $2 \cdot b_n$  by  $|\mathcal{S}_n|$  gives the expected degree  $\delta_n^{MS1}$  under  $\theta = 3$ .

**Theorem 18 (Expected degree of the RNA network (180)):** *The expected degree  $\delta_n^{MS}$  of a random uniform node in the RNA network  $G_n$  of a move set  $MS$  is such that:*

$$\begin{aligned} \delta_n^{MS1} &= \kappa \cdot n(1 + \mathcal{O}(1/n)) \quad \text{with} \quad \kappa \approx 0.473; \\ \delta_n^{MS2} &= \kappa \cdot n(1 + \mathcal{O}(1/n)) \quad \text{with} \quad \kappa \approx 1.56. \end{aligned}$$

To compute the equivalent of  $b_n$  for the shift move only, we built a grammar based on a similar idea as above, encoding the position and outcome of the move using an extended alphabet ( $\star$  for the fixed end of the base pair, and  $\langle \rangle$  for the shifting end). We obtained the following grammar, provably unambiguous and complete with respect to shifting moves:

$$\begin{aligned} \widehat{S} &\rightarrow \widehat{S}\bullet \mid (\widehat{S}) \mid S(\widehat{S}) \mid \widehat{S}(R) \mid \widehat{T} \\ \widehat{T} &\rightarrow \star R \rangle \rangle \mid S \star R \rangle \rangle \mid \star R \rangle S \rangle \mid S \star R \rangle S \rangle \mid \langle \langle R \star \mid S \langle \langle R \star \\ &\quad \mid \langle S \langle R \star \mid S \langle S \langle R \star \mid \langle R \star R \rangle \mid S \langle R \star R \rangle \\ S &\rightarrow \bullet \mid S\bullet \mid (R) \mid S(R) \quad R \rightarrow \emptyset \mid R\bullet \mid (R) \mid S(R) \\ \emptyset &\rightarrow \bullet^\theta \end{aligned}$$

The resolution of the associated system, in itself, justifies investing in a symbolic computation software, but can nevertheless be done, leading to the asymptotics summarized in Theorem 19.

**Clustering coefficient.** Finally, we address the validity of Condition **C**, based on the value of the clustering coefficient, the last missing piece to assess the small world nature

of homopolymer-based RNA networks. This requires the computation of the numbers of triangles and connected triples, two features which we capture in a similar strategy as above, and designing two unambiguous grammars whose generated languages are in bijection with the triangles and triples respectively. The associated grammars, which we spare the reader out of pure humanity, were tediously designed to exhaust (not only its designers but also) all pairs of valid consecutive moves in all contexts, this while capturing the proximity (triangle) or not (triple) of the initial and final structures.

**Theorem 19 (Triangles, triples and clustering in RNA networks (180)):** *Under the  $\mathcal{MS}_2$  move set with  $\theta = 3$ , the RNA network  $G_n$  is such that:*

$$\begin{aligned} \#triangles(G_n) &= \kappa n (1 + \mathcal{O}(1/n)) && \text{with } \kappa \approx 1.22 \\ \#connected\ triples(G_n) &= \lambda n^2 (1 + \mathcal{O}(1/n)) && \text{with } \lambda \approx 0.112 \\ \mathfrak{C}_g(G_n) &= \frac{3 \times \#triangles}{\#connected\ triples} \in \Theta\left(\frac{1}{n}\right) \end{aligned}$$

It follows that  $\mathfrak{C}_g(G_n) \rightarrow 0$  as  $n \rightarrow \infty$ , meaning that Condition **C** is not satisfied. This result is robust to changes of  $\theta$ .

**Corollary 20 (RNA networks are not small world):**

*The family of RNA networks is not small world.*

**Conclusions.** Our result that, in an homopolymer model, RNA networks are not small world seem to contradict the conclusions of Wuchty [204] based on empirical observations. However, it must be noted that Wuchty [204] restricted its observations to the most stable conformations (low-free energy). Moreover, this observed discrepancy between a sequence-free (homopolymer) model and Wuchty’s observations based on a limited subset of RNAs may be reconciled by the existence of a specific selective pressure for this family.

### 5.2.3 RNA shapes: Abstract representations of structures (120)

**RNA shapes**, introduced by Giegerich *et al* [82], represent a hierarchy of abstract representations for the secondary structures of RNA. Each secondary structure is assigned a shape at every level of abstractions, so that shapes at any level represent **equivalence classes** for the folding space. By providing a hierarchy coarse-grained descriptors, they enable a variety of analysis, including a comparison-free clustering of secondary structures [27], e.g. generated by stochastic sampling [51] within the Boltzmann ensemble of



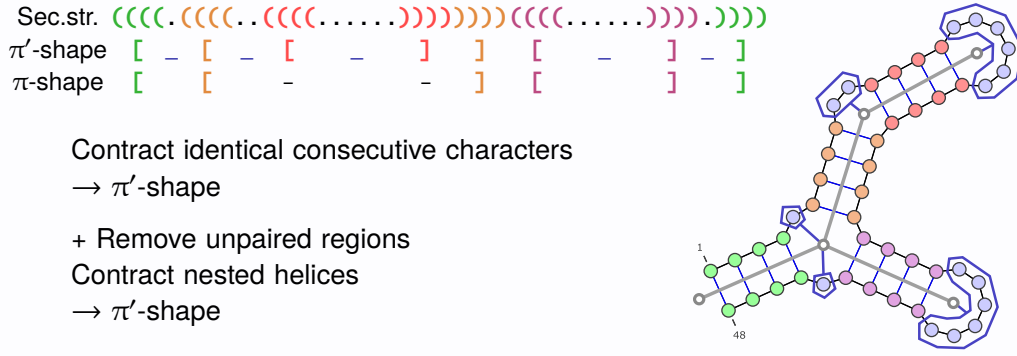


Figure 5.3: **Example of RNA shapes constructions from a secondary structure.** While  $\pi'$  shapes (most detailed) merely contract unpaired regions and helices,  $\pi$  shapes (coarsest) only retain the underlying tree backbone (gray tree).

low energy; or the alignment-free detection of recurrent conformations across homologous sequences [155]; or the indexation of RNA families for a faster attribution [102].

However, while shapes can be computed at various levels of abstractions (see Figure 5.3), and in linear time, the computational tractability, and statistical power, of a shape-based analysis is conditioned by a reasonable growth of the number of shapes populated by the method. To anticipate such a growth, in collaboration with Peter Cote and Andy Lorenz [120], we investigated the enumerative properties of RNA shapes at two extreme levels of abstractions, namely considering  $\pi$  (coarsest) and  $\pi'$  (most detailed).

**$\pi$ -shapes.** As illustrated in Figure 5.3,  $\pi$  shapes quite rudely disregard unpaired nucleotides, and contract consecutive helices separated only by an internal loop/bulge. They can then be seen as a well-parenthesized expression, forbidding directly nested (a.k.a. stacking) pairs of parentheses as these would be contracted by the abstraction process. RNA  $\pi$  shapes (except for the shape of size 0, denoted as  $\epsilon$ ) can therefore be generated by the following grammar

$$S^\pi \rightarrow [T] S^\pi \mid [T] \quad T \rightarrow [T] S^\pi \mid \epsilon$$

Solving the associated system, ultimately adding 1 to recover  $\epsilon$ , gives

$$S^\pi(z) \equiv \sum_{n \geq 0} s_n^\pi z^n = \frac{1 - z^2 - \sqrt{1 - 2z^2 - 3z^4}}{2z^2}$$

where  $s_n^\pi$  is the total number of  $\pi$ -shapes over  $n$  characters, from which one draw two immediate conclusions.

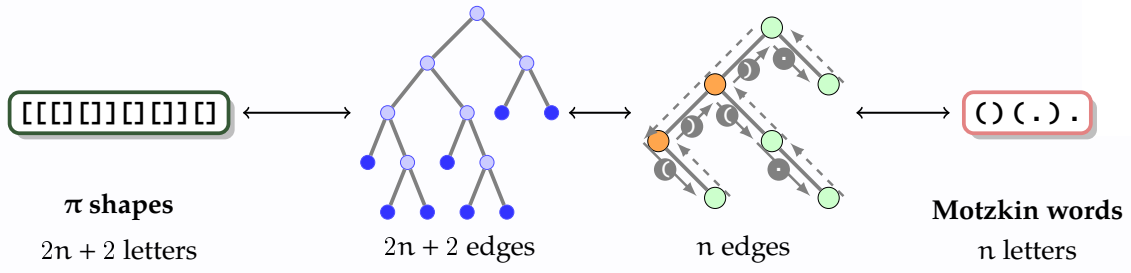


Figure 5.4: **Illustration of bijection between  $\pi$  shapes and Motzkin words.**  $\pi$  shapes of size  $2n + 2$  can be interpreted as Dyck words of same length, and be turned into a binary tree, constrained so that the right child of an internal node cannot be a leaf. Removing leaves, one gets a partial binary tree with  $n$  edges, whose internal nodes must have a right child. Traversing the tree in preorder, emitting a letter  $($  when descending in a right child for the first time, a letter  $.$  for consecutive right descents, and a letter  $)$  when going back up one level, one gets a Motzkin word (OEIS:A001006) of length  $n$ .

**Theorem 21 ( $\pi$ -Shapes (120)):** The number  $s_{2n}^\pi$  of  $\pi$  shapes over  $2n$  characters obeys

$$s_{2n}^\pi = \frac{\sqrt{3}}{2\sqrt{\pi}} \cdot \frac{3^n}{n\sqrt{n}} (1 + \mathcal{O}(1/n))$$

Moreover,  $\pi$ -shapes of size  $2n + 2$  are in bijection with Motzkin words of length  $n$ .

The bijection can be made explicit, and we get:

$$\begin{aligned} \psi : [A]B &\rightarrow \phi(A) \bullet \psi(B) \quad (B \neq \emptyset) & \phi : [A]B &\rightarrow \phi(A) ( \psi(B) ) \\ [A] &\rightarrow \phi(A) & \varepsilon &\rightarrow \varepsilon. \end{aligned}$$

The associated transformation can perhaps be better expressed as a traversal of a tree, as illustrated in by Figure 5.4.

**$\pi'$ -shapes.** At the other end of the shape hierarchy,  $\pi'$  shapes are obtained from secondary structures by **collapsing consecutive unpaired positions** into a single dedicated character  $_$ . The language of  $\pi'$  shapes is then essentially a well-parenthesized expression with an neutral character, i.e. a Motzkin word (OEIS:A001006), excluding:

- Consecutive occurrences  $__$  of unpaired regions;
- Nested/stacking pairs of parentheses  $[[\cdots]]$ .

Expanding the classic Motzkin grammar to capture such constraints gives:

$$S \rightarrow U [T] S \mid U \quad T \rightarrow U [T] U [T] S \mid \_ [T] \_ \mid [T] \_ \mid [T] \_ \mid \varepsilon \quad U \rightarrow \_ \mid \varepsilon$$

Solving the generating function, one gets

$$S^{\pi'}(z) = \frac{A(z) - B(z)\sqrt{\Delta}}{C(z)} \quad \text{with} \quad \Delta := 1 - 2z^2 - 6z^3 - 5z^4 - 6z^5 - z^6 + 2z^7 + z^8,$$

and  $A(z)$ ,  $B(z)$  and  $C(z)$  are polynomials, non-null at  $z = \rho$  the dominant singularity (also smallest root of  $\Delta$ ).

**Theorem 22 ( $\pi'$ -Shapes (120)):** *The number  $s_n^{\pi'}$  of  $\pi'$  shapes over  $n$  characters obeys*

$$s_n^{\pi'} = \kappa \frac{\gamma^n}{n\sqrt{n}} (1 + \mathcal{O}(1/n)) \text{ where } \gamma := \frac{1}{\rho} \approx 2.41 \text{ and } \kappa := \lim_{z \rightarrow \rho} \frac{B(z)}{2\sqrt{\pi}C(z)} \approx 0.985.$$

The chosen level of abstractions for shapes has therefore a strong impact on the number of representatives of the RNA folding landscapes. At the coarsest level, the number of  $\pi$ -shapes only grows in  $\Theta(1.73^n/n\sqrt{n})$ , while the more informative  $\pi'$ -shapes induce a combinatorial explosion in  $\Theta(2.41^n/n\sqrt{n})$ .

**Refined estimates for shapes compatible with a sequence.** The total number of shapes over  $n$  characters represents a very crude upper bound for the expected number of shapes returned by a predictive method when executed on an RNA consisting of  $n$  nucleotides. Indeed, certain sequences may not support the helices required to build a given shape. Ideally, one would therefore like to compute the **expected number of shapes** compatible with a sequence of length  $n$ . However, this turned out to be very challenging, due to apparently intrinsic difficulties to establish a suitable specification.

Indeed, computing an asymptotic equivalent for the expected number of shapes seems beyond the current reach of the symbolic method. For instance, contrary to RNA secondary structures, it does not seem possible to formulate an unambiguous grammars that enumerates all compatible shapes/sequences pairs. This is due to the fact that several secondary structures, all compatible with a given sequence and substantially different, may share a common shape at one or several levels, as shown below:

ACUACAGUGGUAGUACUUUAGAAUGUCUUAGA

$$\begin{aligned} S_1 = .(((...(((.....))))))..((.....)).. &\rightarrow \pi(S_1) = [][] \quad \pi(S_1) = \_ \_ [ \_ ] \_ [ \_ ] \_ \\ S_2 = ((.....)).((.....))..... &\rightarrow \pi(S_2) = [][] \quad \pi(S_2) = [ \_ ] [ \_ ] \_ \\ S_3 = ((((((.....))))))..((((((.....)))))).. &\rightarrow \pi(S_3) = [][] \quad \pi(S_3) = [ \_ ] [ \_ ] \_ \end{aligned}$$

More modestly, it is possible to refine our initial upper bound, by considering the **minimum length of a sequence supporting a given shape**. For instance, hairpin loops, identified by motifs  $[ \ ]$ , are subject to a minimum distance  $\theta$ , and thus can be thought to *consume*  $\theta + 2$  nucleotides. We can capture this phenomenon by replacing each occurrence of  $\varepsilon$  in the above grammars, indicating the occurrence of a hairpin loop by  $\bullet^\theta$ . Summing

over shapes associated with minimum size smaller than  $n$ , one obtains much refined estimates.

**Theorem 23 (Expected number of shapes (120)):** *The expected number  $e_n^\pi$  (resp.  $e_n^{\pi'}$ ) of  $\pi$  shapes (resp.  $\pi'$  shapes), compatible with an RNA of length  $n$  under a min base pairing distance  $\theta = 3$ , obeys:*

$$e_n^\pi \leq \frac{1.28 \times 1.81^n}{n\sqrt{n}}(1 + \mathcal{O}(1/n)) \quad e_n^{\pi'} \leq \frac{2.44 \times 1.32^n}{n\sqrt{n}}(1 + \mathcal{O}(1/n))$$

While these upper bound are considerably tighter than the total number of shapes of length  $n$ , they still appear overly pessimistic in comparison to the empirical observations of Voss *et al* [190]. In this work, the authors conjectured growths in  $\Theta(1.1^n)$  and  $\Theta(1.16^n)$  for  $\pi$  and  $\pi'$  shapes respectively, motivating the development of new analysis techniques for this pragmatic class of conformational descriptors.

#### 5.2.4 Enumerating designable structures (206)

The **inverse folding** of RNA, i.e. the computational search for an RNA sequence that folds stably and preferentially into a **target structure**, is a classic test piece in RNA bioinformatics. Its motivation arises from applications ranging from synthetic biology [182] to RNA therapeutics [203] through systems biology [65] and nanotechnologies [83]. It is NP-hard even under very simple energy models [20], and is typically approached at a practical level through a large collection of methods, based on metaheuristics, constraint programming or random generation paradigms [34].

Since the overarching goal of RNA design is to produce functional molecules, it is tempting to adopt an inverted perspective on the RNA design, and treat the limitations induced by popular sequence generation models as constraints weighing on evolution itself. Indeed, the sequence/structure relationship in RNA is used as a metaphor for the genotype/phenotype map in theoretical evolutionary studies [105] (neutral network theory). In this context, the total [194] or expected [212] number of secondary structures is used as a baseline to quantify the **phenotype space**, the set of functions achieved by a gene/molecule. However, this set is largely overestimated since many structures cannot be adopted as a functional structure, e.g. due to their intrinsic unstability.

In collaboration with Hua-Ting Yao, Cédric Chauve and Mireille Régnier [206], we addressed the quantification of **designable structures**, i.e. structures for which no RNA sequence is deemed acceptable which respect to the objectives of inverse folding. More specifically, following the seminal work of Dirks *et al* [54], notions of **defects** are used as objectives for design. Defect metrics quantify, for a given candidate candidate sequence,

the existence of competitive alternative structures to the target structure. Examples include the **free-energy defect**, the free-energy difference, for a candidate sequence, between the target structure and its best competitor; or the **probably defect**, the overall Boltzmann probability of not adopting the target structure at the thermodynamic equilibrium. This allows to define a **design criterion**, which only accepts sequences whose defect does not exceed some predefined **tolerance**  $\varepsilon > 0$  for the defect  $\mathcal{D}$ .

**Definition 5.2.6 (Designable secondary structure):** Given a defect  $\mathcal{D}$  and a tolerance  $\varepsilon > 0$ , a secondary structure  $S^* \in \mathcal{S}_n$  is **designable** if and only if there exists  $w \in \Sigma^n$  such that  $\mathcal{D}(w, S^*) < \varepsilon$ .

Since their computation requires the joint consideration of: i) a realistic energy model [185]; and ii) the entire space of alternative structures, the computation of defects usually involves the execution of a dynamic programming algorithm. Due to their reliance on dynamic programming, all modern notions of defect are **monotonous with respect to loops**. Namely, as soon as a sequence/structure pair exceeds a certain **defect tolerance** within a (combination of) loop(s), coined a **local obstruction**, it cannot be extended into a larger structure sequence that would meet tolerance. This fact was already observed by Aguirre-Hernández *et al* [1], identifying two local obstructions for the free-energy defect, and by Hales *et al* [86] in a base pair maximization model. We verified the **ubiquity of obstructions**, proposing a *brute force* algorithm running in  $\Theta(\Phi^k 4^k k^3)$ ,  $\Phi = (1 + \sqrt{5})/2$  (a.k.a. the golden ratio), for computing the intrinsic defect of all (combinations of) loops over  $k$  nucleotides, ultimately resulting in a list  $\mathcal{F}$  of local obstructions for a given defect and threshold.

**Remark 5.2.1:** Regardless of the chosen defect, the existence of a polynomial time algorithm for the enumeration all local obstructions would be highly surprising. Indeed, the existence of a complete list of obstructions of size  $n$ , having cardinality polynomial on  $n$ , would suggest a polynomial-time algorithm, using basic motif search in trees, for the decision version of the NP-hard inverse folding problem [20]. Possible alternatives, such as fixed-parameter algorithms, remain a possibility that could be worthy of being explored.

From a collection  $\mathcal{F}$  of local obstructions, the next step is to consider the restricting effect of forbidden motifs on the structure/phenotype space. To that purpose, we adopted a tree representation, where the occurrence of a motif  $m \in \mathcal{F}$  within a structure can be expressed as the exact match of a tree motif associated with  $m$ , as illustrated by Figure 5.5. It is well-known that motifs in trees asymptotically occur a number of time linear on the tree size, almost irrespectively of their precise definition, and that forbidding some motif induces a decay of the exponential growth [35, 39]. The following result ensues immediately.

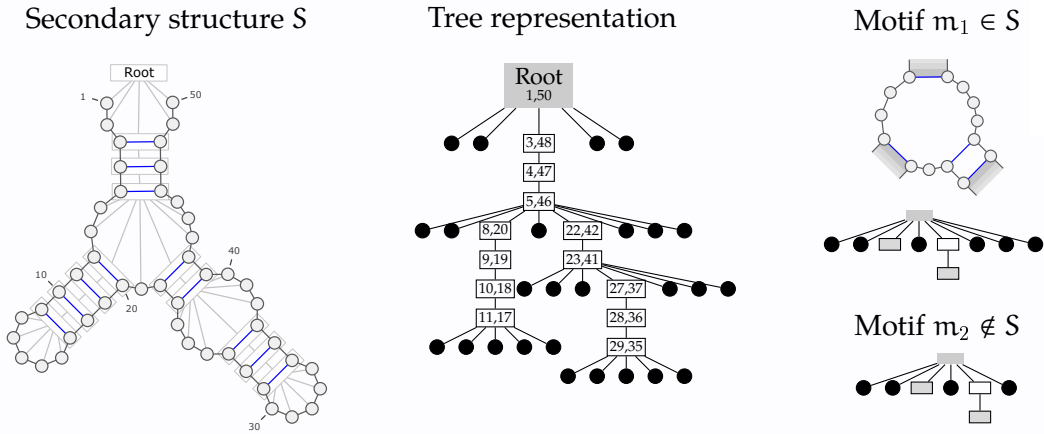


Figure 5.5: **Graph and tree representations of an RNA secondary structure  $S$ .** Our notion of occurrence of a motif in a structure requires an exact match, such that the induced subgraph  $m_2$  of a motif  $m_1 \in S$  does not automatically occur in  $S$ .

**Theorem 24 (Exponentially low density of designable structures (206)):** *For any defect  $D$  and tolerance  $\epsilon$ , the proportion of designable secondary structure in  $S^n$  is exponentially decreasing with  $n$ , i.e. there exists  $\beta < 1$  such that*

$$\frac{|\{S \in S_n \mid S \text{ is designable}\}|}{|S_n|} < \beta^n.$$

To establish this result, the set of secondary structures that avoids any motif in  $\mathcal{F}$  can be used, as it represents a superset for the set of designable structures. It is generated by the grammar:

$$\begin{aligned} S &\rightarrow (T) S \mid \bullet S \mid \epsilon \\ T &\rightarrow S \setminus F \end{aligned}$$

with  $F := \{m' \mid \forall m \in \mathcal{F}, m = (m')\}$ . This translates into the following system:

$$\begin{aligned} S(z) &= z^2 T(z) S(z) + z S(z) + 1 \\ T(z) &= S(z) - F(z, T) \end{aligned}$$

where  $F(z, T)$  is the gen. fun. of enclosed structures within motifs in  $\mathcal{F}$ , such that

$$F(z, T) = \sum_{m \in \mathcal{F}} z^{\gamma(m)} T^{\delta(m)} - c(z, T)$$

where  $\gamma(m)$  (resp.  $\delta(m)$ ) is the size (resp. number of paired leaves) of the motif  $m'$ , and  $c(z, T)$  is a correcting term to account for potential overlaps. Solving the equation numerically gives the asymptotic equivalents listed in Table 5.2.

Defect	#Local		$\rho$	Asymptotic equivalent	Proportion of designable structures* (upper bound)						
	$\varepsilon$	obstructions			Equiv.	$P_{10}$ (%)	$P_{50}$ (%)	$P_{100}$ (%)	$P_{200}$ (%)	$P_{500}$ (%)	$P_{1000}$ (%)
Free-energy	1	104	0.449	$\Theta\left(\frac{2.226^n}{n\sqrt{n}}\right)$	$0.973^n$	76.1	25.4	6.48	$4.19 \cdot 10^{-1}$	$1.14 \cdot 10^{-4}$	$1.30 \cdot 10^{-10}$
Probability	.5	117	0.450	$\Theta\left(\frac{2.224^n}{n\sqrt{n}}\right)$	$0.972^n$	75.3	24.2	5.84	$3.41 \cdot 10^{-1}$	$6.81 \cdot 10^{-5}$	$4.64 \cdot 10^{-11}$
Probability	.1	152	0.460	$\Theta\left(\frac{2.176^n}{n\sqrt{n}}\right)$	$0.95^n$	59.9	7.69	0.59	$3.51 \cdot 10^{-3}$	$7.27 \cdot 10^{-10}$	$5.29 \cdot 10^{-21}$
Probability	.01	174	0.481	$\Theta\left(\frac{2.078^n}{n\sqrt{n}}\right)$	$0.908^n$	38.1	0.80	$6.44 \cdot 10^{-3}$	$4.14 \cdot 10^{-7}$	$1.10 \cdot 10^{-19}$	$1.22 \cdot 10^{-40}$

Table 5.2: **Consequence of obstructions of size up to 12 on proportion of designable secondary structures.** (\*) Proportions of designable sequences computed using an assumption of equal constants for the asymptotic leading terms of the number of secondary structures, respectively allowing and forbidding local obstructions.

## Chapter 6

### Conclusion and perspectives

In this Habilitation, I have attempted to provide a unified perspective over a series of contributions in ensemble dynamic programming, random generation, analysis and design of algorithms with applications in Bioinformatics, focusing on RNA folding from thermodynamic principles and RNA folding.

These research projects, spanning more than a decade of research, more than a dozen of distinct collaborations and 50+ manuscripts published in journals and selective conference, have allowed me to draw fruitful connections between my initial background in computer science and discrete applied mathematics, and my growing expertise and visibility in RNA Bioinformatics. Beyond their absolute value as scientific contributions and tools for the community, they have contributed to build a level of clarity in my objectives, and positioning with respect to my communities, allowing me to (more) confidently envision the future mentoring of young researchers.

**Current projects.** Over the next couple of years, my research will foreseeably be dedicated to two main projects in RNA bioinformatics, namely the **connection between design and RNA evolution** and the **integrative structure prediction informed by probing data**, under respective grants from the French Agence Nationale de la Recherche (DECRYPTED and PARNASSUS projects) awarded for the two projects for the 2020–2024 period.

In the former, we will explore with Simona Cocco, Rémi Monasson and Bruno Sargueil to which extent modern methods in evolutionary analysis, based on the concept of **direct information**, can be used to recover and unravel design principles resulting from negative selective pressures. I also wish to revisit negative design as a null model for assessing the significance of observations, as mentioned in Chapter 3. This will require revisiting the controlled random generation of a NP-hard problem, hopefully helped by my recent enthusiasm for parameterized complexity algorithms.



The latter project, implemented in collaboration with Ronny Lorenz and Bruno Sargueil, attempts to define new methods to exploit information produced in **unconventional settings for RNA probing**. Those include probing experiments in different ionic conditions where pseudoknots are suspected (or new tests to assess their presence), produced using diverse reagents revealing different aspects of RNA architectures or acting with different speeds. One of the requirements for such an endeavor is the design of polynomial DP-based classes of pseudoknots, using conceptual tools introduced in Chapter 2, efficient and focusing on geometrically-feasible structures, enabling reasonable ensemble analyses.

At a more methodological level, I am increasingly convinced that an efficient pursue of finalized projects in RNA Bioinformatics requires to limit the distraction represented by the implementation/debugging of complex DP algorithms (by students having limited experience and patience). For this reason, I plan to invest some of my time to the development of a **generic DP framework** to allow the production of efficient low-level code from a generic DP description. Existing frameworks are indeed either unsufficiently generic, or not easily amenable to support desirable features like random generation. Such a development would also provide motivation to explore connections with formal calculus, from which some of my contributions are already directly originating (through enumerative combinatorics).

Finally, the recent development of methods combining sampling and clustering, following Ding et al. [52] have led us to identify strong limitations in performances, experienced when using both agglomerative and divisive algorithms for unsupervised machine. I am therefore currently venturing in the *darkest pits* of machine learning, hoping to find techniques (e.g. embeddings) that will allow an efficient processing of large sampled sets of structure, hoping to increase the robustness, and reproducibility of downstream analyses.

**Team development.** This decade since my PhD has finally allowed to grow as a scientist, from the postdoc-like status associated with a junior researcher CNRS position, to securing my own research fundings and supervising my student, and finally to taking the acting (2016) and permanent (2017–2018) scientific lead of the AMIBio team at LIX, culminating in the recent recruitment of Sebastian Will.

## Bibliography

- [1] R. Aguirre-Hernández, H. H. Hoos, and A. Condon. Computational RNA secondary structure design: empirical complexity and improved methods. **BMC Bioinformatics**, 8:34, 2007.
- [2] A. V. Aho and M. J. Corasick. Efficient string matching: an aid to bibliographic search. **Communications of the ACM**, 18(6):333–340, jun 1975.
- [3] T. Akutsu. Dynamic programming algorithms for RNA secondary structure prediction with pseudo-knots. **Discrete Applied Mathematics**, 104(1):45 – 62, 2000.
- [4] R. Albert and A.-L. Barabási. Statistical mechanics of complex networks. **Reviews of modern Physics**, 74:47–97, 2002.
- [5] D. Aldous. On the markov chain simulation method for uniform combinatorial distributions and simulated annealing. **Probability in the Engineering and Informational Sciences**, 1(1):33–46, 1987.
- [6] 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**, 2:526–538, Nov. 1985.
- [7] M. Andronescu, A. P. Fejes, F. Hutter, H. H. Hoos, and A. Condon. A new algorithm for RNA secondary structure design. **Journal of Molecular Biology**, 336(3):607–624, 2004.
- [8] M. Andronescu, V. Bereg, H. H. Hoos, and A. Condon. RNA strand: the RNA secondary structure and statistical analysis database. **BMC bioinformatics**, 9:340, Aug. 2008.
- [9] A. Avihoo, A. Churkin, and D. Barash. RNAexinv: An extended inverse RNA folding from shape and physical attributes to sequences. **BMC Bioinformatics**, 12(1):319, 2011.
- [10] J. K. Baker. Trainable grammars for speech recognition. **The Journal of the Acoustical Society of America**, 65(S1):S132–S132, 1979.
- [11] C. Banderier and M. Drmota. Formulae and asymptotics for coefficients of algebraic functions. **Combinatorics, Probability and Computing**, 24(1):1–53, 2015.
- [12] C. Banderier, O. Bodini, Y. Ponty, and H. Tafat. On the diversity of pattern distributions in rational language. In **ANALCO 2012**, pages 107–116, Kyoto, Japan, Jan. 2012. Omnipress.
- [13] E. Barucci, R. Pinzani, and R. Sprugnoli. The random generation of directed animals. **Theoretical Computer Science**, 127(2):333–350, 1994.
- [14] R. Bellman. The theory of dynamic programming. **Bulletin of the American Mathematical Society**, 60(6):503–515, 1954.
- [15] M. Bendkowski, O. Bodini, and S. Dovgal. Polynomial tuning of multiparametric combinatorial samplers. In **2018 Proceedings of the Fifteenth Workshop on Analytic Algorithmics and Combinatorics (ANALCO)**, pages 92–106. SIAM, 2018.
- [16] P. Berenbrink and T. Sauerwald. The weighted coupon collector’s problem and applications. In **15th International Computing and Combinatorics Conference (COCOON’10)**, 2009.
- [17] G. Blin, A. Denise, S. Dulucq, C. Herrbach, and H. Touzet. Alignments of RNA structures. **IEEE/ACM Trans. Comput. Biology Bioinform.**, 7(2):309–322, 2010.
- [18] O. Bodini and Y. Ponty. Multi-dimensional Boltzmann Sampling of Languages. In **AOFA 2010**, volume AM of *DMTCS Proceedings*, pages 49–64, Vienna, Austria, June 2010. Discrete Mathematics and Theoretical Computer Science.

- [19] M. Bon and H. Orland. TT2NE: a novel algorithm to predict RNA secondary structures with pseudo-knots. **Nucleic Acids Research**, 39(14):e93–e93, 05 2011.
- [20] É. Bonnet, P. Rzażewski, and F. Sikora. Designing RNA secondary structures is hard. In **Research in Computational Molecular Biology 22nd Annual International Conference, RECOMB 2018**, volume 10812 of *Lecture Notes in Computer Science*, pages 248–250, Paris, 2018. Springer.
- [21] M. Bousquet-Mélou and Y. Ponty. Culminating paths. **Discrete Mathematics and Theoretical Computer Science**, 10(2):125–152, 2008.
- [22] G. R. Bowman and V. S. Pande. Protein folded states are kinetic hubs. **Proc. Natl. Acad. Sci. U.S.A.**, 107(24):10890–10895, June 2010.
- [23] R. Bundschuh and R. Bruinsma. Melting of branched RNA molecules. **Physical review letters**, 100(14):148101, 2008.
- [24] R. Bundschuh and T. Hwa. RNA secondary structure formation: A solvable model of heteropolymer folding. **Phys. Rev. Lett.**, 83:1479–1482, Aug 1999.
- [25] R. Bundschuh and T. Hwa. Statistical mechanics of secondary structures formed by random RNA sequences. **Physical review. E, Statistical, nonlinear, and soft matter physics**, 65:031903, Mar. 2002.
- [26] R. Bundschuh and T. Hwa. Phases of the secondary structures of RNA sequences. **EPL (Europhysics Letters)**, 59(6):903, 2002.
- [27] H. A. Burbano and E. Andrade. Analysis of tRNA abstract shapes of precursor/derivative amino acids in archaea. **Gene**, 396:75–83, July 2007.
- [28] A. Busch and R. Backofen. INFO-RNA—a fast approach to inverse RNA folding. **Bioinformatics**, 22(15):1823–31, 2006.
- [29] S. Cao and S. Chen. Predicting RNA pseudoknot folding thermodynamics. **Nucleic Acids Research**, 34(9):2634–2652, 2006.
- [30] C. Chauve, Y. Ponty, and J. P. P. Zanetti. Evolution of genes neighborhood within reconciled phylogenies: an ensemble approach. In **BSB 2014**, volume 8826 of *Advances in Bioinformatics and Computational Biology*, pages 49–56, Belo Horizonte, Brazil, Oct. 2014. Springer.
- [31] C. Chauve, Y. Ponty, and J. P. P. Zanetti. Evolution of genes neighborhood within reconciled phylogenies: an ensemble approach. **BMC Bioinformatics**, 16(Suppl 19):S6, Dec. 2015.
- [32] C. Chauve, J. Courtiel, and Y. Ponty. Counting, generating and sampling tree alignments. In **ALCOB 2016**, volume 9702, pages 53–64, Trujillo, Spain, 2016. Springer.
- [33] C. Chauve, J. Courtiel, and Y. Ponty. Counting, generating, analyzing and sampling tree alignments. **International Journal of Foundations of Computer Science**, 29(5):741–767, 2018.
- [34] A. Churkin, M. D. Retwitzer, V. Reinharz, Y. Ponty, J. Waldispühl, and D. Barash. Design of RNAs: comparing programs for inverse RNA folding. **Briefings in Bioinformatics**, 19(2):350–358, Jan. 2018.
- [35] F. Chyzak, M. Drmota, T. Klausner, and G. Kok. The distribution of patterns in random trees. **Combinatorics, Probability and Computing**, 17(1):21–59, 2008.
- [36] P. Clote. Combinatorics of saturated secondary structures of RNA. **Journal of computational biology : a journal of computational molecular cell biology**, 13:1640–1657, Nov. 2006.
- [37] P. Clote, F. Ferré, E. Kranakis, and D. Krizanc. Structural RNA has lower folding energy than random RNA of the same dinucleotide frequency. **RNA (New York, N.Y.)**, 11:578–591, May 2005.
- [38] P. Clote, Y. Ponty, and J.-M. Steyaert. Expected distance between terminal nucleotides of RNA secondary structures. **Journal of Mathematical Biology**, 65(3):581–99, Sept. 2012.
- [39] G. Collet, J. David, and A. Jacquot. Random sampling of ordered trees according to the number of occurrences of a pattern. Submitted, 2018.
- [40] R. Cont and E. Tanimura. Small-world graphs: characterization and alternative constructions. **Adv. in Appl. Probab.**, 40(4):939–965, 2008.
- [41] J. Cooley and J. Tukey. An algorithm for the machine calculation of complex fourier series. **Mathematics of Computation**, 19(90):297–301, 1965.
- [42] J. Cooley, P. Lewis, and P. Welch. The finite Fourier transform. **IEEE Transactions on Audio and Electroacoustics**, 17(2):77–85, June 1969.

- [43] J. Cupal, I. L. Hofacker, and P. F. Stadler. Dynamic programming algorithm for the density of states of RNA secondary structures. In *Proceedings of the German Conference on Bioinformatics, GCB*, pages 184–186, Leipzig, Germany, Sept. 1996.
- [44] K. Darty, A. Denise, and Y. Ponty. VARNAs: Interactive drawing and editing of the RNA secondary structure. *Bioinformatics*, 25(15):1974–5, Aug. 2009.
- [45] N. S. de Groot, A. Armaos, R. Graña-Montes, M. Alriquet, G. Calloni, R. M. Vabulas, and G. G. Tartaglia. RNA structure drives interaction with proteins. *Nature Communications*, 10(1), jul 2019.
- [46] J. Deforges, S. De Breyne, M. Ameur, N. Ulryck, N. Chamond, A. Saaïdi, Y. Ponty, T. Ohlmann, and B. Sargueil. Two ribosome recruitment sites direct multiple translation events within HIV1 Gag open reading frame. *Nucleic Acids Research*, 45(12):7382–7400, July 2017.
- [47] K. E. Deigan, T. W. Li, D. H. Mathews, and K. M. Weeks. Accurate SHAPE-directed RNA structure determination. *Proc Natl Acad Sci U S A*, 106(1):97–102, 2009.
- [48] A. Denise. Génération aléatoire et uniforme de mots. *Discrete Mathematics*, 156:69–84, 1996.
- [49] A. Denise and P. Zimmermann. Uniform random generation of decomposable structures using floating-point arithmetic. *Theoretical Computer Science*, 218(2):233 – 248, 1999.
- [50] A. Denise, Y. Ponty, and M. Termier. Controlled non uniform random generation of decomposable structures. *Theoretical Computer Science*, 411(40-42):3527–3552, 2010.
- [51] Y. Ding and C. E. Lawrence. A statistical sampling algorithm for RNA secondary structure prediction. *Nucleic acids research*, 31:7280–7301, Dec. 2003.
- [52] Y. Ding, C. Y. Chan, and C. E. Lawrence. RNA secondary structure prediction by centroids in a boltzmann weighted ensemble. *RNA (New York, N.Y.)*, 11:1157–1166, Aug. 2005.
- [53] R. Dirks and N. Pierce. A partition function algorithm for nucleic acid secondary structure including pseudoknots. *J Comput Chem*, 24:1664–1677, 2003.
- [54] R. M. Dirks, M. Lin, E. Winfree, and N. A. Pierce. Paradigms for computational nucleic acid design. *Nucleic Acids Research*, 32(4):1392–1403, 2004.
- [55] C. B. Do, D. A. Woods, and S. Batzoglou. Contrafold: RNA secondary structure prediction without physics-based models. *Bioinformatics (Oxford, England)*, 22:e90–e98, July 2006.
- [56] M. Drmota. Systems of functional equations. *Random Structures and Algorithms*, 10(1-2):103–124, 1997.
- [57] M. Drory Retwitzer, V. Reinharz, Y. Ponty, J. Waldispühl, and D. Barash. incaRNAfbinv : a web server for the fragment-based design of RNA sequences. *Nucleic Acids Research*, 44(W1):W308 – W314, 2016.
- [58] J. Du Boisberranger, D. Gardy, and Y. Ponty. The weighted words collector. In *AOFA 2012*, volume AQ, pages 243–264, Montreal, Canada, June 2012. DMTCS.
- [59] W. Duchemin, Y. Anselmetti, M. Patterson, Y. Ponty, S. Bérard, C. Chauve, C. Scornavacca, V. Daubin, and E. Tannier. DeCoSTAR: Reconstructing the ancestral organization of genes or genomes using reconciled phylogenies. *Genome Biology and Evolution*, 9(5):1312–1319, 2017.
- [60] P. Duchon, P. Flajolet, G. Louchard, and G. Schaeffer. Boltzmann samplers for the random generation of combinatorial structures. *Combinatorics, Probability and Computing*, 13(4-5):577–625, 2004.
- [61] J. Duraj. Random walks in cones: The case of nonzero drift. *Stochastic Processes and their Applications*, 124(4):1503–1518, apr 2014.
- [62] S. R. Eddy and R. Durbin. RNA sequence analysis using covariance models. *Nucleic Acids Research*, 22(11):2079–2088, 1994.
- [63] A. Esmaili-Taheri and M. Ganjtabesh. Erd: a fast and reliable tool for RNA design including constraints. *BMC bioinformatics*, 16:20, Jan. 2015.
- [64] A. Esmaili-Taheri, M. Ganjtabesh, and M. Mohammad-Noori. Evolutionary solution for the RNA design problem. *Bioinformatics*, 30(9):1250–1258, 2014.
- [65] S. Findeiß, M. Etzel, S. Will, M. Mörl, and P. F. Stadler. Design of artificial riboswitches as biosensors. *Sensors (Basel, Switzerland)*, 17(9):E1990, Aug. 2017.

- [66] A. V. Finkelstein and M. A. Roytberg. Computation of biopolymers: a general approach to different problems. **Biosystems**, 30(1-3):1–19, 1993.
- [67] P. Flajolet. Analytic models and ambiguity of context-free languages. **Theoretical Computer Science**, 49:283–309, 1987.
- [68] P. Flajolet and A. M. Odlyzko. Singularity analysis of generating functions. **SIAM J. Discrete Math.**, 3(2):216–240, 1990.
- [69] P. Flajolet and R. Sedgewick. *Analytic combinatorics*. cambridge University press, New York, NY, USA, 1 edition, 2009. ISBN 0521898064, 9780521898065.
- [70] P. Flajolet, D. Gardy, and L. Thimonier. Birthday paradox, coupon collectors, caching algorithms and self-organizing search. **Discrete Applied Mathematics**, 39(3):207 – 229, 1992.
- [71] P. Flajolet, P. Zimmermann, and B. V. Cutsem. A calculus for the random generation of labelled combinatorial structures. **Theoretical Computer Science**, 132(1):1 – 35, 1994.
- [72] C. Flamm, W. Fontana, I. Hofacker, and P. Schuster. RNA folding at elementary step resolution. **RNA**, 6:325–338, 2000.
- [73] C. Flamm, I. L. Hofacker, S. Maurer-Stroh, P. F. Stadler, and M. Zehl. Design of multistable RNA molecules. **RNA (New York, N.Y.)**, 7:254–265, Feb 2001.
- [74] W. Fontana, P. F. Stadler, E. G. Bornberg-Bauer, T. Griesmacher, I. L. Hofacker, M. Tacker, P. Tarazona, E. D. Weinberger, and P. Schuster. RNA folding and combinatory landscapes. **Phys. Rev. E**, 47: 2083–2099, Mar 1993.
- [75] E. Freyhult, V. Moulton, and P. Clote. Boltzmann probability of RNA structural neighbors and riboswitch detection. **Bioinformatics (Oxford, England)**, 23:2054–2062, Aug. 2007.
- [76] D. Gamerman and H. F. Lopes. *Markov chain Monte Carlo: stochastic simulation for Bayesian inference*. Chapman and Hall/CRC, 2006.
- [77] J. A. Garcia-Martin, P. Clote, and I. Dotu. RNAiFOLD: a constraint programming algorithm for RNA inverse folding and molecular design. **Journal of Bioinformatics and Computational Biology**, 11(2): 1350001, 2013.
- [78] D. Gardy and Y. Ponty. Weighted random generation of context-free languages: Analysis of collisions in random urn occupancy models. In **GASCOM 2010**, page 14pp, Montréal, Canada, Sept. 2010. LACIM, UQAM.
- [79] Q. Ge and D. Štefankovič. A graph polynomial for independent sets of bipartite graphs. **Combinatorics, Probability and Computing**, 21(05):695–714, 2012.
- [80] R. Giegerich and C. Meyer. Algebraic dynamic programming. In International Conference on Algebraic Methodology and Software Technology, pages 349–364. Springer, 2002.
- [81] R. Giegerich and H. Touzet. Modeling dynamic programming problems over sequences and trees with inverse coupled rewrite systems. **Algorithms**, 7(1):62–144, Mar 2014.
- [82] R. Giegerich, B. Voss, and M. Rehmsmeier. Abstract shapes of RNA. **Nucleic acids research**, 32: 4843–4851, 2004.
- [83] S. Grabbe, H. Haas, M. Diken, L. M. Kranz, P. Langguth, and U. Sahin. Translating nanoparticulate-personalized cancer vaccines into clinical applications: case study with RNA-lipoplexes for the treatment of melanoma. **Nanomedicine (London, England)**, 11:2723–2734, Oct. 2016.
- [84] 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.
- [85] J. Hales, J. Manuch, Y. Ponty, and L. Stacho. Combinatorial RNA Design: Designability and Structure-Approximating Algorithm. In **CPM 2015**, Ischia Island, Italy, June 2015.
- [86] J. Hales, A. Héliou, J. Manuch, Y. Ponty, and L. Stacho. Combinatorial RNA Design: Designability and Structure-Approximating Algorithm in Watson-Crick and Nussinov-Jacobson Energy Models. **Algorithmica**, 79(3):835–856, 2017.
- [87] M. Hamada, H. Kiryu, K. Sato, T. Mituyama, and K. Asai. Prediction of RNA secondary structure using generalized centroid estimators. **Bioinformatics (Oxford, England)**, 25:465–473, Feb. 2009.
- [88] S. Hammer, Y. Ponty, W. Wang, and S. Will. Fixed-Parameter Tractable Sampling for RNA Design with Multiple Target Structures. In **RECOMB 2018**, Paris, France, 2018.



- [89] S. Hammer, C. Günzel, M. Mörl, and S. Findeiß. Evolving methods for rational de novo design of functional RNA molecules. **Methods**, 161:54 – 63, 2019.
- [90] S. Hammer, W. Wang, S. Will, and Y. Ponty. Fixed-parameter tractable sampling for RNA design with multiple target structures. **BMC Bioinformatics**, 20(1):209, Dec. 2019.
- [91] M. Hellmuth, D. Merkle, and M. Middendorf. Extended shapes for the combinatorial design of RNA sequences. **International journal of computational biology and drug design**, 2:371–384, 2009.
- [92] C. Herrbach, A. Denise, and S. Dulucq. Average complexity of the Jiang-Wang-Zhang pairwise tree alignment algorithm and of a RNA secondary structure alignment algorithm. **Theor. Comput. Sci.**, 411(26-28):2423–2432, 2010.
- [93] P. G. Higgs and N. Lehman. The RNA world: molecular cooperation at the origins of life. **Nature Reviews Genetics**, 16(1):7–17, nov 2014.
- [94] F. Hildebrand, A. Meyer, and A. Eyre-Walker. Evidence of selection upon genomic gc-content in bacteria. **PLoS genetics**, 6:e1001107, Sept. 2010.
- [95] I. L. Hofacker, W. Fontana, P. F. Stadler, L. S. Bonhoeffer, M. Tacker, and P. Schuster. Fast folding and comparison of RNA secondary structures. **Monatshefte für Chemie/Chemical Monthly**, 125(2): 167–188, 1994.
- [96] I. L. Hofacker, P. Schuster, and P. F. Stadler. Combinatorics of RNA secondary structures. **Discrete Applied Mathematics**, 88(1-3):207–237, 1998.
- [97] C. Höner zu Siederdisen. Sneaking around concatmap: Efficient combinators for dynamic programming. **SIGPLAN Not.**, 47(9):215–226, Sept. 2012.
- [98] L. Huang and D. Chiang. Better k-best parsing. In Proceedings of the Ninth International Workshop on Parsing Technology, pages 53–64. Association for Computational Linguistics, 2005.
- [99] L. D. Hurst and A. R. Merchant. High guanine-cytosine content is not an adaptation to high temperature: a comparative analysis amongst prokaryotes. **Proceedings. Biological sciences**, 268:493–497, Mar. 2001.
- [100] S. Jeong, M. yang Kao, T. wah Lam, W. kin Sung, and S. ming Yiu. Predicting RNA secondary structures with arbitrary pseudoknots by maximizing the number of stacking pairs. **Journal Of Computational Biology**, 10(6):981–995, 2003.
- [101] E. Jacox, C. Chauve, G. J. Szöllösi, Y. Ponty, and C. Scornavacca. ecceTERA: Comprehensive gene tree-species tree reconciliation using parsimony. **Bioinformatics**, 32(13):2056–2058, July 2016.
- [102] S. Janssen, J. Reeder, and R. Giegerich. Shape based indexing for faster search of RNA family databases. **BMC bioinformatics**, 9:131, Feb. 2008.
- [103] T. Jiang, L. Wang, and K. Zhang. Alignment of trees - an alternative to tree edit. **Theor. Comput. Sci.**, 143(1):137–148, 1995.
- [104] E. Y. Jin and C. M. Reidys. Asymptotic enumeration of RNA structures with pseudoknots. **Bulletin of mathematical biology**, 70:951–970, May 2008.
- [105] T. Jörg, O. C. Martin, and A. Wagner. Neutral network sizes of biological RNA molecules can be computed and are not atypically small. **BMC bioinformatics**, 9:464, Oct. 2008.
- [106] I. Kalvari, J. Argasinska, N. Quinones-Olvera, E. P. Nawrocki, E. Rivas, S. R. Eddy, A. Bateman, R. D. Finn, and A. I. Petrov. Rfam 13.0: shifting to a genome-centric resource for non-coding RNA families. **Nucleic acids research**, 46:D335–D342, Jan. 2018.
- [107] R. Kleinkauf, M. Mann, and R. Backofen. antaRNA: ant colony-based RNA sequence design. **Bioinformatics (Oxford, England)**, 31:3114–3121, Oct. 2015.
- [108] J. W. Klop, M. Bezem, and R. De Vrijer. *Term rewriting systems*. Cambridge University Press, 2001.
- [109] R. V. Koodli, B. Keep, K. R. Coppess, F. Portela, E. participants, and R. Das. Eternabrain: Automated RNA design through move sets and strategies from an internet-scale RNA videogame. **PLOS Computational Biology**, 15(6):1–22, 06 2019.
- [110] M. Kucharík, I. L. Hofacker, P. F. Stadler, and J. Qin. Basin hopping graph: a computational framework to characterize RNA folding landscapes. **Bioinformatics (Oxford, England)**, 30:2009–2017, July 2014.
- [111] D. Lai, J. R. Proctor, and I. M. Meyer. On the importance of cotranscriptional RNA structure formation. **RNA**, 19(11):1461–1473, oct 2013.

- [112] S. P. Lalley. Finite range random walk on free groups and homogeneous trees. **The Annals of Probability**, pages 2087–2130, 1993.
- [113] 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.
- [114] N. B. Leontis and E. Westhof. Geometric nomenclature and classification of RNA base pairs. **RNA**, 7(4):499–512, 2001.
- [115] A. Lescoute, N. B. Leontis, C. Massire, and E. Westhof. Recurrent structural RNA motifs, isostericity matrices and sequence alignments. **Nucleic acids research**, 33:2395–2409, 2005.
- [116] A. Levin, M. Lis, Y. Ponty, C. W. O'Donnell, S. Devadas, B. Berger, and J. Waldispühl. A global sampling approach to designing and reengineering RNA secondary structures. **Nucleic Acids Research**, 40(20):10041–52, 2012.
- [117] A. Levin, M. Lis, Y. Ponty, C. W. O'Donnell, S. Devadas, B. Berger, and J. Waldispühl. A global sampling approach to designing and reengineering RNA secondary structures. **Nucleic Acids Research**, 40(20):10041–10052, Nov. 2012.
- [118] A. Lorenz and Y. Ponty. Non-redundant random generation algorithms for weighted context-free languages. **Theoretical Computer Science**, 502:177–194, Sept. 2013.
- [119] R. Lorenz, C. Flamm, and I. L. Hofacker. 2d projections of RNA folding landscapes. In German Conference on Bioinformatics 2009, pages 11–20, Martin Luther University Halle-Wittenberg, Germany, Sept. 2009.
- [120] W. A. Lorenz, P. Clote, and Y. Ponty. Asymptotics of RNA shapes. **Journal of Computational Biology**, 15(1):31–63, 2008.
- [121] Z. J. Lu, J. W. Gloor, and D. H. Mathews. Improved RNA secondary structure prediction by maximizing expected pair accuracy. **RNA**, 15(10):1805–1813, Oct. 2009.
- [122] J. Lumbroso, M. Mishna, and Y. Ponty. Taming reluctant random walks in the positive quadrant. In **GASCOM 2016**, volume 59 of *Electronic Notes in Discrete Mathematics*, pages 99 – 114, Bastia, France, 2017.
- [123] R. B. Lyngsø and C. N. Pedersen. RNA pseudoknot prediction in energy-based models. **Journal of computational biology : a journal of computational molecular cell biology**, 7:409–427, 2000.
- [124] R. Lyngsø. Complexity of pseudoknot prediction in simple models. In Proceedings of ICALP, 2004.
- [125] R. B. Lyngsø, J. W. Anderson, E. Sizikova, A. Badugu, T. Hyland, and J. Hein. FRNAkenstein: multiple target inverse RNA folding. **BMC Bioinformatics**, 13:260, 2012.
- [126] J. Mañuch, C. Thachuk, L. Stacho, and A. Condon. NP-completeness of the energy barrier problem without pseudoknots and temporary arcs. **Natural Computing**, 10(1):391–405, nov 2010.
- [127] D. H. Mathews. Using an RNA secondary structure partition function to determine confidence in base pairs predicted by free energy minimization. **RNA**, 10(8):1178–1190, jul 2004.
- [128] D. H. Mathews and D. H. Turner. Dynalign: an algorithm for finding the secondary structure common to two RNA sequences. **Journal of Molecular Biology**, 317(2):191–203, mar 2002.
- [129] 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. **Proc Natl Acad Sci U S A**, 101(19):7287–92, 2004.
- [130] J. S. McCaskill. The equilibrium partition function and base pair binding probabilities for RNA secondary structure. **Biopolymers: Original Research on Biomolecules**, 29(6-7):1105–1119, 1990.
- [131] Z. Miao, RNA-Puzzles Consortium, and E. Westhof. RNA-Puzzles Round III: 3D RNA structure prediction of five riboswitches and one ribozyme. **RNA**, 23:655–672, 2017.
- [132] J. Michalik. *Non-redundant sampling in RNA bioinformatics*. PhD thesis, Interface Graduate school – Université Paris-Saclay, 2019.
- [133] J. Michalik, H. Touzet, and Y. Ponty. Efficient approximations of RNA kinetics landscape using non-redundant sampling. In **ISMB/ECCB 2017**, volume 33, pages i283 – i292, Prague, Czech Republic, 2017.
- [134] F. Michel and E. Westhof. Modelling of the three-dimensional architecture of group i catalytic introns based on comparative sequence analysis. **Journal of Molecular Biology**, 216(3):585–610, dec 1990.

- [135] I. Miklós. *Computational Complexity of Counting and Sampling*. Discrete Mathematics and Its Applications. Chapman and Hall/CRC, feb 2019.
- [136] I. Miklós, I. M. Meyer, and B. Nagy. Moments of the boltzmann distribution for RNA secondary structures. **Bull Math Biol**, 67(5):1031–1047, Sep 2005.
- [137] M. Mishna and A. Rechnitzer. Two non-holonomic lattice walks in the quarter plane. **Theoretical Computer Science**, 410(38-40):3616–3630, 2009.
- [138] M. Mohri. Semiring frameworks and algorithms for shortest-distance problems. **J. Autom. Lang. Comb.**, 7(3):321–350, Jan. 2002.
- [139] M. E. Nebel. Combinatorial properties of RNA secondary structures. **Journal of Computational Biology**, 9(3):541–573, 2002.
- [140] Y. Nesterov and A. Nemirovskii. *Interior-point polynomial algorithms in convex programming*, volume 13. Siam, 1994.
- [141] M. E. Newman, S. H. Strogatz, and D. J. Watts. Random graphs with arbitrary degree distributions and their applications. **Phys. Rev. E**, 64(2):026118, August 2001.
- [142] C. Nicaud and D. Gouyou-Beauchamps. Random generation using binomial approximations. **Discrete Mathematics & Theoretical Computer Science**, 2010.
- [143] R. Nussinov and A. B. Jacobson. Fast algorithm for predicting the secondary structure of single-stranded RNA. **Proceedings of the National Academy of Sciences of the United States of America**, 77:6309–6313, Nov. 1980.
- [144] C. Pivoteau. *Génération aléatoire de structures combinatoires : méthode de Boltzmann effective*. phdthesis, Université Pierre et Marie Curie - Paris VI, 2008.
- [145] C. Pivoteau, B. Salvy, and M. Soria. Algorithms for combinatorial structures: Well-founded systems and newton iterations. **Journal of Combinatorial Theory, Series A**, 119(8):1711–1773, nov 2012.
- [146] Y. Ponty. *Models for structured genomic sequences, random generation and applications*. Theses, Université Paris Sud - Paris XI, Nov. 2006.
- [147] Y. Ponty. Non-redundant random generation from weighted context-free languages. In **GASCOM 2008**, page 12pp, Bibbiena, Italy, 2008.
- [148] Y. Ponty. Efficient sampling of RNA secondary structures from the Boltzmann ensemble of low-energy: The boustrophedon method. **Journal of Mathematical Biology**, 56(1-2):107–127, 2008.
- [149] Y. Ponty. Rule-weighted and terminal-weighted context-free grammars have identical expressivity. Research report, 2012.
- [150] Y. Ponty and C. Saule. A Combinatorial Framework for Designing (Pseudoknotted) RNA Algorithms. In **WABI 2011**, Saarbrücken, Germany, 2011.
- [151] Y. Ponty, M. Termier, and A. Denise. GenRGenS: Software for Generating Random Genomic Sequences and Structures. **Bioinformatics**, 22(12):1534–1535, 2006.
- [152] J. G. Propp and D. B. Wilson. Exact sampling with coupled markov chains and applications to statistical mechanics. **Random Structures & Algorithms**, 9(1-2):223–252, 1996.
- [153] A. Rajaraman, C. Chauve, and Y. Ponty. Assessing the robustness of parsimonious predictions for gene neighborhoods from reconciled phylogenies. In **ISBRA 2015**, volume 9096, pages 260–271, Norfolk, Virginia, United States, June 2015.
- [154] J. Reeder and R. Giegerich. Design, implementation and evaluation of a practical pseudoknot folding algorithm based on thermodynamics. **BMC Bioinformatics**, 5:104, 2004.
- [155] J. Reeder and R. Giegerich. Consensus shapes: an alternative to the sankoff algorithm for RNA consensus structure prediction. **Bioinformatics (Oxford, England)**, 21:3516–3523, Sept. 2005.
- [156] 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, Apr 2011.
- [157] V. Reinharz, Y. Ponty, and J. Waldispühl. A linear inside-outside algorithm for correcting sequencing errors in structured RNA sequences. In **RECOMB 2013**, Beijing, China, Apr. 2013.



- [158] V. Reinharz, Y. Ponty, and J. Waldispühl. A weighted sampling algorithm for the design of RNA sequences with targeted secondary structure and nucleotides distribution. In **ISMB/ECCB 2013**, Berlin, Germany, July 2013.
- [159] V. Reinharz, Y. Ponty, and J. Waldispühl. Using Structural and Evolutionary Information to Detect and Correct Pyrosequencing Errors in Noncoding RNAs. **Journal of Computational Biology**, 20(11): 905–19, Nov. 2013.
- [160] V. Reinharz, Y. Ponty, and J. Waldispühl. A weighted sampling algorithm for the design of RNA sequences with targeted secondary structure and nucleotide distribution. **Bioinformatics**, 29(13): i308–15, July 2013.
- [161] V. Reinharz, Y. Ponty, and J. Waldispühl. Combining structure probing data on RNA mutants with evolutionary information reveals RNA-binding interfaces. **Nucleic Acids Research**, 44(11):e104–e104, 2016.
- [162] K.-J. Räihä and E. Ukkonen. The shortest common supersequence problem over binary alphabet is NP-complete. **Theoretical Computer Science**, 16(2):187–198, 1981.
- [163] P. Rinaudo, Y. Ponty, D. Barth, and A. Denise. Tree decomposition and parameterized algorithms for RNA structure-sequence alignment including tertiary interactions and pseudoknots. In **WABI 2012**, tba, Ljubljana, Slovenia, Sept. 2012. University of Ljubljana.
- [164] E. Rivas and S. Eddy. A dynamic programming algorithm for RNA structure prediction including pseudoknots. **J Mol Biol**, 285:2053–2068, 1999.
- [165] C. Rovetta, J. Michálik, R. Lorenz, A. Tanzer, and Y. Ponty. Non-Redundant Sampling and Statistical Estimators for RNA Structural Properties at the Thermodynamic Equilibrium. working paper or preprint, Sept. 2019.
- [166] F. Runge, D. Stoll, S. Falkner, and F. Hutter. Learning to design RNA. In **International Conference on Learning Representations**, 2019.
- [167] D. Sankoff. Simultaneous solution of the RNA folding, alignment and protosequence problems. **SIAM Journal on Applied Mathematics**, 45(5):810–825, 1985.
- [168] C. Saule, M. Régnier, J.-M. Steyaert, and A. Denise. Counting RNA pseudoknotted structures. **Journal of computational biology : a journal of computational molecular cell biology**, 18:1339–1351, Oct. 2011.
- [169] G. Sauthoff, M. Möhl, S. Janssen, and R. Giegerich. Bellman’s gap—a language and compiler for dynamic programming in sequence analysis. **Bioinformatics**, 29(5):551–560, 2013.
- [170] A. Scala, L. Nunes Amaral, and M. Barthélémy. Small-world networks and the conformation space of a short lattice polymer chain. **Europhys. Lett.**, 55(4):594–600, 2001.
- [171] S. Schirmer and R. Giegerich. Forest alignment with affine gaps and anchors, applied in RNA structure comparison. **Theor. Comput. Sci.**, 483:51–67, 2013.
- [172] M. Schnall-Levin, L. Chindelevitch, and B. Berger. Inverting the Viterbi algorithm: an abstract framework for structure design. In **Machine Learning, Proceedings of the Twenty 9, 2008**, pages 904–911, 2008.
- [173] E. Senter, S. Sheikh, I. Dotu, Y. Ponty, and P. Clote. Using the Fast Fourier Transform to Accelerate the Computational Search for RNA Conformational Switches. **PLoS ONE**, 7(12):e50506, Dec. 2012.
- [174] E. Senter, S. Sheikh, I. Dotu, Y. Ponty, and P. Clote. Using the Fast Fourier Transform to accelerate the computational search for RNA conformational switches (extended abstract). In **RECOMB 2013**, Beijing, China, Apr. 2013.
- [175] S. Sheikh, R. Backofen, and Y. Ponty. Impact Of The Energy Model On The Complexity Of RNA Folding With Pseudoknots. In **CPM 2012**, volume 7354 of *Combinatorial Pattern Matching*, pages 321–333, Helsinki, Finland, July 2012. Juha Kärkkäinen, Springer.
- [176] M. J. Smola, G. M. Rice, S. Busan, N. A. Siegfried, and K. M. Weeks. Selective 2′-hydroxyl acylation analyzed by primer extension and mutational profiling (SHAPE-MaP) for direct, versatile and accurate RNA structure analysis. **Nat Protoc**, 10:1643–1669, 2015.
- [177] A. Spasic, S. M. Assmann, P. C. Bevilacqua, and D. H. Mathews. Modeling RNA secondary structure folding ensembles using SHAPE mapping data. **Nucleic Acids Res**, 46(1):314–323, 2017.

- [178] J. Stombaugh, C. L. Zirbel, E. Westhof, and N. B. Leontis. Frequency and isostericity of RNA base pairs. **Nucleic acids research**, 37:2294–2312, Apr. 2009.
- [179] D. Sundfeld, J. H. Havgaard, A. C. M. A. de Melo, and J. Gorodkin. Foldalign 2.5: multithreaded implementation for pairwise structural RNA alignment. **Bioinformatics**, 32(8):1238–1240, dec 2015.
- [180] D. Surujon, Y. Ponty, and P. Clote. Small-world networks and RNA secondary structures. **Journal of computational biology : a journal of computational molecular cell biology**, 26(1):16–26, Jan. 2019.
- [181] J. E. Tabaska, R. B. Cary, H. N. Gabow, and G. D. Stormo. An RNA folding method capable of identifying pseudoknots and base triples. **Bioinformatics**, 14(8):691–699, 1998.
- [182] M. K. Takahashi and J. B. Lucks. A modular strategy for engineering orthogonal chimeric RNA transcription regulators. **Nucleic Acids Research**, 41(15):7577–7588, 2013.
- [183] C. Theis, S. Janssen, and R. Giegerich. Prediction of RNA secondary structure including kissing hairpin motifs. In *Algorithms in Bioinformatics*, pages 52–64, Berlin, Heidelberg, 2010. Springer Berlin Heidelberg. ISBN 978-3-642-15294-8.
- [184] I. Tinoco and C. Bustamante. How RNA folds. **J Mol Biol**, 293(2):271–81, Oct 1999.
- [185] D. H. Turner and D. H. Mathews. NNDB: the nearest neighbor parameter database for predicting stability of nucleic acid secondary structure. **Nucleic acids research**, 38(Database issue):D280–D282, Jan. 2010.
- [186] V. Van Noort, B. Snel, and M. A. Huynen. The yeast coexpression network has a small-world, scale-free architecture and can be explained by a simple model. **EMBO Rep.**, 5(3):280–284, March 2004.
- [187] G. Viennot and M. Vauchassade de Chaumont. Enumeration of RNA secondary structures by complexity. In *Mathematics in Biology and Medicine*, pages 360–365, Berlin, Heidelberg, 1985. Springer Berlin Heidelberg. ISBN 978-3-642-93287-8.
- [188] A. Viterbi. Error bounds for convolutional codes and an asymptotically optimum decoding algorithm. **IEEE Transactions on Information Theory**, 13(2):260–269, apr 1967.
- [189] F. Voisin and M.-C. Gaudel. Drawing uniformly at random in dynamic sets of paths. HAL preprint, Oct. 2019.
- [190] B. Voss, R. Giegerich, and M. Rehmsmeier. Complete probabilistic analysis of RNA shapes. **BMC biology**, 4:5, Feb. 2006.
- [191] J. Waldispühl, S. Devadas, B. Berger, and P. Clote. Efficient algorithms for probing the RNA mutation landscape. **PLoS computational biology**, 4:e1000124, Aug. 2008.
- [192] J. Waldispühl and Y. Ponty. An unbiased adaptive sampling algorithm for the exploration of RNA mutational landscapes under evolutionary pressure. In **RECOMB 2011**, volume 6577 of *Lecture Notes in Computer Science*, pages 501–515, Vancouver, Canada, Mar. 2011. Springer Berlin / Heidelberg.
- [193] J. Waldispühl and Y. Ponty. An unbiased adaptive sampling algorithm for the exploration of RNA mutational landscapes under evolutionary pressure. **Journal of Computational Biology**, 18(11): 1465–79, Nov. 2011.
- [194] M. Waterman. Secondary structure of single-stranded nucleic acids. **Advances in Mathematics: Supplementary Studies**, 1:167–212, 1978.
- [195] D. J. Watts and S. H. Strogatz. Collective dynamics of ‘small-world’ networks. **Nature**, 393(6684): 440–442, June 1998.
- [196] S. E. Wells, P. E. Hillner, R. D. Vale, and A. B. Sachs. Circularization of mRNA by eukaryotic translation initiation factors. **Molecular cell**, 2:135–140, July 1998.
- [197] H. S. Wilf. A unified setting for sequencing, ranking, and selection algorithms for combinatorial objects. **Advances in Mathematics**, 24:281–291, 1977.
- [198] K. A. Wilkinson, E. J. Merino, and K. M. Weeks. Selective 2'-hydroxyl acylation analyzed by primer extension (SHAPE): quantitative RNA structure analysis at single nucleotide resolution. **Nat Protoc**, 1:1610–1616, 2006.
- [199] S. Will, T. Joshi, I. L. Hofacker, P. F. Stadler, and R. Backofen. LocaRNA-p: accurate boundary prediction and improved detection of structural RNAs. **RNA (New York, N.Y.)**, 18:900–914, May 2012.

- [200] S. Will, C. Otto, M. Miladi, M. Möhl, and R. Backofen. Sparse: quadratic time simultaneous alignment and folding of RNAs without sequence-based heuristics. **Bioinformatics (Oxford, England)**, 31: 2489–2496, Aug. 2015.
- [201] M. T. Wolfinger, W. A. Svrcek-Seiler, C. Flamm, I. L. Hofacker, and P. F. Stadler. Efficient computation of RNA folding dynamics. **Journal of Physics A: Mathematical and General**, 37(17):4731–4741, apr 2004.
- [202] A. R. Woods. Coloring rules for finite trees, and probabilities of monadic second order sentences. **Random Structures & Algorithms**, 10(4):453–485, 1997.
- [203] S. Y. Wu, G. Lopez-Berestein, G. A. Calin, and A. K. Sood. RNAi therapies: Drugging the undruggable. **Science Translational Medicine**, 6(240):240ps7, 2014.
- [204] S. Wuchty. Small worlds in RNA structures. **Nucleic. Acids. Res.**, 31(3):1108–1117, February 2003.
- [205] A. Xayaphoummine, T. Bucher, F. Thalmann, and H. Isambert. Prediction and statistics of pseudoknots in RNA structures using exactly clustered stochastic simulations. **Proc. Natl. Acad. Sci. U. S. A.**, 100(26):15310–15315, 2003.
- [206] H.-T. Yao, C. Chauve, M. Regnier, and Y. Ponty. Exponentially few RNA structures are designable. In **ACM-BCB 2019**, pages 289–298, Niagara-Falls, United States, 2019. ACM Press.
- [207] A. M. Yoffe, P. Prinsen, W. M. Gelbart, and A. Ben-Shaul. The ends of a large RNA molecule are necessarily close. **Nucleic acids research**, 39:292–299, Jan. 2011.
- [208] J. N. Zadeh, B. R. Wolfe, and N. A. Pierce. Nucleic acid sequence design via efficient ensemble defect optimization. **Journal of Computational Chemistry**, 32(3):439–52, 2011.
- [209] Y. Zhang, Y. Ponty, M. Blanchette, E. Lecuyer, and J. Waldispühl. SPARCS: a web server to analyze (un)structured regions in coding RNA sequences. **Nucleic Acids Research**, 41(Web Server issue): W480–5, July 2013.
- [210] Y. Zhou, Y. Ponty, S. Vialette, J. Waldispühl, Y. Zhang, and A. Denise. Flexible RNA design under structure and sequence constraints using formal languages. In **ACM-BCB 2013**, Bethesda, Washington DC, United States, Sept. 2013.
- [211] C. H. zu Siederdissen, S. J. Prohaska, and P. F. Stadler. Algebraic dynamic programming over general data structures. **BMC bioinformatics**, 16(19):S2, 2015.
- [212] M. Zuker and D. Sankoff. RNA secondary structures and their prediction. **Bulletin of mathematical biology**, 46(4):591–621, 1984.
- [213] M. Zuker and P. Stiegler. Optimal computer folding of large RNA sequences using thermodynamics and auxiliary information. **Nucleic Acids Research**, 9(1):133–148, 1981.